### Contents

**Chapter 8: CRUD-Based Calls**
- `deploy()` ............................................. 46
- `checkDeployStatus()` .................................. 55
- `cancelDeploy()` ........................................ 55
- `deployRecentValidation()` ................................ 59
- `retrieve()` ........................................ 64
- `checkRetrieveStatus()` ................................... 70

**Chapter 9: Utility Calls**
- `createMetadata()` ........................................ 73
- `readMetadata()` ......................................... 76
- `updateMetadata()` ....................................... 77
- `upsertMetadata()` ....................................... 80
- `deleteMetadata()` ....................................... 82
- `renameMetadata()` ....................................... 84
- `create()` ........................................... 86
- `delete()` ........................................... 87
- `update()` ........................................... 89

**Chapter 10: Result Objects**
- `AsyncResult` .......................................... 101
- `CancelDeployResult` .................................. 105
- `DeployResult` .......................................... 105
- `DescribeMetadataResult` .......................... 113
- `DescribeValueTypeResult` ......................... 114
- `ReadResult` ........................................... 116
- `RetrieveResult` ....................................... 116
- `SaveResult` .......................................... 118
- `DeleteResult` .......................................... 119
- `UpsertResult` .......................................... 119
- `Error` ............................................... 120

**Chapter 11: Metadata Types**
- `Metadata Components and Types` ................ 121
- `Metadata Coverage Report` ....................... 122
- `Unsupported Metadata Types` ...................... 122
- `Special Behavior in Metadata API Deployments` 124
<table>
<thead>
<tr>
<th>Custom Metadata Types (CustomObject)</th>
<th>283</th>
</tr>
</thead>
<tbody>
<tr>
<td>CustomMetadata</td>
<td>286</td>
</tr>
<tr>
<td>ArticleType</td>
<td>136</td>
</tr>
<tr>
<td>ArticleType Layout</td>
<td>139</td>
</tr>
<tr>
<td>ChannelLayout</td>
<td>141</td>
</tr>
<tr>
<td>ArticleType CustomField</td>
<td>142</td>
</tr>
<tr>
<td>ApexClass</td>
<td>145</td>
</tr>
<tr>
<td>ApexComponent</td>
<td>148</td>
</tr>
<tr>
<td>ApexPage</td>
<td>149</td>
</tr>
<tr>
<td>ApexTestSuite</td>
<td>151</td>
</tr>
<tr>
<td>ApexTrigger</td>
<td>152</td>
</tr>
<tr>
<td>AppMenu</td>
<td>154</td>
</tr>
<tr>
<td>ApprovalProcess</td>
<td>157</td>
</tr>
<tr>
<td>AssignmentRules</td>
<td>168</td>
</tr>
<tr>
<td>Audience</td>
<td>171</td>
</tr>
<tr>
<td>AuraDefinitionBundle</td>
<td>175</td>
</tr>
<tr>
<td>AuthProvider</td>
<td>178</td>
</tr>
<tr>
<td>AutoResponseRules</td>
<td>183</td>
</tr>
<tr>
<td>Bot</td>
<td>186</td>
</tr>
<tr>
<td>BotVersion</td>
<td>192</td>
</tr>
<tr>
<td>BrandingSet</td>
<td>203</td>
</tr>
<tr>
<td>CallCenter</td>
<td>205</td>
</tr>
<tr>
<td>CampaignInfluenceModel</td>
<td>208</td>
</tr>
<tr>
<td>CaseSubjectParticle</td>
<td>209</td>
</tr>
<tr>
<td>Certificate</td>
<td>211</td>
</tr>
<tr>
<td>ChatterExtension</td>
<td>212</td>
</tr>
<tr>
<td>CleanDataService</td>
<td>214</td>
</tr>
<tr>
<td>CMSConnectSource</td>
<td>218</td>
</tr>
<tr>
<td>Community (Zone)</td>
<td>226</td>
</tr>
<tr>
<td>CommunityTemplateDefinition</td>
<td>229</td>
</tr>
<tr>
<td>CommunityThemeDefinition</td>
<td>235</td>
</tr>
<tr>
<td>ConnectedApp</td>
<td>239</td>
</tr>
<tr>
<td>ContentAsset</td>
<td>249</td>
</tr>
<tr>
<td>CorsWhitelistOrigin</td>
<td>252</td>
</tr>
<tr>
<td>CspTrustedSite</td>
<td>253</td>
</tr>
<tr>
<td>CustomApplication</td>
<td>255</td>
</tr>
<tr>
<td>CustomApplicationComponent</td>
<td>276</td>
</tr>
<tr>
<td>CustomFeedFilter</td>
<td>277</td>
</tr>
<tr>
<td>CustomHelpMenuSection</td>
<td>279</td>
</tr>
<tr>
<td>CustomLabels</td>
<td>281</td>
</tr>
<tr>
<td>Custom Metadata Types (CustomObject)</td>
<td>283</td>
</tr>
<tr>
<td>CustomMetadata</td>
<td>286</td>
</tr>
<tr>
<td>Contents</td>
<td>Page</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>CustomNotificationType</td>
<td>291</td>
</tr>
<tr>
<td>CustomObject</td>
<td>292</td>
</tr>
<tr>
<td>ActionOverride</td>
<td>304</td>
</tr>
<tr>
<td>BusinessProcess</td>
<td>307</td>
</tr>
<tr>
<td>CompactLayout</td>
<td>309</td>
</tr>
<tr>
<td>CustomField</td>
<td>311</td>
</tr>
<tr>
<td>FieldSet</td>
<td>323</td>
</tr>
<tr>
<td>HistoryRetentionPolicy</td>
<td>324</td>
</tr>
<tr>
<td>Index</td>
<td>325</td>
</tr>
<tr>
<td>ListView</td>
<td>328</td>
</tr>
<tr>
<td>NamedFilter</td>
<td>331</td>
</tr>
<tr>
<td>Picklist (Including Dependent Picklist)</td>
<td>334</td>
</tr>
<tr>
<td>RecordType</td>
<td>338</td>
</tr>
<tr>
<td>SearchLayouts</td>
<td>340</td>
</tr>
<tr>
<td>SharingReason</td>
<td>343</td>
</tr>
<tr>
<td>SharingRecalculation</td>
<td>344</td>
</tr>
<tr>
<td>ValidationRule</td>
<td>345</td>
</tr>
<tr>
<td>WebLink</td>
<td>347</td>
</tr>
<tr>
<td>Metadata Field Types</td>
<td>351</td>
</tr>
<tr>
<td>CustomObjectTranslation</td>
<td>354</td>
</tr>
<tr>
<td>CustomPageWebLink</td>
<td>364</td>
</tr>
<tr>
<td>CustomPermission</td>
<td>367</td>
</tr>
<tr>
<td>CustomSite</td>
<td>369</td>
</tr>
<tr>
<td>CustomTab</td>
<td>375</td>
</tr>
<tr>
<td>CustomValue</td>
<td>379</td>
</tr>
<tr>
<td>Dashboard</td>
<td>382</td>
</tr>
<tr>
<td>DataCategoryGroup</td>
<td>403</td>
</tr>
<tr>
<td>DelegateGroup</td>
<td>408</td>
</tr>
<tr>
<td>Document</td>
<td>410</td>
</tr>
<tr>
<td>DuplicateRule</td>
<td>412</td>
</tr>
<tr>
<td>EclairGeoData</td>
<td>417</td>
</tr>
<tr>
<td>EmailServicesFunction</td>
<td>419</td>
</tr>
<tr>
<td>EmailTemplate</td>
<td>424</td>
</tr>
<tr>
<td>EmbeddedServiceBranding</td>
<td>428</td>
</tr>
<tr>
<td>EmbeddedServiceConfig</td>
<td>430</td>
</tr>
<tr>
<td>EmbeddedServiceFieldService</td>
<td>433</td>
</tr>
<tr>
<td>EmbeddedServiceFlowConfig</td>
<td>434</td>
</tr>
<tr>
<td>EmbeddedServiceLiveAgent</td>
<td>435</td>
</tr>
<tr>
<td>EntitlementProcess</td>
<td>439</td>
</tr>
<tr>
<td>EntitlementTemplate</td>
<td>443</td>
</tr>
<tr>
<td>EscalationRules</td>
<td>444</td>
</tr>
<tr>
<td>EventDelivery</td>
<td>447</td>
</tr>
<tr>
<td>EventSubscription</td>
<td>449</td>
</tr>
<tr>
<td>ExperienceBundle</td>
<td>451</td>
</tr>
</tbody>
</table>
ExternalDataSource ................................................................. 465
ExternalServiceRegistration ..................................................... 470
FeatureParameterBoolean ........................................................ 471
FeatureParameterDate .............................................................. 473
FeatureParameterInteger ......................................................... 475
FlexiPage ............................................................................. 477
Flow .................................................................................... 490
FlowCategory ........................................................................... 525
FlowDefinition ........................................................................ 526
Folder ...................................................................................... 527
    FolderShare ................................................................. 529
GlobalPicklist ........................................................................... 531
GlobalPicklistValue ................................................................. 533
GlobalValueSet .......................................................................... 536
GlobalValueSetTranslation ....................................................... 537
Group ..................................................................................... 539
HomePageComponent .............................................................. 540
HomePageLayout ................................................................. 542
InstalledPackage ................................................................. 543
KeywordList .............................................................................. 544
Layout .................................................................................. 546
Letterhead ............................................................................... 567
LightningBolt ........................................................................... 570
LightningComponentBundle ..................................................... 573
LightningExperienceTheme ....................................................... 576
LiveChatAgentConfig .............................................................. 577
LiveChatButton ................................................................. 581
LiveChatDeployment ............................................................. 586
LiveChatSensitiveDataRule ...................................................... 588
ManagedTopics ........................................................................ 590
MatchingRule ........................................................................... 593
Metadata ................................................................................ 595
MetadataWithContent ........................................................... 596
MilestoneType ........................................................................... 597
MIDomain (Beta) ...................................................................... 598
ModerationRule ........................................................................ 602
NamedCredential ................................................................. 606
Network .................................................................................. 609
NetworkBranding ................................................................. 629
Package .................................................................................. 631
PathAssistant ........................................................................... 633
PermissionSet ........................................................................... 635
PermissionSetGroup (Pilot) ....................................................... 642
PlatformCachePartition ......................................................... 644
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>MacroSettings</td>
<td>807</td>
</tr>
<tr>
<td>MapAndLocationSettings</td>
<td>808</td>
</tr>
<tr>
<td>MobileSettings</td>
<td>809</td>
</tr>
<tr>
<td>NameSettings</td>
<td>812</td>
</tr>
<tr>
<td>NotificationsSettings</td>
<td>813</td>
</tr>
<tr>
<td>OauthCustomScope</td>
<td>814</td>
</tr>
<tr>
<td>OmniChannelSettings</td>
<td>816</td>
</tr>
<tr>
<td>OpportunitySettings</td>
<td>817</td>
</tr>
<tr>
<td>OrderSettings</td>
<td>818</td>
</tr>
<tr>
<td>OrgPreferenceSettings</td>
<td>820</td>
</tr>
<tr>
<td>OrgSettings</td>
<td>823</td>
</tr>
<tr>
<td>PathAssistantSettings</td>
<td>824</td>
</tr>
<tr>
<td>ProductSettings</td>
<td>825</td>
</tr>
<tr>
<td>ProfileSessionSetting</td>
<td>826</td>
</tr>
<tr>
<td>QuoteSettings</td>
<td>827</td>
</tr>
<tr>
<td>SearchSettings</td>
<td>828</td>
</tr>
<tr>
<td>SecuritySettings</td>
<td>831</td>
</tr>
<tr>
<td>SocialCustomerServiceSettings</td>
<td>839</td>
</tr>
<tr>
<td>Territory2Settings</td>
<td>840</td>
</tr>
<tr>
<td>UserManagementSettings</td>
<td>841</td>
</tr>
<tr>
<td>OwnerSharingRule</td>
<td>860</td>
</tr>
<tr>
<td>CriteriaBasedSharingRule</td>
<td>854</td>
</tr>
<tr>
<td>BaseSharingRule</td>
<td>853</td>
</tr>
<tr>
<td>SharedTo</td>
<td>842</td>
</tr>
<tr>
<td>SharingRules</td>
<td>845</td>
</tr>
<tr>
<td>SharingSet</td>
<td>866</td>
</tr>
<tr>
<td>SiteDotCom</td>
<td>870</td>
</tr>
<tr>
<td>Skill</td>
<td>871</td>
</tr>
<tr>
<td>StandardValueSet</td>
<td>873</td>
</tr>
<tr>
<td>StandardValueSetTranslation</td>
<td>875</td>
</tr>
<tr>
<td>StaticResource</td>
<td>876</td>
</tr>
<tr>
<td>SynonymDictionary</td>
<td>877</td>
</tr>
<tr>
<td>Territory</td>
<td>879</td>
</tr>
<tr>
<td>Territory2</td>
<td>880</td>
</tr>
<tr>
<td>Territory2Model</td>
<td>885</td>
</tr>
<tr>
<td>Territory2Rule</td>
<td>887</td>
</tr>
<tr>
<td>Territory2Type</td>
<td>889</td>
</tr>
<tr>
<td>TimeSheetTemplate</td>
<td>890</td>
</tr>
<tr>
<td>TopicsForObjects</td>
<td>893</td>
</tr>
<tr>
<td>TransactionSecurityPolicy</td>
<td>894</td>
</tr>
<tr>
<td>Translations</td>
<td>898</td>
</tr>
<tr>
<td>UserCriteria</td>
<td>910</td>
</tr>
<tr>
<td>UserInterfaceSettings</td>
<td>911</td>
</tr>
</tbody>
</table>
CHAPTER 1  Understanding Metadata API

Use Metadata API to retrieve, deploy, create, update or delete customization information, such as custom object definitions and page layouts, for your organization. This API is intended for managing customizations and for building tools that can manage the metadata model, not the data itself. To create, retrieve, update or delete records, such as accounts or leads, use data SOAP API or REST API.

The easiest way to access the functionality in Metadata API is to use the Force.com IDE or Ant Migration Tool. Both tools are built on top of Metadata API and use the standard Eclipse and Ant tools, respectively, to simplify working with Metadata API.

- Force.com IDE is built on the Eclipse platform, for programmers familiar with integrated development environments. Code, compile, test, and deploy from within the IDE.
- The Ant Migration Tool is ideal if you use a script or the command line for moving metadata between a local directory and a Salesforce org.

For more information about the Force.com IDE or Ant Migration Tool, see developer.salesforce.com.

The underlying calls of Metadata API have been exposed for you to use directly, if you prefer to build your own client applications. This guide gives you more information about working directly with Metadata API.

You can use the Metadata API to manage setup and customization information (metadata) for your organizations. For example:

- Export the customizations in your organization as XML metadata files. See Working with the Zip File and retrieve().
- Migrate configuration changes between organizations. See deploy() and retrieve().
- Modify existing customizations in your organization using XML metadata files. See deploy() and retrieve().
- Manage customizations in your organization programmatically. See CRUD-Based Metadata Development.

You can modify metadata in test organizations on Developer Edition or sandbox, and then deploy tested changes to production organizations on Enterprise, Unlimited, or Performance Editions. You can also create scripts to populate a new organization with your custom objects, custom fields, and other components.

SEE ALSO:
- Deploying and Retrieving Metadata
- CRUD-Based Metadata Development
- Metadata Components and Types

Supported Salesforce Editions


It is strongly recommended that you use a sandbox, which is an exact replica of your production organization. Enterprise, Unlimited, and Performance Editions come with free developer sandboxes. For more information, see http://www.salesforce.com/platform/cloud-infrastructure/sandbox.jsp.
Alternatively, you can use a Developer Edition org, which provides access to all of the features available with Enterprise Edition, but is limited by the number of users and the amount of storage space. A Developer Edition org isn’t a copy of your production org, but it provides an environment where you can build and test your solutions without affecting your organization’s data. Developer Edition accounts are available for free at http://developer.salesforce.com/signup.

Note: A metadata component must be visible in the org for Metadata API to act on it. Also, a user must have the API Enabled permission to have access to metadata components.

Metadata API Access for Professional Edition

ISV partners can request Metadata API access to Professional Edition orgs for apps that have passed the AppExchange Security Review. Access is granted through an API token (client ID). This special key enables the app to make Metadata API calls to customers’ Professional Edition orgs.

As an ISV partner, you can request Metadata API access by following these steps.

2. After your app is approved, log a case in the Partner Community in AppExchange and Feature Requests > API Token Request, and specify SOAP for the type of token.

To make calls to the Metadata API, append the API token to the CallOptions SOAP header in your calls.

Modify Metadata Through Metadata API Functions Permission

Users with the Modify Metadata Through Metadata API Functions permission can update metadata (including Apex) through Metadata API even if they don’t also have the Modify All Data permission. Metadata API is used for deployments using change sets, the Ant Migration Tool, or the Salesforce CLI. Users must have the permission that enables use of the feature supported by the metadata they’re trying to modify. They must also have the permission that enables their deployment tool.

The Modify Metadata Through Metadata API Functions permission doesn’t impact direct customization of metadata using Setup UI pages, because those pages don’t use Metadata API for updates.

Some metadata, such as Apex, executes in system context, so be careful how you delegate the Modify Metadata Through Metadata API Functions permission. Modify Metadata Through Metadata API Functions allows deployment of Apex metadata, but it doesn’t allow some Apex development and debugging features that still require the Modify All Data permission.

Modify Metadata Through Metadata API Functions is enabled automatically when either the Deploy Change Sets or the Author Apex permission is selected.

Development Platforms

Metadata API supports both file-based and CRUD-based development.

File-Based Development

The declarative or file-based asynchronous Metadata API deploy() and retrieve() operations deploy or retrieve a .zip file that holds components in a set of folders, and a manifest file named package.xml. For more information, see Deploying and Retrieving Metadata on page 16. The easiest way to access the file-based functionality is to use the Force.com IDE or Ant Migration Tool.
CRUD-Based Development

The CRUD Metadata API calls act upon the metadata components in a manner similar to the way synchronous API calls in the enterprise WSDL act upon objects. For more information about the enterprise WSDL, see the SOAP API Developer Guide.

Standards Compliance

Metadata API is implemented to comply with the following specifications:

<table>
<thead>
<tr>
<th>Standard Name</th>
<th>Website</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Description Language (WSDL) 1.1</td>
<td><a href="http://www.w3.org/TR/2001/NOTE-wsdl-20010315">http://www.w3.org/TR/2001/NOTE-wsdl-20010315</a></td>
</tr>
<tr>
<td>WS-I Basic Profile 1.1</td>
<td><a href="http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html">http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html</a></td>
</tr>
</tbody>
</table>

Metadata API Support Policy

Salesforce supports previous versions of Metadata API. However, your new client applications should use the most recent version of the Lightning Platform Metadata API WSDL file to fully exploit the benefits of richer features and greater efficiency.

Backward Compatibility

Salesforce strives to make backward compatibility easy when using the Lightning Platform.

Each new Salesforce release consists of two components:

- A new release of platform software that resides on Salesforce systems
- A new version of the API

For example, the Spring '07 release included API version 9.0 and the Summer '07 release included API version 10.0.

We maintain support for each API version across releases of the platform software. The API is backward compatible in that an application created to work with a given API version will continue to work with that same API version in future platform software releases.

Salesforce does not guarantee that an application written against one API version will work with future API versions: Changes in method signatures and data representations are often required as we continue to enhance the API. However, we strive to keep the API consistent from version to version with minimal, if any, changes required to port applications to newer API versions.

For example, an application written using API version 9.0, which shipped with the Spring '07 release, will continue to work with API version 9.0 on the Summer '07 release, and on future releases beyond that. However, that same application might not work with API version 10.0 without modifications to the application.

API End-of-Life

Salesforce is committed to supporting each API version for a minimum of three years from the date of first release. In order to mature and improve the quality and performance of the API, versions that are more than three years old might cease to be supported.

When an API version is to be deprecated, advance notice is given at least one year before support ends. Salesforce will directly notify customers using API versions planned for deprecation.
Related Resources

The Salesforce developer website provides a full suite of developer toolkits, sample code, sample SOAP messages, community-based support, and other resources to help you with your development projects. Be sure to visit https://developer.salesforce.com/page/Getting Started for more information, or visit http://developer.salesforce.com/signup to sign up for a free Developer Edition account.

You can visit these websites to find out more about Salesforce applications:

- Salesforce Developers provides a wealth of information for developers.
- Salesforce for information about the Salesforce application.
- Lightning Platform AppExchange for access to apps created for Salesforce.
- Salesforce.com Community for services to ensure Salesforce customer success.
CHAPTER 2  Quick Start

Use Metadata API to retrieve, deploy, create, update, or delete customizations for your org. The most common use is to migrate changes from a sandbox or testing org to your production environment. Metadata API is intended for managing customizations and for building tools that can manage the metadata model, not the data itself.

However, the underlying calls of Metadata API have been exposed for you to use directly, if you prefer to build your own client applications. This quick start gives you all the information you need to start writing applications that directly use Metadata API to manage customizations for your organization. It shows you how to get started with File-Based Development. For an example of CRUD-Based Development, see Java Sample for CRUD-Based Development with Synchronous Calls.

Prerequisites

Make sure you complete these prerequisites before you start using Metadata API.

- Create a development environment.

  It is strongly recommended that you use a sandbox, which is an exact replica of your production organization. Enterprise, Unlimited, and Performance Editions come with free developer sandboxes. For more information, see http://www.salesforce.com/platform/cloud-infrastructure/sandbox.jsp.

  Alternatively, you can use a Developer Edition org, which provides access to all of the features available with Enterprise Edition, but is limited by the number of users and the amount of storage space. A Developer Edition org isn’t a copy of your production org, but it provides an environment where you can build and test your solutions without affecting your organization’s data. Developer Edition accounts are available for free at http://developer.salesforce.com/signup.

- Identify a user that has the API Enabled permission and the Modify All Data or Modify Metadata Through Metadata API Functions permissions. These permissions are required to access Metadata API calls.

  ![Note:](Note: If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

- Install a SOAP client. Metadata API works with current SOAP development environments, including, but not limited to, Visual Studio®.NET and the Web Service Connector (WSC).

  In this document, we provide Java examples based on WSC and JDK 6 (Java Platform Standard Edition Development Kit 6). To run the samples, first download the latest force-wsc JAR file and its dependencies (dependencies are listed on the page when you select a version) from mvnrepository.com/artifact/com.force.api/force-wsc/.

  ![Note:](Note: Development platforms vary in their SOAP implementations. Implementation differences in certain development platforms might prevent access to some or all of the features in Metadata API.
Step 1: Generate or Obtain the Web Service WSDLs for Your Organization

To access Metadata API calls, you need a Web Service Description Language (WSDL) file. The WSDL file defines the Web service that is available to you. Your development platform uses this WSDL to generate stub code to access the Web service it defines. You can either obtain the WSDL file from your organization’s Salesforce administrator, or you can generate it yourself if you have access to the WSDL download page in the Salesforce user interface. For more information about WSDL, see http://www.w3.org/TR/wsdl.

Before you can access Metadata API calls, you must authenticate to use the Web service using the `login()` call, which is defined in the enterprise WSDL and the partner WSDL. Therefore, you must also obtain one of these WSDLs.

Any user with the Modify All Data or Modify Metadata Through Metadata API Functions permission can download the WSDL file to integrate and extend the Salesforce platform.

**Note:** If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

The sample code in Step 3: Walk Through the Java Sample Code on page 7 uses the enterprise WSDL, though the partner WSDL works equally well.

To generate the metadata and enterprise WSDL files for your organization:

1. Log in to your Salesforce account. You must log in as an administrator or as a user who has the “Modify All Data” permission.
2. From Setup, enter API in the Quick Find box, then select API.
3. Click Generate Metadata WSDL and save the XML WSDL file to your file system.
4. Click Generate Enterprise WSDL and save the XML WSDL file to your file system.

Step 2: Import the WSDL Files Into Your Development Platform

Once you have the WSDL files, import them into your development platform so that your development environment can generate the necessary objects for use in building client Web service applications. This section provides sample instructions for WSC. For instructions about other development platforms, see your platform’s product documentation.

**Note:** The process for importing WSDL files is identical for the metadata and enterprise WSDL files.

Instructions for Java Environments (WSC)

Java environments access the API through Java objects that serve as proxies for their server-side counterparts. Before using the API, you must first generate these objects from your organization’s WSDL file.

Each SOAP client has its own tool for this process. For WSC, use the `wsdlc` utility.

**Note:** Before you run `wsdlc`, you must have the WSC JAR file installed on your system and referenced in your classpath. You can download the latest force-wsc JAR file and its dependencies (dependencies are listed on the page when you select a version) from mvnrepository.com/artifact/com.force.api/force-wsc/.

The basic syntax for `wsdlc` is:

```
java -classpath pathToWsc;pathToWscDependencies com.sforce.ws.tools.wsdlc pathToWsdl/WsdlFilename pathToOutputJar/OutputJarFilename
```
For example, on Windows:

```
java -classpath force-wsc-30.0.0.jar;ST4-4.0.7.jar;antlr-runtime-3.5.jar
   com.sforce.ws.tools.wsdlc metadata.wsdl metadata.jar
```

On Mac OS X and Unix, use a colon instead of a semicolon in between items in the classpath:

```
java -classpath force-wsc-30.0.0.jar:ST4-4.0.7.jar:antlr-runtime-3.5.jar
   com.sforce.ws.tools.wsdlc metadata.wsdl metadata.jar
```

wsdlc generates a JAR file and Java source code and bytecode files for use in creating client applications. Repeat this process for the enterprise WSDL to create an enterprise JAR file.

### Step 3: Walk Through the Java Sample Code

When you have imported the WSDL files, you can build client applications that use Metadata API. The sample is a good starting point for writing your own code.

Before you run the sample, modify your project and the code to:

1. Include the WSC JAR, its dependencies, and the JAR files you generated from the WSDLs.

   **Note:** Although WSC has other dependencies, the following sample only requires Rhino (js-1.7R2.jar), which you can download from mvnrepository.com/artifact/rhino/js.

2. Update USERNAME and PASSWORD variables in the MetadataLoginUtil.login() method with your user name and password. If your current IP address isn’t in your organization’s trusted IP range, you’ll need to append a security token to the password.

3. If you are using a sandbox, be sure to change the login URL.

### Login Utility

Java users can use ConnectorConfig to connect to Enterprise, Partner, and Metadata SOAP API. MetadataLoginUtil creates a ConnectorConfig object and logs in using the Enterprise WSDL login method. Then it retrieves sessionId and metadataServerUrl to create a ConnectorConfig and connects to Metadata API endpoint. ConnectorConfig is defined in WSC.

The MetadataLoginUtil class abstracts the login code from the other parts of the sample, allowing portions of this code to be reused without change across different Salesforce APIs.

```java
import com.sforce.soap.enterprise.EnterpriseConnection;
import com.sforce.soap.enterprise.LoginResult;
import com.sforce.soap.metadata.MetadataConnection;
import com.sforce.ws.ConnectionException;
import com.sforce.ws.ConnectorConfig;

/**
 * Login utility.
 */
public class MetadataLoginUtil {

    public static MetadataConnection login() throws ConnectionException {
        final String USERNAME = "user@company.com";
        // This is only a sample. Hard coding passwords in source files is a bad practice.
```
final String PASSWORD = "password";
final String URL = "https://login.salesforce.com/services/Soap/c/46.0";
final LoginResult loginResult = loginToSalesforce(USERNAME, PASSWORD, URL);
return createMetadataConnection(loginResult);
}

private static MetadataConnection createMetadataConnection(
    final LoginResult loginResult) throws ConnectionException {
    final ConnectorConfig config = new ConnectorConfig();
    config.setServiceEndpoint(loginResult.getMetadataServerUrl());
    config.setSessionId(loginResult.getSessionId());
    return new MetadataConnection(config);
}

private static LoginResult loginToSalesforce(
    final String username,
    final String password,
    final String loginUrl) throws ConnectionException {
    final ConnectorConfig config = new ConnectorConfig();
    config.setAuthEndpoint(loginUrl);
    config.setServiceEndpoint(loginUrl);
    config.setManualLogin(true);
    return (new EnterpriseConnection(config)).login(username, password);
}

Note: This example uses user and password authentication to obtain a session ID, which is then used for making calls to Metadata API. Alternatively, you can use OAuth authentication. After you authenticate with OAuth to Salesforce, pass the returned access token instead of the session ID. For example, pass the access token to the `setSessionId()` call on `ConnectorConfig`. To learn how to use OAuth authentication in Salesforce, see Authenticating Apps with OAuth in the Salesforce Help.

Java Sample Code for File-Based Development

The sample code logs in using the `login` utility. Then it displays a menu with retrieve, deploy, and exit.

The `retrieve()` and `deploy()` calls both operate on a .zip file named `components.zip`. The `retrieve()` call retrieves components from your organization into `components.zip`, and the `deploy()` call deploys the components in `components.zip` to your organization. If you save the sample to your computer and execute it, run the retrieve option first so that you have a `components.zip` file that you can subsequently deploy. After a retrieve call, the sample calls `checkRetrieveStatus()` in a loop until the operation is completed. Similarly, after a deploy call, the sample checks `checkDeployStatus()` in a loop until the operation is completed.

The `retrieve()` call uses a manifest file to determine the components to retrieve from your organization. A sample `package.xml` manifest file follows. For more details on the manifest file structure, see Deploying and Retrieving Metadata with the Zip File. For this sample, the manifest file retrieves all custom objects, custom tabs, and page layouts.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>/*</members>
        <name>CustomObject</name>
    </types>
    <types>
        <members>/*</members>
```
<name>CustomTab</name>
</types>
<types>
  <members>*</members>
  <name>Layout</name>
</types>
<version>46.0</version>
</Package>

Note the error handling code that follows each API call.

Note: This sample requires API version 34.0 or later.

```java
import java.io.*;
import java.nio.channels.Channels;
import java.nio.channels.FileChannel;
import java.nio.channels.ReadableByteChannel;
import java.rmi.RemoteException;
import java.util.*;
import javax.xml.parsers.*;
import org.w3c.dom.*;
import org.xml.sax.SAXException;
import com.sforce.soap.metadata.*;

/**
 * Sample that logs in and shows a menu of retrieve and deploy metadata options.
 */
public class FileBasedDeployAndRetrieve {

  private MetadataConnection metadataConnection;

  private static final String ZIP_FILE = "components.zip";

  // manifest file that controls which components get retrieved
  private static final String MANIFEST_FILE = "package.xml";

  private static final double API_VERSION = 29.0;

  // one second in milliseconds
  private static final long ONE_SECOND = 1000;

  // maximum number of attempts to deploy the zip file
  private static final int MAX_NUM_POLL_REQUESTS = 50;

  private BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));

  public static void main(String[] args) throws Exception {
    FileBasedDeployAndRetrieve sample = new FileBasedDeployAndRetrieve();
    sample.run();
  }
```
public FileBasedDeployAndRetrieve() {
}

private void run() throws Exception {
    this.metadataConnection = MetadataLoginUtil.login();

    // Show the options to retrieve or deploy until user exits
    String choice = getUsersChoice();
    while (choice != null && !choice.equals("99")) {
        if (choice.equals("1")) {
            retrieveZip();
        } else if (choice.equals("2")) {
            deployZip();
        } else {
            break;
        }
        // show the options again
        choice = getUsersChoice();
    }
}

/*
 * Utility method to present options to retrieve or deploy.
 */
private String getUsersChoice() throws IOException {
    System.out.println(" 1: Retrieve");
    System.out.println(" 2: Deploy");
    System.out.println("99: Exit");
    System.out.println();
    System.out.print("Enter 1 to retrieve, 2 to deploy, or 99 to exit: ");
    // wait for the user input.
    String choice = reader.readLine();
    return choice != null ? choice.trim() : "";
}

private void deployZip() throws Exception {
    byte[] zipBytes = readZipFile();
    DeployOptions deployOptions = new DeployOptions();
    deployOptions.setPerformRetrieve(false);
    deployOptions.setRollbackOnError(true);
    AsyncResult asyncResult = metadataConnection.deploy(zipBytes, deployOptions);
    DeployResult result = waitForDeployCompletion(asyncResult.getId());
    if (!result.isSuccess()) {
        printErrors(result, "Final list of failures:
        throw new Exception("The files were not successfully deployed");
    }
    System.out.println("The file " + ZIP_FILE + " was successfully deployed\n");
}

/*
 * Read the zip file contents into a byte array.
 */
private byte[] readZipFile() throws Exception {
    byte[] result = null;
// We assume here that you have a deploy.zip file.  
// See the retrieve sample for how to retrieve a zip file. 
File zipFile = new File(ZIP_FILE); 
if (zipFile.exists() || zipFile.isFile()) {
    throw new Exception("Cannot find the zip file for deploy() on path:"  
        + zipFile.getAbsolutePath());
}

FileInputStream fileInputStream = new FileInputStream(zipFile);  
try {
    ByteArrayOutputStream bos = new ByteArrayOutputStream();  
    byte[] buffer = new byte[4096];  
    int bytesRead = 0;  
    while (-1 != (bytesRead = fileInputStream.read(buffer))) {  
        bos.write(buffer, 0, bytesRead);
    }
    result = bos.toByteArray();
} finally {
    fileInputStream.close();
} 
return result;

/* 
 * Print out any errors, if any, related to the deploy.  
 * @param result - DeployResult 
 */
private void printErrors(DeployResult result, String messageHeader) {
    DeployDetails details = result.getDetails(); 
    StringBuilder stringBuilder = new StringBuilder(); 
    if (details != null) {
        DeployMessage[] componentFailures = details.getComponentFailures();
        for (DeployMessage failure : componentFailures) {
            String loc = "(" + failure.getLineNumber() + ", " + failure.getColumnNumber();
            if (loc.length() == 0 && !failure.getFileName().equals(failure.getFullName()))
                loc = "(" + failure.getFullName() + ");
            stringBuilder.append(failure.getFileName() + loc + ":"  
                + failure.getProblem()).append('
');
        }
        RunTestsResult rtr = details.getRunTestResult();
        if (rtr.getFailures() != null) {
            for (RunTestFailure failure : rtr.getFailures()) {
                String n = (failure.getNamespace() == null ? ""  
                    : (failure.getNamespace() + ".") + failure.getName());
                stringBuilder.append("Test failure, method: " + n + ": "  
                    + failure.getMethodName() + " -- " + failure.getMessage() +  
                    + " stack " + failure.getStackTrace() + "\n\n");
            }
        }
    }
}  

11
if (rtr.getCodeCoverageWarnings() != null) {
   for (CodeCoverageWarning ccw : rtr.getCodeCoverageWarnings()) {
      stringBuilder.append("Code coverage issue");
      if (ccw.getName() != null) {
         String n = (ccw.getNamespace() == null ? "" :
                     (ccw.getNamespace() + ".") + ccw.getName);
         stringBuilder.append(n);
      }
      stringBuilder.append(" -- " + ccw.getMessage() + "\n");
   }
}

if (stringBuilder.length() > 0) {
   stringBuilder.insert(0, messageHeader);
   System.out.println(stringBuilder.toString());
}

private void retrieveZip() throws Exception {
RetrieveRequest retrieveRequest = new RetrieveRequest();
// The version in package.xml overrides the version in RetrieveRequest
retrieveRequest.setApiVersion(API_VERSION);
setUnpackaged(retrieveRequest);

AsyncResult asyncResult = metadataConnection.retrieve(retrieveRequest);
RetrieveResult result = waitForRetrieveCompletion(asyncResult);

if (result.getStatus() == RetrieveStatus.Failed) {
   throw new Exception(result.getErrorStatusCode() + " msg: " +
                        result.getErrorMessage());
} else if (result.getStatus() == RetrieveStatus.Succeeded) {
   // Print out any warning messages
   StringBuilder stringBuilder = new StringBuilder();  
   if (result.getMessages() != null) {
      for (RetrieveMessage rm : result.getMessages()) {
         stringBuilder.append(rm.getFileName() + " - " + rm.getProblem() + "\n");
      }
   }
   if (stringBuilder.length() > 0) {
      System.out.println("Retrieve warnings:\n" + stringBuilder);
   }
   System.out.println("Writing results to zip file");
   File resultsFile = new File(ZIP_FILE);
   FileOutputStream os = new FileOutputStream(resultsFile);
   try {
      os.write(result.getZipFile());
   } finally {
      os.close();
   }
}
private DeployResult waitForDeployCompletion(String asyncResultId) throws Exception {
    int poll = 0;
    long waitTimeMilliSecs = ONE_SECOND;
    DeployResult deployResult;
    boolean fetchDetails;
    do {
        Thread.sleep(waitTimeMilliSecs);
        // double the wait time for the next iteration
        waitTimeMilliSecs *= 2;
        if (poll++ > MAX_NUM_POLL_REQUESTS) {
            throw new Exception("Request timed out. If this is a large set of metadata components, "
                       + "ensure that MAX_NUM_POLL_REQUESTS is sufficient.");
        }
        // Fetch in-progress details once for every 3 polls
        fetchDetails = (poll % 3 == 0);
        deployResult = metadataConnection.checkDeployStatus(asyncResultId, fetchDetails);
        System.out.println("Status is: " + deployResult.getStatus());
        if (!deployResult.isDone() && fetchDetails) {
            printErrors(deployResult, "Failures for deployment in progress:
        }
    } while (!deployResult.isDone());
    if (!deployResult.isSuccess() && deployResult.getErrorStatusCode() != null) {
        throw new Exception(deployResult.getErrorStatusCode() + " msg: " +
                            deployResult.getErrorMessage());
    }
    if (!fetchDetails) {
        // Get the final result with details if we didn't do it in the last attempt.
        deployResult = metadataConnection.checkDeployStatus(asyncResultId, true);
    }
    return deployResult;
}

private RetrieveResult waitForRetrieveCompletion(AsyncResult asyncResult) throws Exception {
    // Wait for the retrieve to complete
    int poll = 0;
    long waitTimeMilliSecs = ONE_SECOND;
    String asyncResultId = asyncResult.getId();
    RetrieveResult result = null;
    do {
        Thread.sleep(waitTimeMilliSecs);
        // Double the wait time for the next iteration
        waitTimeMilliSecs *= 2;
        if (poll++ > MAX_NUM_POLL_REQUESTS) {
            throw new Exception("Request timed out. If this is a large set of metadata components, "
                    + "ensure that MAX_NUM_POLL_REQUESTS is sufficient.");
        }
        // Fetch in-progress details once for every 3 polls
        result = metadataConnection.checkRetrieveStatus(asyncResultId, poll % 3 == 0);
        System.out.println("Status is: " + result.getStatus());
        if (!result.isSuccess() && result.getErrorStatusCode() != null) {
            throw new Exception(result.getErrorStatusCode() + " msg: " +
                                result.getErrorMessage());
        }
    } while (!result.isSuccess());
    return result;
}
waitTimeMilliSecs *= 2;
if (poll++ > MAX_NUM_POLL_REQUESTS) {
    throw new Exception("Request timed out. If this is a large set " +
    "of metadata components, check that the time allowed " +
    "by MAX_NUM_POLL_REQUESTS is sufficient.");
}
result = metadataConnection.checkRetrieveStatus(
    asyncResultId, true);
System.out.println("Retrieve Status: " + result.getStatus());
while (!result.isDone());
return result;
}

private void setUnpackaged(RetrieveRequest request) throws Exception {
    // Edit the path, if necessary, if your package.xml file is located elsewhere
    File unpackedManifest = new File(MANIFEST_FILE);
    System.out.println("Manifest file: " + unpackedManifest.getAbsolutePath());
    if (!unpackedManifest.exists() || !unpackedManifest.isFile()) {
        throw new Exception("Should provide a valid retrieve manifest " +
            "for unpackaged content. Looking for " +
            unpackedManifest.getAbsolutePath());
    }
    // Note that we use the fully qualified class name because
    // of a collision with the java.lang.Package class
    com.sforce.soap.metadata.Package p = parsePackageManifest(unpackedManifest);
    request.setUnpackaged(p);
}

private com.sforce.soap.metadata.Package parsePackageManifest(File file)
    throws ParserConfigurationException, IOException, SAXException {
    com.sforce.soap.metadata.Package packageManifest = null;
    List<PackageTypeMembers> listPackageTypes = new ArrayList<PackageTypeMembers>();
    DocumentBuilder db = DocumentBuilderFactory.newInstance().newDocumentBuilder();
    InputStream inputStream = new FileInputStream(file);
    Element d = db.parse(inputStream).getDocumentElement();
    for (Node c = d.getFirstChild(); c != null; c = c.getNextSibling()) {
        if (c instanceof Element) {
            Element ce = (Element) c;
            NodeList nodeList = ce.getElementsByTagName("name");
            if (nodeList.getLength() == 0) {
                continue;
            }
            String name = nodeList.item(0).getTextContent();
            NodeList m = ce.getElementsByTagName("members");
            List<String> members = new ArrayList<String>();
            for (int i = 0; i < m.getLength(); i++) {
                Node mm = m.item(i);
                members.add(mm.getTextContent());
            }
            PackageTypeMembers packageTypes = new PackageTypeMembers();
        }
packageTypes.setName(name);
packageTypes.setMembers(members.toArray(new String[members.size()]));
listPackageTypes.add(packageTypes);
}
packageManifest = new com.sforce.soap.metadata.Package();
PackageTypeMembers[] packageTypesArray =
    new PackageTypeMembers[listPackageTypes.size()];
packageManifest.setTypes(listPackageTypes.toArray(packageTypesArray));
packageManifest.setVersion(API_VERSION + "");
return packageManifest;
}
CHAPTER 3  Deploying and Retrieving Metadata

Use the `deploy()` and `retrieve()` calls to move metadata (XML files) between a Salesforce organization and a local file system. Once you retrieve your XML files into a file system, you can manage changes in a source-code control system, copy and paste code or setup configurations, diff changes to components, and perform many other file-based development operations. At any time you can deploy those changes to another Salesforce organization.

**Note:** The Force.com IDE and the Ant Migration Tool use the `deploy()` and `retrieve()` calls to move metadata. If you use these tools, interaction with Metadata API is seamless and invisible. Therefore, most developers will find it much easier to use these tools than write code that calls `deploy()` and `retrieve()` directly.

Data in XML files is formatted using the English (United States) locale. This ensures that fields that depend on locale, such as date fields, are interpreted consistently during data migrations between organizations using different languages. Organizations can support multiple languages for presentation to their users.

The `deploy()` and `retrieve()` calls are used primarily for the following development scenarios:

- Development of a custom application (or customization) in a sandbox organization. After development and testing is completed, the application or customization is then deployed into a production organization using Metadata API.
- Team development of an application in a Developer Edition organization. After development and testing is completed, you can then distribute the application via Lightning Platform AppExchange.

**SEE ALSO:**
- Metadata Components and Types
- Unsupported Metadata Types

Deploying and Retrieving Metadata with the Zip File

The `deploy()` and `retrieve()` calls are used to deploy and retrieve a `.zip` file. Within the `.zip` file is a project manifest (`package.xml`) that lists what to retrieve or deploy, and one or more XML components that are organized into folders.

**Note:** A component is an instance of a metadata type. For example, `CustomObject` is a metadata type for custom objects, and the `MyCustomObject__c` component is an instance of a custom object.

The files that are retrieved or deployed in a `.zip` file might be unpackaged components that reside in your organization (such as `standard objects`) or packaged components that reside within named packages.

**Note:** You can deploy or retrieve up to 10,000 files at once and the maximum size of the deployed or retrieved `.zip` file is 39 MB. (In API version 43.0 and later, AppExchange packages can contain up to 12,500 files.) Note the following:

- If using the Ant Migration Tool to deploy an unzipped folder, all files in the folder are compressed first. The maximum size of uncompressed components in an unzipped folder is 400 MB or less depending on the compression ratio. If the files have a high compression ratio, you can migrate a total of approximately 400 MB because the compressed size would be under 39 MB. However, if the components can’t be compressed much, like binary static resources, you can migrate less than 400 MB.
• Metadata API base-64 encodes components after they’re compressed. The resulting .zip file can’t exceed 50 MB, which is the limit for SOAP messages. Base-64 encoding increases the size of the payload, so your compressed payload can’t exceed approximately 39 MB before encoding.

• You can perform a retrieve() call for a big object only if its index is defined. If a big object is created in Setup and doesn’t yet have an index defined, you can’t retrieve it.

Every .zip file contains a project manifest, a file that’s named package.xml, and a set of directories that contain the components. The manifest file defines the components that you’re trying to retrieve or deploy in the .zip file and the API version that’s used for the deployment or retrieval.

The following is a sample package.xml file. Note that you can retrieve an individual component for a metadata type by specifying its fullName field value in a members element, or you can also retrieve all components of a metadata type by using <members>*</members>.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>MyCustomObject__c</members>
        <name>CustomObject</name>
    </types>
    <types>
        <members>*</members>
        <name>CustomTab</name>
    </types>
    <types>
        <members>Standard</members>
        <name>Profile</name>
    </types>
    <version>46.0</version>
</Package>
```

The following elements can be defined in package.xml.

• <full_name> contains the name of the server-side package. If no <full_name> exists, this is a client-side unpackaged package.

• <types> contains the name of the metadata type (for example, CustomObject) and the named members (for example, myCustomObject__c) to be retrieved or deployed. You can add multiple <types> elements in a manifest file.

• <members> contains the fullName of the component, for example MyCustomObject__c. The listMetadata() call is useful for determining the fullName for components of a particular metadata type, if you want to retrieve an individual component. For many metadata types, you can replace the value in members with the wildcard character * (asterisk) instead of listing each member separately. See the reference topic for a specific type to determine whether that type supports wildcards.

  Note: You specify Security in the <members> element and Settings in the name element when retrieving the SecuritySettings component type.

• <name> contains the metadata type, for example CustomObject or Profile. There is one name defined for each metadata type in the directory. Any metadata type that extends Metadata is a valid value. The name that’s entered must match a metadata type that’s defined in the Metadata API WSDL. See Metadata Types for a list.

• <version> is the API version number that’s used when the .zip file is deployed or retrieved. Currently the valid value is 46.0.

For more sample package.xml manifest files that show you how to work with different subsets of metadata, see Sample package.xml Manifest Files.
To delete components, see Deleting Components from an Organization.

SEE ALSO:
- Metadata Types

Sample package.xml Manifest Files

This section includes sample package.xml manifest files that show you how to work with different subsets of metadata. A manifest file can include multiple <types> elements so you could combine the individual samples into one package.xml manifest file if you want to work with all the metadata in one batch. For more information about the structure of a manifest file, see Deploying and Retrieving Metadata with the Zip File. The following samples are listed:

- Standard Objects
- All Custom Objects
- Standard Picklist Fields
- Custom and Standard Fields
- List Views for Standard Objects
- Packages
- Security Settings
- Assignment Rules, Auto-response Rules, Escalation Rules
- Sharing Rules
- Managed Component Access

Standard Objects

This sample package.xml manifest file illustrates how to work with the standard Account object. Retrieving or deploying a standard object includes all custom and standard fields except for standard fields that aren't customizable. All custom fields are supported. Only standard fields that you can customize are supported, that is, standard fields to which you can add help text or enable history tracking or Chatter feed tracking. Other standard fields aren't supported, including system fields (such as CreatedById or LastModified) and autonumber fields.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Account</members>
    <name>CustomObject</name>
  </types>
  <version>46.0</version>
</Package>
```

Note how you work with the standard Account object by specifying it as a member of a CustomObject type. However, you cannot use an asterisk wildcard to work with all standard objects; each standard object must be specified by name.
All Custom Objects

This sample package.xml manifest file illustrates how to work with all custom objects.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>CustomObject</name>
  </types>
  <version>46.0</version>
</Package>
```

This manifest file can be used to retrieve or deploy all custom objects. This does not include all standard objects.

Standard Picklist Fields

In API version 38.0 and later, the StandardValueSet type represents standard picklists. Picklists are no longer represented by fields as in earlier versions. This sample package.xml represents the Industry standard picklist as a StandardValueSet type.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Industry</members>
    <name>StandardValueSet</name>
  </types>
  <version>46.0</version>
</Package>
```

Note: The name of a standard value set is case-sensitive.

The Industry standard value set corresponds to the Account.Industry or Lead.Industry field in API version 37.0 and earlier. This example shows a package.xml sample for the Account.Industry picklist.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Account.Industry</members>
    <name>CustomField</name>
  </types>
  <version>37.0</version>
</Package>
```

Note: The name of a picklist field is case-sensitive.

Note the `objectName.picklistField` syntax in the `<members>` field where `objectName` is the name of the object, such as Account, and `picklistField` is the name of the standard picklist field, such as Industry.

This next package.xml sample represents opportunity team roles in API version 38.0 and later. Specify opportunity team roles as a SalesTeamRole standard value set. Opportunity team roles have the same picklist values as the account team roles.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>SalesTeamRole</members>
  </types>
</Package>
```
The SalesTeamRole standard value set corresponds to one of these field names in API version 37.0 and earlier:
OpportunityTeamMember.TeamMemberRole, UserAccountTeamMember.TeamMemberRole, 
UserTeamMember.TeamMemberRole, and AccountTeamMember.TeamMemberRole. Opportunity team roles are represented in this sample package.xml as the OpportunityTeamMember.TeamMemberRole field.

To learn about the names of standard value sets and how they map to picklist field names, see StandardValueSet Names and Standard Picklist Fields.

Custom and Standard Fields

This sample package.xml manifest file illustrates how to work with custom fields in custom and standard objects and standard fields in a standard object.

Note the objectName.field syntax in the <members> field where objectName is the name of the object, such as Account, and field is the name of the custom or standard field, such as an SLA picklist field representing a service-level agreement option. The MyCustomField custom field in the MyCustomObject custom object is uniquely identified by its full name, 
MyCustomObject__c.MyCustomField__c. Similarly, the Phone standard field in the Account standard object is uniquely identified by its full name, Account.Phone.

All custom fields are supported. Only standard fields that you can customize are supported, that is, standard fields to which you can add help text or enable history tracking or Chatter feed tracking. Other standard fields aren’t supported, including system fields (such as CreatedById or LastModifiedById) and autonumber fields.
List Views for Standard Objects

The easiest way to retrieve list views for a standard object is to retrieve the object. The list views are included in the retrieved component. See Standard Objects on page 18.

You can also work with individual list views if you do not want to retrieve all the details for the object. This sample package.xml manifest file illustrates how to work with a list view for the standard Account object.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Account.AccountTeam</members>
    <name>ListView</name>
  </types>
  <version>46.0</version>
</Package>
```

Note the `objectName.listViewUniqueName` syntax in the `<members>` field where `objectName` is the name of the object, such as Account, and `listViewUniqueName` is the View Unique Name for the list view. If you retrieve this list view, the component is stored in `objects/Account.object`.

Packages

To retrieve a package, set the name of the package in the `packageNames` field in `RetrieveRequest` when you call `retrieve()`. The package.xml manifest file is automatically populated in the retrieved `.zip` file. The `<fullName>` element in package.xml contains the name of the retrieved package.

If you use an asterisk wildcard in a `<members>` element to retrieve all the components of a particular metadata type, the retrieved contents do not include components in managed packages. For more information about managed packages, see the ISVforce Guide.

The easiest way to retrieve a component in a managed package is to retrieve the complete package by setting the name of the package in the `packageNames` field in `RetrieveRequest`, as described above. The following sample package.xml manifest file illustrates an alternative to retrieve an individual component in a package.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>myns__MyCustomObject__c</members>
    <name>CustomObject</name>
  </types>
  <version>46.0</version>
</Package>
```

Note the `namespacePrefix__objectName` syntax in the `<members>` field where `namespacePrefix` is the namespace prefix of the package and `objectName` is the name of the object. A namespace prefix is a 1 to 15-character alphanumeric identifier that distinguishes your package and its contents from other publishers' packages. For more information, see “Register a Namespace Prefix” in the Salesforce Help.
Security Settings

This sample package.xml manifest file illustrates how to work with an organization’s security settings. You specify Security in the <members> element and Settings in the name element when retrieving the SecuritySettings component type.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Security</members>
    <name>Settings</name>
  </types>
  <version>46.0</version>
</Package>
```

Assignment Rules, Auto-response Rules, Escalation Rules

Assignment rules, auto-response rules and escalation rules use different package.xml type names to access sets of rules or individual rules for object types. For example, the following sample package.xml manifest file illustrates how to access an organization’s assignment rules for just Cases and Leads.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Case</members>
    <members>Lead</members>
    <name>AssignmentRules</name>
  </types>
  <version>46.0</version>
</Package>
```

The following sample package.xml manifest file illustrates how to access just the “samplerule” Case assignment rule and the “newrule” Lead assignment rule. Notice that the type name is AssignmentRule and not AssignmentRules.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Case.samplerule</members>
    <members>Lead.newrule</members>
    <name>AssignmentRule</name>
  </types>
  <version>46.0</version>
</Package>
```


Sharing Rules

In API version 33.0 and later, you can retrieve and deploy sharing rules for all standard and custom objects. This sample package.xml manifest file illustrates how to work with an organization’s sharing rules, which includes retrieving a specific criteria-based sharing rule.
for the lead object, retrieving all ownership-based sharing rules for all objects, and retrieving all territory-based sharing rules for the account object.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Lead.testShareRule</members>
    <name>SharingCriteriaRule</name>
  </types>
  <types>
    <members>*</members>
    <name>SharingOwnerRule</name>
  </types>
  <types>
    <members>Account.*</members>
    <name>SharingTerritoryRule</name>
  </types>
  <version>33.0</version>
</Package>
```

### Managed Component Access

In API version 29.0 and later, you can retrieve and deploy access settings for the following managed components in profiles and permission sets:

- Apex classes
- Apps
- Custom field permissions
- Custom object permissions
- Custom tab settings
- External data sources
- Record types
- Visualforce pages

When retrieving and deploying managed component permissions, specify the namespace followed by two underscores. Wildcards are not supported.

For example, let's say you install a managed package with the namespace `MyNamespace` and the custom object `JobRequest__c`. To set object permissions for `JobRequest__c` in the package to the custom profile `MyProfile`, you would add the following to the `.profile` file.

To deploy:

```xml
<objectPermissions>
  <allowCreate>true</allowCreate>
  <allowDelete>true</allowDelete>
  <allowEdit>true</allowEdit>
  <allowRead>true</allowRead>
  <viewAllRecords>false</viewAllRecords>
  <modifyAllRecords>false</modifyAllRecords>
  <object>MyNamespace__JobRequest__c</object>
</objectPermissions>
```
When retrieving permission sets and profiles, make sure that you also retrieve any components that are related to the permissions and settings. For example, when retrieving app visibilities, you must also retrieve the associated app, and when retrieving object or field permissions, you must also retrieve the associated object.

Running Tests in a Deployment

Default Test Execution in Production

When no test level is specified in the deployment options, the default test execution behavior depends on the contents of your deployment package. When deploying to production, all tests, except those that originate from managed packages, are executed if your deployment package contains Apex classes or triggers. If your package doesn’t contain Apex components, no tests are run by default.

In API version 33.0 and earlier, tests were run for components that required tests, such as custom objects, and not only for Apex components. For example, if your package contains a custom object, all tests are run in API version 33.0 and earlier. In contrast, starting with API version 34.0, no tests are run for this package. The API version corresponds to the version of your API client or the version of the tool you’re using (Ant Migration Tool).

You can run tests for a deployment of non-Apex components. You can override the default test execution behavior by setting the test level in your deployment options. Test levels are enforced regardless of the types of components present in your deployment package.

We recommend that you run all local tests in your development environment, such as sandbox, before deploying to production. Running tests in your development environment reduces the number of tests needed to run in a production deployment.

Default Test Execution in Production for API Version 33.0 and Earlier

For deployment to a production organization, all local tests in your organization are run by default. Tests that originate from installed managed packages aren’t run by default. If any test fails, the entire deployment is rolled back.

If the deployment includes components for the following metadata types, all local tests are run.

- ApexClass
- ApexComponent
- ApexPage
- ApexTrigger
- ArticleType
- BaseSharingRule
- CriteriaBasedSharingRule
- CustomField
- CustomObject
- DataCategoryGroup
For example, no tests are run for the following deployments:

- 1 CustomApplication component
- 100 Report components and 40 Dashboard components

But all local tests are run for any of the following example deployments, because they include at least one component from the list above:

- 1 CustomField component
- 1 ApexComponent component and 1 ApexClass component
- 5 CustomField components and 1 ApexPage component
- 100 Report components, 40 Dashboard components, and 1 CustomField component

SEE ALSO:

deploy()

---

### Running a Subset of Tests in a Deployment

Test levels enable you to have more control over which tests are run in a deployment. To shorten deployment time to production, run a subset of tests when deploying Apex components. The default test execution behavior in production has also changed. By default, if no test level is specified, no tests are executed, unless your deployment package contains Apex classes or triggers.

If the code coverage of an Apex component in the deployment is less than 75%, the deployment fails. If one of the specified tests fails, the deployment also fails. We recommend that you test your deployment in sandbox first to ensure that the specified tests cover each component sufficiently. Even if your organization’s overall code coverage is 75% or more, the individual coverage of the Apex components being deployed can be less. If the code coverage requirement isn’t met, write more tests and include them in the deployment.

To run a subset of tests, set the `RunSpecifiedTests` test level on the `DeployOptions` object. Next, specify each test class to run in `DeployOptions`. Finally, pass `DeployOptions` as an argument to the `deploy()` call. The following example performs those steps to run only the specified test classes.

```java
// Create the DeployOptions object.
DeployOptions deployOptions = new DeployOptions();
```
// Set the appropriate test level.
deployOptions.setTestLevel(TestLevel.RunSpecifiedTests);

// Specify the test classes to run.
// String array contains test class names.
String[] tests = {"TestClass1", "TestClass2", "TestClass3"};
// Add the test class names array to the deployment options.
deployOptions.setRunTests(tests);

// Call deploy() by passing the deployment options object as an argument.
AsyncResult asyncResult = metadatabinding.deploy(zipBytes,deployOptions);

Notes About Running Specific Tests

- You can specify only test classes. You can’t specify individual test methods.
- We recommend that you refactor test classes to include the minimum number of tests that meet code coverage requirements. Refactoring your test classes can contribute to shorter test execution times, and as a result, shorter deployment times.
- You can deactivate a trigger in the target organization by deploying it with an inactive state. However, the trigger must have been previously deployed with an active state.

Run the Same Tests in Sandbox and Production Deployments

Starting in API version 34.0, you can choose which tests to run in your development environment, such as only local tests, to match the tests run in production. In earlier versions, if you enabled tests in your sandbox deployment, you couldn’t exclude managed package tests.

By default, no tests are run in a deployment to a non-production organization, such as a sandbox or a Developer Edition organization. To specify tests to run in your development environment, set a testLevel deployment option. For example, to run local tests in a deployment and to exclude managed package tests, set testLevel on the DeployOptions object to TestLevel.RunLocalTests. Next, pass this object as an argument to the deploy() call as follows.

// Create the DeployOptions object.
DeployOptions deployOptions = new DeployOptions();

// Set the appropriate test level.
deployOptions.setTestLevel(TestLevel.RunLocalTests);

// Call deploy() by passing the deployment options object as an argument.
AsyncResult asyncResult = metadatabinding.deploy(zipBytes,deployOptions);

Note: The RunLocalTests test level is enforced regardless of the contents of the deployment package. In contrast, tests are executed by default in production only if your deployment package contains Apex classes or triggers. You can use RunLocalTests for sandbox and production deployments.

Maintaining User References

User fields are preserved during a metadata deployment.
When a component in your deployment refers to a specific user, such as a recipient of a workflow email notification or a dashboard running user, then Salesforce attempts to locate a matching user in the destination organization by comparing usernames during the deployment.

For example, when you copy data to a sandbox, the fields containing usernames from the production organization are altered to include the sandbox name. In a sandbox named test, the username user@acme.com becomes user@acme.com.test. When you deploy the metadata in the sandbox to another organization, the test in the username is ignored.

For user references in deployments, Salesforce performs the following sequence:

1. Salesforce compares usernames in the source environment to the destination environment and adapts the organization domain name.
2. If two or more usernames match, Salesforce lists the matching names and requests one of the users in the source environment be renamed.
3. If a username in the source environment doesn’t exist in the destination environment, Salesforce displays an error, and the deployment stops until the usernames are removed or resolved to users in the destination environment.
CHAPTER 4   CRUD-Based Metadata Development

Use the CRUD-based metadata calls to create, update, or delete setup and configuration components for your organization or application. These configuration components include custom objects, custom fields, and other configuration metadata. The metadata calls mimic the behavior in the Salesforce user interface for creating, updating, or deleting components. Whatever rules apply there also apply to these calls.

Metadata calls are different from the core, synchronous API calls in the following ways:

- **Metadata API calls are available in a separate WSDL.** To download the WSDL, log into Salesforce, from Setup, enter API in the Quick Find box, then select API and click the Download Metadata WSDL link.
- **After logging in, you must send Metadata API calls to the Metadata API endpoint, which has a different URL than the SOAP API.** Retrieve the metadataServerUrl from the LoginResult returned by your SOAP API login() call. For more information about the SOAP API, see the SOAP API Developer Guide.
- **Metadata calls are either synchronous or asynchronous.** CRUD calls are synchronous in API version 30.0 and later, and similar to the API core calls the results are returned in a single call. In earlier API versions, create, update, and delete are only asynchronous, which means that the results are not immediately returned in one call.
- **There are synchronous metadata calls that map to the corresponding core SOAP API synchronous calls.**
  - createMetadata() maps to the create() SOAP API call.
  - updateMetadata() maps to the update() SOAP API call.
  - deleteMetadata() maps to the delete() SOAP API call.

**Note:** Metadata API also supports retrieve() and deploy() calls for retrieving and deploying metadata components. For more information, see Deploying and Retrieving Metadata.

Java Sample for CRUD-Based Development with Synchronous Calls

This section guides you through a sample Java client application that uses CRUD-based calls. This sample application performs the following main tasks.

1. Uses the MetadataLoginUtil.java class to create a Metadata connection. For more information, see Step 3: Walk Through the Java Sample Code.
2. Calls createMetadata() to create a custom object. This call returns the result in one call.
3. Inspects the returned SaveResult object to check if the operation succeeded, and if it didn’t, writes the component name, error message, and status code to the output.

```java
import com.sforce.soap.metadata.*;

/**
 * Sample that logs in and creates a custom object through the metadata API
 */
public class CRUDSampleCreate {
```
private MetadataConnection metadataConnection;

// one second in milliseconds
private static final long ONE_SECOND = 1000;

public CRUDSampleCreate() {
}

public static void main(String[] args) throws Exception {
    CRUDSampleCreate crudSample = new CRUDSampleCreate();
    crudSample.runCreate();
}

/**
 * Create a custom object. This method demonstrates usage of the
 * create() and checkStatus() calls.
 * @param uniqueName Custom object name should be unique.
 */
private void createCustomObjectSync(final String uniqueName) throws Exception {
    final String label = "My Custom Object";
    CustomObject co = new CustomObject();
    co.setFullName(uniqueName);
    co.setDeploymentStatus(DeploymentStatus.Deployed);
    co.setDescription("Created by the Metadata API Sample");
    co.setEnableActivities(true);
    co.setLabel(label);
    co.setPluralLabel(label + "s");
    co.setSharingModel(SharingModel.ReadWrite);

    // The name field appears in page layouts, related lists, and elsewhere.
    CustomField nf = new CustomField();
    nf.setType(FieldType.Text);
    nf.setDescription("The custom object identifier on page layouts, related lists etc");
    nf.setLabel(label);
    nf.setFullName(uniqueName);
    customObject.setNameField(nf);

    SaveResult[] results = metadataConnection
        .createMetadata(new Metadata[] { co });

    for (SaveResult r : results) {
        if (r.isSuccess()) {
            System.out.println("Created component: " + r.getFullName());
        } else {
            System.out.println("Errors were encountered while creating "
                               + r.getFullName());
            for (Error e : r.getErrors()) {
                System.out.println("Error message: " + e.getMessage());
                System.out.println("Status code: " + e.getStatusCode());
            }
        }
    }
}
private void runCreate() throws Exception {
    metadataConnection = MetadataLoginUtil.login();
    // Custom objects and fields must have __c suffix in the full name.
    final String uniqueObjectName = "MyCustomObject__c";
    createCustomObjectSync(uniqueObjectName);
}

Java Sample for CRUD-Based Development with Asynchronous Calls

⚠️ **Important:** The sample in this section depends on the asynchronous create() CRUD call. Asynchronous CRUD calls are no longer available as of API version 31.0 and are available only in earlier API versions.

This section guides you through a sample Java client application that uses asynchronous CRUD-based calls. This sample application performs the following main tasks:

1. Uses the MetadataLoginUtil.java class to create a Metadata connection. For more information, see Step 3: Walk Through the Java Sample Code.

2. Calls create() to create a new custom object.
   Salesforce returns an AsyncResult object for each component you tried to create. The AsyncResult object is updated with status information as the operation moves from a queue to completed or error state.

3. Calls checkStatus() in a loop until the status value in AsyncResult indicates that the create operation is completed.
   Note the error handling code that follows each API call.

```java
public class CRUDSample {

    private static final long ONE_SECOND = 1000;

    public CRUDSample() {

    }

    public static void main(String[] args) throws Exception {
        CRUDSample crudSample = new CRUDSample();
        crudSample.runCreate();
    }

    /**
     * Create a custom object. This method demonstrates usage of the
     * create() and checkStatus() calls.
     * @param uniqueObjectName Custom object name should be unique.
     */
```
private void createCustomObject(final String uniqueName) throws Exception {
    final String label = "My Custom Object";
    CustomObject customObject = new CustomObject();
    customObject.setFullName(uniqueName);
    customObject.setDeploymentStatus(DeploymentStatus.Deployed);
    customObject.setDescription("Created by the Metadata API Sample");
    customObject.setLabel(label);
    customObject.setPluralLabel(label + "s");
    customObject.setSharingModel(SharingModel.ReadWrite);

    // The name field appears in page layouts, related lists, and elsewhere.
    CustomField nf = new CustomField();
    nf.setType(FieldType.Text);
    nf.setDescription("The custom object identifier on page layouts, related lists etc");
    nf.setLabel(label);
    nf.setFullName(uniqueName);
    customObject.setNameField(nf);

    AsyncResult[] asyncResults = metadataConnection.create(
        new CustomObject[]{customObject});
    if (asyncResults == null) {
        System.out.println("The object was not created successfully");
        return;
    }

    long waitTimeMilliSecs = ONE_SECOND;
    // After the create() call completes, we must poll the results of the checkStatus()
    // call until it indicates that the create operation has completed.
    do {
        printAsyncResultStatus(asyncResults);
        waitTimeMilliSecs *= 2;
        Thread.sleep(waitTimeMilliSecs);
        asyncResults = metadataConnection.checkStatus(new String[]{asyncResults[0].getId()});
    } while (!asyncResults[0].isDone());

    printAsyncResultStatus(asyncResults);
}

private void printAsyncResultStatus(AsyncResult[] asyncResults) throws Exception {
    if (asyncResults == null || asyncResults.length == 0 || asyncResults[0] == null) {
        throw new Exception("The object status cannot be retrieved");
    }

    AsyncResult asyncResult = asyncResults[0]; //we are creating only 1 metadata object

    if (asyncResult.getStatusCode() != null) {
        System.out.println("Error status code: " +
asyncResult.getStatusCode();
System.out.println("Error message: " + asyncResult.getMessage());
}

System.out.println("Object with id: " + asyncResult.getId() + " is " + asyncResult.getState());
}

private void runCreate() throws Exception {
    metadataConnection = MetadataLoginUtil.login();
    // Custom objects and fields must have __c suffix in the full name.
    final String uniqueObjectName = "MyCustomObject__c";
    createCustomObject(uniqueObjectName);
}
Use the REST resource `deployRequest` to move metadata (XML files) between a Salesforce organization and a local file system.

Data in XML files is formatted using the English (United States) locale. This approach ensures that fields that depend on locale, such as date fields, are interpreted consistently during data migrations between organizations using different languages. Organizations can support multiple languages for presentation to their users.

Metadata deployment is used primarily for the following development scenarios.

- Development of a custom application (or customization) in a sandbox organization. After development and testing are completed, the application or customization is then deployed into a production organization using Metadata API.
- Team development of an application in a Developer Edition organization. After development and testing are completed, you can then distribute the application via Lightning Platform AppExchange.

### Working with the Zip File

The `deployRequest` resource is used to deploy a `.zip` file. Within the `.zip` file is a project manifest (`package.xml`) that lists what to retrieve or deploy, and one or more XML components that are organized into folders.

**Note:** A component is an instance of a metadata type. For example, `CustomObject` is a metadata type for custom objects, and the `MyCustomObject__c` component is an instance of a custom object.

The files that are deployed in a `.zip` file might be unpackaged components that reside in your organization (such as `standard objects`). The files might also be packaged components that reside within named packages.

**Note:** You can deploy up to 10,000 files at once. (In API version 43.0 and later, AppExchange packages can contain up to 12,500 files.) The `.zip` file size limit that applies to SOAP calls doesn’t apply to the `deployRequest` REST resource. However, the 400-GB limit for components that are uncompressed into an unzipped folder after upload applies to both SOAP and REST deployments.

Every `.zip` file contains a project manifest, a file that’s named `package.xml`, and a set of directories that contain the components. The manifest file defines the components that you’re trying to retrieve or deploy and the API version used for the deployment or retrieval.

The following is a sample `package.xml` file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
```
The following elements can be defined in `package.xml`.

- `<fullName>` contains the name of the server-side package. If no `<fullName>` exists, this is a client-side unpackaged package.
- `<types>` contains the name of the metadata type (for example, `CustomObject`) and the named members (for example, `myCustomObject__c`) to be deployed. You can add multiple `<types>` elements in a manifest file.
- `<members>` contains the `fullName` of the component, such as `myCustomObject__c`. For many metadata types, you can replace the value in `<members>` with the wildcard character `*` (asterisk) instead of listing each member separately. For a list of metadata types that allow the wildcard character, see the “Allows Wildcard (?)” column in Metadata Types.
  
  Note: You specify Security in the `<members>` element and Settings in the `name` element when retrieving the SecuritySettings component type.
- `<name>` contains the metadata type, for example `CustomObject` or `Profile`. There is one name defined for each metadata type in the directory. Any metadata type that extends `Metadata` is a valid value. The name that’s entered must match a metadata type that’s defined in the Metadata API WSDL. See Metadata Types for a list.
- `<version>` is the API version number that’s used when the `.zip` file is deployed or retrieved. Currently the valid value is `46.0`.

For more sample `package.xml` manifest files that show you how to work with different subsets of metadata, see Sample `package.xml` Manifest Files.

To delete components, see Deleting Components from an Organization.
Deploy Metadata with Apex Testing Using REST

Deploy using the deployRequest REST resource to initiate a request that handles all operations for the deployment.

You can deploy up to 10,000 files at once. (In API version 43.0 and later, AppExchange packages can contain up to 12,500 files.) If the files are uncompressed in an unzipped folder, the size limit is 400 MB.

URI
https://host/services/data/vXX.0/metadata/deployRequest

Formats
JSON

HTTP Method
POST

Authentication
Authorization: Bearer token

deployOptions Parameters

Note: To review the default testing behavior for deployments and approaches that can save time while still enabling you to meet testing requirements, see Running Tests in a Deployment and Run the Same Tests in Sandbox and Production Deployments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowMissingFiles</td>
<td>Boolean. If files that are specified in package.xml are not in the .zip file, specifies whether a deployment can still succeed. Do not set this argument for deployment to production orgs.</td>
</tr>
<tr>
<td>autoUpdatePackage</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>checkOnly</td>
<td>Boolean. Defaults to false. Set to true to perform a test deployment (validation) of components without saving the components in the target org. A validation enables you to verify the results of tests that would be generated in a deployment, but doesn’t commit any changes. After a validation finishes with passing tests, it might qualify for deployment without rerunning tests. See Deploy a Recently Validated Component Set Without Tests.</td>
</tr>
<tr>
<td>ignoreWarnings</td>
<td>Boolean. Indicates whether a deployment is allowed to complete successfully despite one or more warnings (true) or not (false). Defaults to false. The DeployMessage object for a warning contains the following values:</td>
</tr>
<tr>
<td></td>
<td>• problemType—Warning</td>
</tr>
<tr>
<td></td>
<td>• problem—The text of the warning.</td>
</tr>
<tr>
<td></td>
<td>If a warning occurs and ignoreWarnings is set to true, the success field in DeployMessage is true. If ignoreWarnings is set to false, success is set to false and the warning is treated like an error.</td>
</tr>
<tr>
<td>performRetrieve</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>purgeOnDelete</td>
<td>Boolean. If true, the deleted components in the destructiveChanges.xml manifest file aren’t stored in the Recycle Bin. Instead, they become immediately eligible for deletion. This option only works in Developer Edition or sandbox orgs. It doesn’t work in production orgs.</td>
</tr>
<tr>
<td>rollbackOnError</td>
<td>Boolean. Indicates whether any failure causes a complete rollback (true) or not (false). If false, whatever actions can be performed without errors are performed, and errors are returned for the remaining actions. This parameter must be set to true if you are deploying to a production org. The default is false.</td>
</tr>
<tr>
<td>runTests</td>
<td>String[]. A list of Apex tests to run during deployment. Specify the class name, one name per instance. The class name can also specify a namespace with a dot notation. For more information, see Running a Subset of Tests in a Deployment. To use this option, set testLevel to RunSpecifiedTests.</td>
</tr>
<tr>
<td>singlePackage</td>
<td>Boolean. Indicates whether the specified .zip file points to a directory structure with a single package (true) or a set of packages (false).</td>
</tr>
<tr>
<td>testLevel</td>
<td>TestLevel (enumeration of type string). Optional. Specifies which tests are run as part of a deployment. The test level is enforced regardless of the types of components that are present in the deployment package. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>• NoTestRun—No tests are run. This test level applies only to deployments to development environments, such as sandbox, Developer Edition, or trial organizations. This test level is the default for development environments.</td>
</tr>
<tr>
<td></td>
<td>• RunSpecifiedTests—Only the tests that you specify in the runTests option are run. Code coverage requirements differ from the default coverage requirements when using this test level. Each class and trigger in the deployment package must be covered by the executed tests for a minimum of 75% code coverage. This coverage is computed for each class and trigger individually and is different than the overall coverage percentage.</td>
</tr>
<tr>
<td></td>
<td>• RunLocalTests—All tests in your org are run, except the ones that originate from installed managed packages. This test level is the default for production deployments that include Apex classes or triggers.</td>
</tr>
<tr>
<td></td>
<td>• RunAllTestsInOrg—All tests are run. The tests include all tests in your org, including tests of managed packages.</td>
</tr>
<tr>
<td></td>
<td>If you don’t specify a test level, the default test execution behavior is used. See Running Tests in a Deployment.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Apex tests that run as part of a deployment always run synchronously and serially.</td>
</tr>
</tbody>
</table>
Request Body: Deploy Metadata

The JSON request body creates a deployRequest object that has a deployOptions child object that you use to pass in the deployment parameters.

```
{
   "deployOptions" : {
      "allowMissingFiles" : false,
      "autoUpdatePackage" : false,
      "checkOnly" : false,
      "ignoreWarnings" : false,
      "performRetrieve" : false,
      "purgeOnDelete" : false,
      "rollbackOnError" : false,
      "runTests" : null,
      "singlePackage" : true,
      "testLevel" : "RunAllTestsInOrg"
   }
}
```

Response Body: Deploy Metadata

When an HTTP status code of 201 (Created) is returned, your request has succeeded and resulted in the creation of a deployment that is being processed.

```
{
   "id" : "0Afxx000000001VPCAY",
   "deployOptions" : {
      "checkOnly" : false,
      "singlePackage" : false,
      "allowMissingFiles" : false,
      "performRetrieve" : false,
      "autoUpdatePackage" : false,
      "rollbackOnError" : true,
      "ignoreWarnings" : false,
      "purgeOnDelete" : false,
      "runAllTests" : false
   },
   "deployResult" : {
      "id" : "0Afxx000000001VPCAY",
      "success" : false,
      "checkOnly" : false,
      "ignoreWarnings" : false,
      "rollbackOnError" : true,
      "status" : "Pending",
      "runTestsEnabled" : false,
      "done" : false
   }
}
```
## deployResult Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID. ID of the component being deployed.</td>
</tr>
<tr>
<td>canceledBy</td>
<td>ID. The ID of the user who canceled the deployment.</td>
</tr>
<tr>
<td>canceledByName</td>
<td>String. The full name of the user who canceled the deployment.</td>
</tr>
<tr>
<td>checkOnly</td>
<td>Boolean. Indicates whether this deployment is being used to check the validity of the deployed files without making any changes in the org (true) or not (false). A check-only deployment doesn’t deploy any components or change the organization in any way.</td>
</tr>
<tr>
<td>completedDate</td>
<td>DateTime. Timestamp for when the deployment process ended.</td>
</tr>
<tr>
<td>createdBy</td>
<td>ID. The ID of the user who created the deployment.</td>
</tr>
<tr>
<td>createdByName</td>
<td>String. The full name of the user who created the deployment.</td>
</tr>
<tr>
<td>createdDate</td>
<td>DateTime. Timestamp for when the deploy request was received.</td>
</tr>
<tr>
<td>details</td>
<td>DeployDetails. Provides the details of a deployment that is in-progress or ended if ?includeDetails=true is added as a query to the GET request.</td>
</tr>
<tr>
<td>done</td>
<td>Boolean. Indicates whether the server finished processing the deploy request for the specified id.</td>
</tr>
<tr>
<td>errorMessage</td>
<td>String. Message corresponding to the values in the errorStatusCode field, if any.</td>
</tr>
<tr>
<td>errorStatusCode</td>
<td>String. If an error occurred during the deploy request, a status code is returned, and the message corresponding to the status code is returned in errorMessage field.</td>
</tr>
<tr>
<td>ignoreWarnings</td>
<td>Boolean. Optional. Defaults to false. Specifies whether a deployment continues even if the deployment generates warnings. Do not set this argument to true for deployments to production organizations.</td>
</tr>
<tr>
<td>lastModifiedDate</td>
<td>DateTime. Timestamp of the last update for the deployment process.</td>
</tr>
<tr>
<td>numberComponentErrors</td>
<td>Int. The number of components deployed in the deployment process. Use this value with the numberComponentsTotal value to get an estimate of the deployment’s progress.</td>
</tr>
<tr>
<td>numberComponentsTotal</td>
<td>Int. The total number of components in the deployment. Use this value with the numberComponentsDeployed value to get an estimate of the deployment’s progress.</td>
</tr>
<tr>
<td>numberTestErrors</td>
<td>Int. The number of Apex tests that have generated errors during this deployment.</td>
</tr>
<tr>
<td>numberTestsCompleted</td>
<td>The number of completed Apex tests for this deployment. Use this value with the numberTestsTotal value to get an estimate of the deployment’s test progress.</td>
</tr>
<tr>
<td>numberTestsTotal</td>
<td>Int. The total number of Apex tests for this deployment. Use this value with the numberTestsCompleted value to get an estimate of the deployment’s test progress. The value in this field is not accurate until the deployment has started running tests for the components being deployed.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>runTestsEnabled</td>
<td>Boolean. Indicates whether Apex tests were run as part of this deployment (true) or not (false). Tests are either automatically run as part of a deployment or can be set to run in the deployOptions child object.</td>
</tr>
<tr>
<td>rollbackOnError</td>
<td>Boolean. Defaults to true. Indicates whether any failure causes a complete rollback (true) or not (false). If false, whatever set of actions can be performed without errors are performed, and errors are returned for the remaining actions. This parameter must be set to true if you are deploying to a production org.</td>
</tr>
<tr>
<td>startDate</td>
<td>DateTime. Timestamp for when the deployment process began.</td>
</tr>
<tr>
<td>stateDetail</td>
<td>String. Indicates which component is being deployed or which Apex test class is running.</td>
</tr>
</tbody>
</table>
| status                | Indicates the current state of the deployment. The valid values are:  
  • Pending  
  • InProgress  
  • Succeeded  
  • SucceededPartial  
  • Failed  
  • Canceling  
  • Canceled                                                                 |
| success               | Boolean. Indicates whether the deployment was successful (true) or not (false).                                                                                                                                |

**Check the Status of Your Deployment Using REST Resources**

Check the status of your deployment by using passing the deployment request ID in the URL. The response body is similar to that returned by the original deployment request, but it includes information about the deployment in progress.

**URI**

https://host/services/data/vXX.0/metadata/deployRequest/deployRequestId

To include more details in the response, use:

https://host/services/data/vXX.0/metadata/deployRequest/deployRequestId?includeDetails=true

**Formats**

JSON

**HTTP Method**

GET

**Authentication**

Authorization: Bearer **token**
Response Body: Deploy Metadata

The following example shows the response when ?includeDetails=true is added as a query to the GET request.

```
{
    "id" : "0Afxx00000000lWCAQ",
    "url" : "https://host/services/data/vXX.0/metadata/deployRequest/0Afxx00000000lWCAQ?includeDetails=true",
    "deployResult" : {
        "checkOnly" : "false",
        "ignoreWarnings" : "false",
        "rollbackOnError" : "false",
        "status" : "InProgress",
        "numberComponentsDeployed" : "10",
        "numberComponentsTotal" : "1032",
        "numberComponentErrors" : "0",
        "numberTestsCompleted" : "45",
        "numberTestsTotal" : "135",
        "numberTestErrors" : "0",
        "details" : {
            "componentFailures" : [],
            "componentSuccesses" : [],
            "retrieveResult" : null,
            "runTestResults" : {
                "numRun" : 0,
                "successes" : [ ... ],
                "failures" : []
            }
        },
        "createdDate" : "2017-10-10T08:22Z",
        "startDate" : "2017-10-10T08:22Z",
        "lastModifiedDate" : "2017-10-10T08:44Z",
        "completedDate" : "2017-10-10T08:44Z",
        "errorStatusCode" : null,
        "errorMessage" : null,
        "stateDetail" : "Processing Type: Apex Component",
        "createdBy" : "005xx0000001Sv1m",
        "createdByName" : "stephanie stevens",
        "canceledBy" : null,
        "canceledByName" : null,
        "isRunTestsEnabled" : null
    }
    "deployOptions" : {
        "allowMissingFiles" : false,
        "autoUpdatePackage" : false,
        "checkOnly" : true,
        "ignoreWarnings" : false,
    }
```

Check the Status of Your Deployment Using REST Resources
Deploy a Recently Validated Component Set Without Tests

You can deploy components to production in less time by skipping the execution of Apex tests when testing requirements have already been met.

- The components have been validated successfully for the target environment within the last 10 days.
- As part of the validation, Apex tests in the target org have passed.
- Code coverage requirements are met.
  - If all tests in the org or all local tests are run, overall code coverage is at least 75%, and Apex triggers have some coverage.
  - If specific tests are run with the `RunSpecifiedTests` test level, each class and trigger to be deployed is covered by at least 75% individually.

This operation is equivalent to performing a quick deployment of a recent validation on the Deployment Status page in the Salesforce user interface.

To validate but not deploy a set of components when using the `deployRequest` resource, set the `checkOnly` parameter of `deployOptions` to `true`. Note the deployment request ID in the response. Use this ID (associated with a successful validation) later to deploy the component set without repeating the validation.

**URI**

https://host/services/data/vXX.0/metadata/deployRequest/validatedDeployRequestId

**Formats**

JSON

**HTTP Method**

POST

**Authentication**

Authorization: Bearer token

**Request Body: Deploy a Recently Validated Component Set Without Tests**

Note: The HTTP method for deploying a recently validated component set is POST, not PATCH. Using PATCH would create a new deployment.

```json
{
  "validatedDeployRequestId" : "0Afxx00000000lWCAQ"
}
```

If there is no corresponding deployment package that meets the validation requirements, you receive an HTTP status code of 404 (Not Found). If the validated deployment package is found, the HTTP status code returned is 201 (Created).
Response Body: Deploy a Recently Validated Component Set Without Tests

**Note:** The response body from the deployment without validation request includes a new request ID, because it is separate from the earlier request for a validation-only deployment.

```json
{
    "validatedDeployRequestId": "0Afxx000000001WCAQ",
    "id": "0Afxx000000001WMEM",
    "url": "https://host/services/data/vXX.0/metadata/deployRequest/0Afxx000000001WMEM",
    "deployOptions": {
        "allowMissingFiles": false,
        "autoUpdatePackage": false,
        "checkOnly": true,
        "ignoreWarnings": false,
        "performRetrieve": false,
        "purgeOnDelete": false,
        "rollbackOnError": false,
        "runTests": null,
        "singlePackage": true,
        "testLevel": "RunAllTestsInOrg"
    }
}
```

When an HTTP status code of 201 (Created) is returned, your request has succeeded and resulted in the creation of a deployment that is being processed. In the preceding example response body, the ID of the validation-only deployment request is 0Afxx000000001WCAQ; the ID of the deployment without validation request is 0Afxx000000001WMEM.

**Cancel a Deployment in Progress Using REST**

You can request a cancellation of a deployment that's already in progress. Make the cancellation request by patching the status of an ongoing deployRequest. The cancellation is processed asynchronously.

**URI**

`https://host/services/data/vXX.0/metadata/deployRequest/deployRequestId`

**Formats**

JSON

**HTTP Method**

PATCH

**Authentication**

Authorization: Bearer {token}

**Request Body: Request Deployment Cancelation**

The JSON request body for a deployment cancellation includes a PATCH to the status of the original deployRequest.

```json
{
    "deployResult": {
        
```
Response Body: Request Deployment Cancelation

Because the cancellation request is processed asynchronously, the status shown in the response body can be either Canceling or Canceled.

```
{
   "id" : "0Afxx00000000lWCAQ",
   "url" : "https://host/services/data/vXX.0/metadata/deployRequest/0Afxx00000000lWCAQ",
   "deployResult":
   {
      "checkOnly" : "false",
      "ignoreWarnings" : "false",
      "rollbackOnError" : "false",
      "status" : "Canceling",  // or Canceled
      "numberComponentsDeployed" : "10",
      "numberComponentsTotal" : "1032",
      "numberComponentErrors" : "0",
      "numberTestsCompleted" : "45",
      "numberTestsTotal" : "135",
      "numberTestErrors" : "0",
      "details" : {
         "componentFailures" : [],
         "componentSuccesses" : [],
         "retrieveResult" : null,
         "runTestResults" : {
            "numRun" : 0,
            "successes" : [ ... ],
            "failures" : []
         }
      },
      "createdDate" : "2017-10-10T08:22Z",
      "startDate" : "2017-10-10T08:22Z",
      "lastModifiedDate" : "2017-10-10T08:44Z",
      "completedDate" : "2017-10-10T08:44Z",
      "errorStatusCode" : null,
      "errorMessage" : null,
      "stateDetail" : "Processing Type: Apex Component",
      "createdBy" : "005xx000000001Sv1m",
      "createdByName" : "steve stevens",
      "canceledBy" : null,
      "canceledByName" : null,
      "isRunTestsEnabled" : null
   }
}
```

When an HTTP status code of 202 (Accepted) is returned, your cancelation request is in progress or successful.
Use the Metadata REST API with the Salesforce CLI

You can use the Metadata REST API with all deployments, either via the `deployRequest` REST resource or via the Salesforce CLI. You can enable the Salesforce CLI to use Metadata REST API deployments by using a CLI runtime configuration value.

You must have the Modify Metadata Through Metadata API Functions permission or the Modify All Data permission to perform deployments. This requirement includes deployments using Metadata REST API for Deploy.

By default, the Salesforce CLI uses SOAP for deployments. Deployments using REST aren’t bound by the 39 MB .zip file size limit that applies to SOAP deployments. The limit for the number of components that are uncompressed into an unzipped folder after upload is the same for both SOAP and REST deployments.

You enable the Metadata REST API for Deploy in the Salesforce CLI by using a CLI runtime configuration value. For more about setting CLI runtime configuration values, see the Salesforce DX Developer Guide.

- Set the Salesforce CLI to use the Metadata REST API for all deployments in the current project:

  ```
  sfdx force:config:set restDeploy=true
  ```

After Metadata REST API for Deploy is enabled for the Salesforce CLI, deployments initiated using the `mdapi:deploy` or `source:push` commands use REST.
CHAPTER 6   Error Handling

Metadata API calls return error information that your client application can use to identify and resolve runtime errors. The Metadata API provides the following types of error handling:

• Since the Metadata API uses the enterprise or partner WSDLs to authenticate, it uses SOAP fault messages defined in those WSDLs for errors resulting from badly formed messages, failed authentication, or similar problems. Each SOAP fault has an associated ExceptionCode. For more details, see “Error Handling” in the SOAP API Developer Guide.

• For errors with the asynchronous create(), update(), and delete() calls, see the error status code in the statusCode field in the AsyncResult object for the associated component.

• For errors with the synchronous CRUD calls, see the error status code in the statusCode field of the Error object corresponding to each error in the array returned by the errors field of the appropriate result object. For example, the result object of createMetadata() is SaveResult.

• For errors with deploy(), see the problem and success fields in the DeployMessage object for the associated component.

• For errors with retrieve(), see the problem field in the RetrieveMessage object for the associated component.

For sample code, see Step 3: Walk Through the Java Sample Code on page 7.

Error Handling for Session Expiration

When you sign on via the login() call, a new client session begins and a corresponding unique session ID is generated. Sessions automatically expire after the amount of time specified in the Security Controls setup area of the Salesforce application (default two hours). When your session expires, the exception code INVALID_SESSION_ID is returned. If this happens, you must invoke the login() call again. For more information about login(), see the SOAP API Developer Guide.
CHAPTER 7  File-Based Calls

Use file-based calls to deploy or retrieve XML components.

- `deploy()`
- `deployRecentValidation()`
- `retrieve()`

**deploy()**

Uses file representations of components to create, update, or delete those components in a Salesforce org.

**Syntax**

```
AsyncResult = metadatabinding.deploy(base64 zipFile, DeployOptions deployOptions)
```

**Usage**

Use this call to take file representations of components and deploy them into an org by creating, updating, or deleting the components they represent.

**Note:** You can deploy or retrieve up to 10,000 files at once and the maximum size of the deployed or retrieved .zip file is 39 MB. (In API version 43.0 and later, AppExchange packages can contain up to 12,500 files.) Note the following:

- If using the Ant Migration Tool to deploy an unzipped folder, all files in the folder are compressed first. The maximum size of uncompressed components in an unzipped folder is 400 MB or less depending on the compression ratio. If the files have a high compression ratio, you can migrate a total of approximately 400 MB because the compressed size would be under 39 MB. However, if the components can’t be compressed much, like binary static resources, you can migrate less than 400 MB.
- Metadata API base-64 encodes components after they’re compressed. The resulting .zip file can’t exceed 50 MB, which is the limit for SOAP messages. Base-64 encoding increases the size of the payload, so your compressed payload can’t exceed approximately 39 MB before encoding.
- You can perform a `retrieve()` call for a big object only if its index is defined. If a big object is created in Setup and doesn’t yet have an index defined, you can’t retrieve it.

In API version 29.0, Salesforce improved the deployment status properties and removed the requirement to use `checkStatus()` after a `deploy()` call to get information about deployments. Salesforce continues to support the use of `checkStatus()` when using `deploy()` with API version 28.0 or earlier.

For API version 29.0 or later, deploy (create or update) packaged or unpackaged components using the following steps.

1. Issue a `deploy()` call to start the asynchronous deployment. An `AsyncResult` object is returned. Note the value in the `id` field and use it for the next step.
2. Issue a **checkDeployStatus()** call in a loop until the **done** field of the returned **DeployResult** contains **true**, which means that the call is completed. The **DeployResult** object contains information about an in-progress or completed deployment started using the **deploy()** call. When calling **checkDeployStatus()**, pass in the **id** value from the **AsyncResult** object from the first step.

For API version 28.0 or earlier, **deploy** (create or update) packaged or unpackaged components using the following steps:

1. Issue a **deploy()** call to start the asynchronous deployment. An **AsyncResult** object is returned. If the call is completed, the **done** field contains **true**. Most often, the call is not completed quickly enough to be noted in the first result. If it is completed, note the value in the **id** field returned and skip the next step.

2. If the call is not complete, issue a **checkStatus()** call in a loop. In the loop, use the value in the **id** field of the **AsyncResult** object returned by the **deploy()** call in the previous step. Check the **AsyncResult** object which is returned until the **done** field contains **true**. The time taken to complete a **deploy()** call depends on the size of the zip file being deployed. Therefore, use a longer wait time between iterations as the size of the zip file increases.

3. Issue a **checkDeployStatus()** call to obtain the results of the **deploy()** call, using the **id** value returned in the first step.

To track the status of deployments that are in progress or completed in the last 30 days, from Setup, enter **Deployment Status** in the Quick Find box, then select **Deployment Status**.

You can cancel a deployment while it’s in progress or in the queue by clicking **Cancel** next to the deployment. The deployment then has the status **Cancel Requested** until the deployment is completely canceled. A canceled deployment is listed in the Failed section.

The **package.xml** file is a project manifest that lists all the components that you want to retrieve or deploy. You can use **package.xml** to add components. To delete components, add another manifest file. See **Deleting Components from an Organization**.

### Permissions

Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

**Note:** If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

### Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>zipFile</td>
<td>base64</td>
<td>Base 64-encoded binary data. Client applications must encode the binary data as base64.</td>
</tr>
<tr>
<td>deployOptions</td>
<td>DeployOptions</td>
<td>Encapsulates options for determining which packages or files are deployed.</td>
</tr>
</tbody>
</table>

### DeployOptions

The following deployment options can be selected for this call:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowMissingFiles</td>
<td>boolean</td>
<td>If files that are specified in <strong>package.xml</strong> are not in the <strong>.zip</strong> file, specifies whether a deployment can still succeed. Do not set this argument for deployment to <strong>production orgs</strong>.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>autoUpdatePackage</td>
<td>boolean</td>
<td>If a file is in the <code>.zip</code> file but not specified in <code>package.xml</code>, specifies whether the file is automatically added to the package. A <code>retrieve()</code> is issued with the updated <code>package.xml</code> that includes the <code>.zip</code> file. Do not set this argument for deployment to <code>production orgs</code>.</td>
</tr>
<tr>
<td>checkOnly</td>
<td>boolean</td>
<td>Defaults to <code>false</code>. Set to <code>true</code> to perform a test deployment (validation) of components without saving the components in the target org. A validation enables you to verify the results of tests that would be generated in a deployment, but doesn’t commit any changes. After a validation finishes with passing tests, it might qualify for deployment without rerunning tests. See <code>deployRecentValidation()</code>.</td>
</tr>
</tbody>
</table>

**Note:** If you change a field type from Master-Detail to Lookup or vice versa, the change isn’t supported when using the `checkOnly` option to test a deployment. This change isn’t supported for test deployments to avoid permanently altering your data. If a change that isn’t supported for test deployments is included in a deployment package, the test deployment fails and issues an error.

If your deployment package changes a field type from Master-Detail to Lookup or vice versa, you can still validate the changes before you deploy to production. Perform a full deployment to another test sandbox. A full deployment includes a validation of the changes as part of the deployment process.

A Metadata API deployment that includes Master-Detail relationships deletes all detail records in the Recycle Bin in the following cases.

1. For a deployment with a new Master-Detail field, soft delete (send to the Recycle Bin) all detail records before proceeding to deploy the Master-Detail field, or the deployment fails. During the deployment, detail records are permanently deleted from the Recycle Bin and cannot be recovered.

2. For a deployment that converts a Lookup field relationship to a Master-Detail relationship, detail records must reference a master record or be soft-deleted (sent to the Recycle Bin) for the deployment to succeed. However, a successful deployment permanently deletes any detail records in the Recycle Bin.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| ignoreWarnings     | boolean| Indicates whether a warning should allow a deployment to complete successfully (true) or not (false). Defaults to false. The **DeployMessage** object for a warning contains the following values:  
  - problemType—Warning  
  - problem—The text of the warning.  
  If a warning occurs and ignoreWarnings is set to true, the success field in **DeployMessage** is true. If ignoreWarnings is set to false, success is set to false and the warning is treated like an error.  
  This field is available in API version 18.0 and later. Prior to version 18.0, there was no distinction between warnings and errors. All problems were treated as errors and prevented a successful deployment. |
| performRetrieve    | boolean| Indicates whether a **retrieve()** call is performed immediately after the deployment (true) or not (false). Set to true to retrieve whatever was just deployed. |
| purgeOnDelete      | boolean| If true, the deleted components in the **destructiveChanges.xml** manifest file aren't stored in the Recycle Bin. Instead, they become immediately eligible for deletion.  
  This field is available in API version 22.0 and later.  
  This option only works in Developer Edition or sandbox orgs. It doesn’t work in production orgs.  
  **Note:** When you delete a roll-up summary field using Metadata API, the field isn’t saved in the Recycle Bin. The field is purged even if you don’t set the **purgeOnDelete** deployment option to true. |
| rollbackOnError    | boolean| Indicates whether any failure causes a complete rollback (true) or not (false). If false, whatever actions can be performed without errors are performed, and errors are returned for the remaining actions. This parameter must be set to true if you are deploying to a production org. The default is false. |
| runAllTests        | boolean| (Deprecated and only available in API version 33.0 and earlier.) This field defaults to false. Set to true to run all Apex tests after deployment, including tests that originate from installed managed packages.  
  **Note:** Apex tests that run as part of a deployment always run synchronously and serially. |
File-Based Calls

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>runTests</td>
<td>string[]</td>
<td>A list of Apex tests to run during deployment. Specify the class name, one name per instance. The class name can also specify a namespace with a dot notation. For more information, see Running a Subset of Tests in a Deployment. To use this option, set testLevel to RunSpecifiedTests.</td>
</tr>
<tr>
<td>singlePackage</td>
<td>boolean</td>
<td>Indicates whether the specified .zip file points to a directory structure with a single package (true) or a set of packages (false).</td>
</tr>
<tr>
<td>testLevel</td>
<td>TestLevel (enumeration of type string)</td>
<td>Optional. Specifies which tests are run as part of a deployment. The test level is enforced regardless of the types of components that are present in the deployment package. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NoTestRun—No tests are run. This test level applies only to deployments to development environments, such as sandbox, Developer Edition, or trial organizations. This test level is the default for development environments.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RunSpecifiedTests—Only the tests that you specify in the runTests option are run. Code coverage requirements differ from the default coverage requirements when using this test level. Each class and trigger in the deployment package must be covered by the executed tests for a minimum of 75% code coverage. This coverage is computed for each class and trigger individually and is different than the overall coverage percentage.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RunLocalTests—All tests in your org are run, except the ones that originate from installed managed packages. This test level is the default for production deployments that include Apex classes or triggers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RunAllTestsInOrg—All tests are run. The tests include all tests in your org, including tests of managed packages.</td>
</tr>
</tbody>
</table>

If you don’t specify a test level, the default test execution behavior is used. See Running Tests in a Deployment.

Note: Apex tests that run as part of a deployment always run synchronously and serially.

This field is available in API version 34.0 and later.

Response

AsyncResult

50
Sample Code—Java

This sample shows how to deploy components in a zip file. See the `retrieve()` sample code for details on how to retrieve a zip file.

```java
package com.doc.samples;

import java.io.*;
import java.rmi.RemoteException;

import com.sforce.soap.metadata.AsyncResult;
import com.sforce.soap.metadata.DeployDetails;
import com.sforce.soap.metadata.MetadataConnection;
import com.sforce.soap.metadata.DeployOptions;
import com.sforce.soap.metadata.DeployResult;
import com.sforce.soap.metadata.DeployMessage;
import com.sforce.soap.metadata.RunTestsResult;
import com.sforce.soap.metadata.RunTestFailure;
import com.sforce.soap.metadata.CodeCoverageWarning;
import com.sforce.soap.enterprise.LoginResult;
import com.sforce.soap.enterprise.EnterpriseConnection;
import com.sforce.ws.ConnectionException;
import com.sforce.ws.ConnectorConfig;

/**
 * Deploy a zip file of metadata components.
 * Prerequisite: Have a deploy.zip file that includes a package.xml manifest file that
 * details the contents of the zip file.
 */
public class DeploySample {
    // binding for the metadata WSDL used for making metadata API calls
    private MetadataConnection metadataConnection;

    static BufferedReader rdr = new BufferedReader(new InputStreamReader(System.in));

    private static final String ZIP_FILE = "deploy.zip";
    // one second in milliseconds
    private static final long ONE_SECOND = 1000;
    // maximum number of attempts to deploy the zip file
    private static final int MAX_NUM_POLL_REQUESTS = 50;

    public static void main(String[] args) throws Exception {
        final String USERNAME = "user@company.com";
        // This is only a sample. Hard coding passwords in source files is a bad practice.
        final String PASSWORD = "password";
        final String URL = "https://login.salesforce.com/services/Soap/c/29.0";

        DeploySample sample = new DeploySample(USERNAME, PASSWORD, URL);
        sample.deployZip();
    }

    public DeploySample(String username, String password, String loginUrl)
        throws ConnectionException {
```
```java
public void deployZip()
throws RemoteException, Exception
{
    byte zipBytes[] = readZipFile();
    DeployOptions deployOptions = new DeployOptions();
    deployOptions.setPerformRetrieve(false);
    deployOptions.setRollbackOnError(true);
    AsyncResult asyncResult = metadataConnection.deploy(zipBytes, deployOptions);
    String asyncResultId = asyncResult.getId();

    // Wait for the deploy to complete
    int poll = 0;
    long waitTimeMillis = ONE_SECOND;
    DeployResult deployResult = null;
    boolean fetchDetails;
    do {
        Thread.sleep(waitTimeMillis);
        // double the wait time for the next iteration
        waitTimeMillis *= 2;
        if (poll++ > MAX_NUM_POLL_REQUESTS) {
            throw new Exception("Request timed out. If this is a large set " +
                    "of metadata components, check that the time allowed by " +
                    "MAX_NUM_POLL_REQUESTS is sufficient.");
        }

        // Fetch in-progress details once for every 3 polls
        fetchDetails = (poll % 3 == 0);
        deployResult = metadataConnection.checkDeployStatus(asyncResultId, fetchDetails);

        System.out.println("Status is: " + deployResult.getStatus());
        if (!deployResult.isDone() && fetchDetails) {
            printErrors(deployResult, "Failures for deployment in progress:\n");
        }
    } while (!deployResult.isDone());

    if (!deployResult.isSuccess() && deployResult.getErrorStatusCode() != null) {
        throw new Exception(deployResult.getErrorStatusCode() + " msg: " +
                    deployResult.getErrorMessage());
    }

    if (!fetchDetails) {
        // Get the final result with details if we didn't do it in the last attempt.
        deployResult = metadataConnection.checkDeployStatus(asyncResultId, true);
    }

    if (!deployResult.isSuccess()) {
        printErrors(deployResult, "Final list of failures:\n");
        throw new Exception("The files were not successfully deployed");
    }
}
```
System.out.println("The file " + ZIP_FILE + " was successfully deployed");
}

/**
 * Read the zip file contents into a byte array.
 * @return byte[]
 * @throws Exception - if cannot find the zip file to deploy
 */
private byte[] readZipFile()
throws Exception
{
    // We assume here that you have a deploy.zip file.
    // See the retrieve sample for how to retrieve a zip file.
    File deployZip = new File(ZIP_FILE);
    if (!deployZip.exists() || !deployZip.isFile())
        throw new Exception("Cannot find the zip file to deploy. Looking for " +
                        deployZip.getAbsolutePath());

    FileInputStream fos = new FileInputStream(deployZip);
    ByteArrayOutputStream bos = new ByteArrayOutputStream();
    int readbyte = -1;
    while ((readbyte = fos.read()) != -1) {
        bos.write(readbyte);
    }
    fos.close();
    bos.close();
    return bos.toByteArray();
}

/**
 * Print out any errors, if any, related to the deploy.
 * @param result - DeployResult
 */
private void printErrors(DeployResult result, String messageHeader)
{
    DeployDetails deployDetails = result.getDetails();
    StringBuilder errorMessageBuilder = new StringBuilder();
    if (deployDetails != null) {
        DeployMessage[] componentFailures = deployDetails.getComponentFailures();
        for (DeployMessage message : componentFailures) {
            String loc = (message.getLineNumber() == 0 ? "" :
                            "(" + message.getLineNumber() + "," +
                            message.getColumnNumber() + ")");
            if (loc.length() == 0
                && !message.getFileName().equals(message.getFullName())) {
                loc = "(" + message.getFullName() + ");";
            }
            errorMessageBuilder.append(message.getFileName() + loc + ":" +
                            message.getProblem()).append('
');
        }
        RunTestsResult rtr = deployDetails.getRunTestResult();
        if (rtr.getFailures() != null) {
            errorMessageBuilder.append(rtr.getFailures());
        }
    } else {
        errorMessageBuilder.append("No deploy details available.");
    }
    System.out.println(errorMessageBuilder.toString());
}
for (RunTestFailure failure : rtr.getFailures()) {
    String n = (failure.getNamespace() == null ? "" : (failure.getNamespace() + ".") + failure.getName();
    errorMessageBuilder.append("Test failure, method: " + n + "." + failure.getMethodName() + " -- " + failure.getMessage() + " stack " + failure.getStackTrace() + "\n\n");
}

if (rtr.getCodeCoverageWarnings() != null) {
    for (CodeCoverageWarning ccw : rtr.getCodeCoverageWarnings()) {
        errorMessageBuilder.append("Code coverage issue");
        if (ccw.getName() != null) {
            String n = (ccw.getNamespace() == null ? "" : (ccw.getNamespace() + ".") + ccw.getName();
            errorMessageBuilder.append(" class: " + n);
        }
        errorMessageBuilder.append(" -- " + ccw.getMessage() + "\n");
    }
}

if (errorMessageBuilder.length() > 0) {
    errorMessageBuilder.insert(0, messageHeader);
    System.out.println(errorMessageBuilder.toString());
}

private void createMetadataConnection(
    final String username,
    final String password,
    final String loginUrl) throws ConnectionException {

    final ConnectorConfig loginConfig = new ConnectorConfig();
    loginConfig.setAuthEndpoint(loginUrl);
    loginConfig.setServiceEndpoint(loginUrl);
    loginConfig.setManualLogin(true);
    LoginResult loginResult = (new EnterpriseConnection(loginConfig)).login(
        username, password);

    final ConnectorConfig metadataConfig = new ConnectorConfig();
    metadataConfig.setServiceEndpoint(loginResult.getMetadataServerUrl());
    metadataConfig.setSessionId(loginResult.getSessionId());
    this.metadataConnection = new MetadataConnection(metadataConfig);
}
IN THIS SECTION:

1. **Deleting Components from an Organization**
   To delete components, perform a deployment with the `deploy()` call by using a destructive changes manifest file that lists the components to remove from your organization. You can perform a deployment that only deletes components, or a deployment that deletes and adds components. In API version 33.0 and later, you can specify components to delete before and after other components are added or updated. In earlier API versions, if deletions and additions are specified for the same deployment, the `deploy()` call performs the deletions first.

2. **checkDeployStatus()**
   Checks the status of declarative metadata call `deploy()`.

3. **cancelDeploy()**
   Cancels a deployment that hasn't completed yet.

**SEE ALSO:**
- Running Tests in a Deployment

### Deleting Components from an Organization

To delete components, perform a deployment with the `deploy()` call by using a destructive changes manifest file that lists the components to remove from your organization. You can perform a deployment that only deletes components, or a deployment that deletes and adds components. In API version 33.0 and later, you can specify components to delete before and after other components are added or updated. In earlier API versions, if deletions and additions are specified for the same deployment, the `deploy()` call performs the deletions first.

### Deleting Components in a Deployment

To delete components, use the same procedure as with deploying components, but also include a delete manifest file that’s named `destructiveChanges.xml` and list the components to delete in this manifest. The format of this manifest is the same as `package.xml` except that wildcards aren’t supported.

Note: You can’t use `destructiveChanges.xml` to delete items that are associated with an active Lightning page, such as a custom object, a component on the page, or the page itself. First, you must remove the page’s action override by deactivating it in the Lightning App Builder.

The following sample `destructiveChanges.xml` file names a single custom object to be deleted:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>MyCustomObject__c</members>
    <name>CustomObject</name>
  </types>
</Package>
```

To deploy the destructive changes, you must also have a `package.xml` file that lists no components to deploy, includes the API version, and is in the same directory as `destructiveChanges.xml`:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <version>46.0</version>
</Package>
```
Note:

- To bypass the Recycle Bin, set the purgeOnDelete option to true.
- When you delete a roll-up summary field using Metadata API, the field isn’t saved in the Recycle Bin. The field is purged even if you don’t set the purgeOnDelete deployment option to true.
- If you try to delete some components that don’t exist in the organization, the rest of the deletions are still attempted.

Adding and Deleting Components in a Single Deployment

You can perform a deployment that specifies components to delete in destructiveChanges.xml and components to add or update in package.xml. The process is the same as with performing a delete-only deployment except that package.xml contains the components to add or update.

By default, deletions are processed before component additions. In API version 33.0 and later, you can specify components to be deleted before and after component additions. The process is the same as with performing a delete-only deployment except that the name of the deletion manifest file is different.

- To delete components before adding or updating other components, create a manifest file that’s named destructiveChangesPre.xml and include the components to delete.
- To delete components after adding or updating other components, create a manifest file that’s named destructiveChangesPost.xml and include the components to delete.

The ability to specify when deletions are processed is useful when you’re deleting components with dependencies. For example, if a custom object is referenced in an Apex class, you can’t delete it unless you modify the Apex class first to remove the dependency on the custom object. In this example, you can perform a single deployment that updates the Apex class to clear the dependency and then deletes the custom object by using destructiveChangesPost.xml. The following are samples of the package.xml and destructiveChangesPost.xml manifests that would be used in this example.

Sample package.xml, which specifies the class to update:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>SampleClass</members>
        <name>ApexClass</name>
    </types>
    <version>46.0</version>
</Package>
```

Sample destructiveChangesPost.xml, which specifies the custom object to delete after the class update:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>MyCustomObject__c</members>
        <name>CustomObject</name>
    </types>
</Package>
```

Note: The API version that the deployment uses is the API version that’s specified in package.xml.
**checkDeployStatus()**

Checks the status of declarative metadata call `deploy()`.

**Syntax**

```java
DeployResult = metadatabinding.checkDeployStatus(ID id, includeDetails boolean);
```

**Usage**

`checkDeployStatus` is used as part of the process for deploying packaged or unpackaged components to an organization:

1. Issue a `deploy()` call to start the asynchronous deployment. An `AsyncResult` object is returned. Note the value in the `id` field and use it for the next step.

2. Issue a `checkDeployStatus()` call in a loop until the `done` field of the returned `DeployResult` contains `true`, which means that the call is completed. The `DeployResult` object contains information about an in-progress or completed deployment started using the `deploy()` call. When calling `checkDeployStatus()`, pass in the `id` value from the `AsyncResult` object from the first step.

**Note:** Calls to `checkDeployStatus()` don’t count against the API request limits and allocations.

In API version 29.0, Salesforce improved the deployment status properties and removed the requirement to use `checkStatus()` after a `deploy()` call to get information about deployments. Salesforce continues to support the use of `checkStatus()` when using `deploy()` with API version 28.0 or earlier.

**Sample Code—Java**

See the `deploy()` sample code for sample usage of this call.

**Arguments**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID</td>
<td>ID obtained from an <code>AsyncResult</code> object returned by <code>deploy()</code> or a subsequent <code>checkDeployStatus()</code> call.</td>
</tr>
<tr>
<td>includeDetails</td>
<td>boolean</td>
<td>Sets the <code>DeployResult</code> object to include <code>DeployDetails</code> information (<code>true</code>) or not (<code>false</code>). The default is <code>false</code>. Available in API version 29.0 and later.</td>
</tr>
</tbody>
</table>

**Response**

`DeployResult`

**cancelDeploy()**

Cancels a deployment that hasn’t completed yet.
Syntax

```csharp
CancelDeployResult = metadatabinding.cancelDeploy(string id)
```

Usage

Use the `cancelDeploy()` operation to cancel a deployment in your organization started by the `deploy()` operation, which includes deployments started by the Lightning Platform Migration Tool and the Lightning Platform IDE. The deployment can be in a queue waiting to get started, or can be in progress. This operation takes the ID of the deployment you wish to cancel and returns a `CancelDeployResult` object. When the deployment is in the queue and hasn’t started yet, calling `cancelDeploy()` cancels the deployment immediately. When the deployment has started and is in progress, it might not get canceled immediately, so you should call `checkDeployStatus()` to check the status of the cancellation.

Cancel a deployment using these steps.

1. Obtain the ID of the deployment you wish to cancel. For example, you can obtain the ID from the `deploy()` call in the `AsyncResult` object `id` field. Alternatively, you can obtain the ID in the Salesforce user interface from Setup by entering `Deployment Status` in the Quick Find box, selecting `Deployment Status`, and then noting the ID of a deployment started by the API.

2. Issue a `cancelDeploy()` call to start the cancellation process. This call returns a `CancelDeployResult` object.

3. Check the value in the `done` field of the returned `CancelDeployResult`. If the `done` field value is `true`, the deployment has been canceled and you’re done. If the `done` field value is `false`, the cancellation is in progress, and follow these steps to check the cancellation status.
   
   a. Call `checkDeployStatus()` using the deployment ID you obtained earlier.

   b. In the returned `DeployResult` object, check the `status` field. If the status is `Canceling`, this means the cancellation is still in progress, and repeat steps a and b. Otherwise, if the status is `Canceled`, this means the deployment has been canceled and you’re done.

The `deploy()` operation throws these API faults.

- `INVALID_ID_FIELD` with the message **Invalid deploy ID**
  The specified ID argument doesn’t correspond to a valid deployment.

- `INVALID_ID_FIELD` with the message **Deployment already completed**
  The specified deployment has already completed.

Version

Available in API version 30.0 and later.

Permissions

Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

**Note:** If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.
Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>The ID of the deployment to cancel.</td>
</tr>
</tbody>
</table>

Response

CancelDeployResult

Sample Code—Java

This sample shows how to cancel a deployment. The sample calls `cancelDeploy()` by passing it a given deployment ID. Next, it checks whether the cancellation has completed, and if not, calls `checkDeployStatus` in a loop.

```java
public void cancelDeploy(String asyncId) throws Exception {
    // Issue the deployment cancellation request
    CancelDeployResult result = metadataConnection.cancelDeploy(asyncId);

    // If the deployment cancellation completed, write a message to the output.
    if (result.isDone()) {
        System.out.println("Your deployment was canceled successfully!");
    } else {
        // The deployment cancellation is still in progress, so get a new status
        DeployResult deployResult = metadataConnection.checkDeployStatus(asyncId, false);

        // Check whether the deployment is done. If not done, this means
        // that the cancellation is still in progress and the status is Canceling.
        while (!deployResult.isDone()) {
            // Assert that the deployment status is Canceling
            assert deployResult.getStatus() == DeployStatus.Canceling;
            // Wait 2 seconds
            Thread.sleep(2000);
            // Get the deployment status again
            deployResult = metadataConnection.checkDeployStatus(asyncId, false);
        }

        // The deployment is done. Write the status to the output.
        // (When the deployment is done, the cancellation should have completed
        // and the status should be Canceled. However, in very rare cases,
        // the deployment can complete before it is canceled.)
        System.out.println("Final deploy status = " + deployResult.getStatus());
    }
}
```

deployRecentValidation()

Deploys a recently validated component set without running Apex tests.
Syntax

```java
string = metadatabinding.deployRecentValidation(ID validationID)
```

Usage

Use `deployRecentValidation()` to deploy your components to production in less time by skipping the execution of Apex tests. Ensure that the following requirements are met before deploying a recent validation.

- The components have been validated successfully for the target environment within the last 10 days.
- As part of the validation, Apex tests in the target org have passed.
- Code coverage requirements are met.
  - If all tests in the org or all local tests are run, overall code coverage is at least 75%, and Apex triggers have some coverage.
  - If specific tests are run with the `RunSpecifiedTests` test level, each class and trigger that was deployed is covered by at least 75% individually.

This call is equivalent to performing a quick deployment of a recent validation on the Deployment Status page in the Salesforce user interface.

Before you call `deployRecentValidation()`, your organization must have a validation that was recently run. You can run a validation on a set of components by calling `deploy()` with the `checkOnly` property of the `deployOptions` parameter set to `true`. Note the ID that you obtained from the `deploy()` call. You'll use this ID for the `deployRecentValidation()` call in the next step.

After you’ve run a validation successfully, use these steps to quick-deploy the validation to the same target environment.

1. To start an asynchronous quick deployment, call `deployRecentValidation()` and pass it the ID of a recent validation. This ID is obtained from the previous `deploy()` call. The `deployRecentValidation()` call returns the ID of the quick deployment. Note this value. You'll use it in the next step.
2. Check for the completion of the call. This process is similar to that of `deploy()`. Issue a `checkDeployStatus()` call in a loop until the `done` field of the returned `DeployResult` contains `true`, which means that the call is completed. The `DeployResult` object contains information about an in-progress or completed deployment that was started by using the `deployRecentValidation()` call. When calling `checkDeployStatus()`, pass in the ID value that you obtained in the first step.

Version

Available in API version 33.0 and later.

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validationID</td>
<td>string</td>
<td>The ID of a recent validation.</td>
</tr>
</tbody>
</table>

Response

Type: string
The ID of the quick deployment.

Sample Code—Java

```java
package com.salesforce.test.metadata;

import java.rmi.RemoteException;
import com.sforce.soap.metadata.CodeCoverageWarning;
import com.sforce.soap.metadata.DeployDetails;
import com.sforce.soap.metadata.DeployMessage;
import com.sforce.soap.metadata.DeployResult;
import com.sforce.soap.metadata.MetadataConnection;
import com.sforce.soap.metadata.RunTestFailure;
import com.sforce.soap.metadata.RunTestsResult;
import com.sforce.soap.partner.Connector;
import com.sforce.ws.ConnectionException;
import com.sforce.ws.ConnectorConfig;

/**
 * Quick-deploy a recent validation.
 * Prerequisite: A successful validation (check-only deploy) has been done in the org recently.
 */
public class DeployRecentValidationSample {
    // binding for the metadata WSDL used for making metadata API calls
    private MetadataConnection metadataConnection;

    // one second in milliseconds
    private static final long ONE_SECOND = 1000;
    // maximum number of attempts to deploy the zip file
    private static final int MAX_NUM_POLL_REQUESTS = 50;

    public static void main(String[] args) throws Exception {
        final String USERNAME = args[0];
        final String PASSWORD = args[1];
        final String URL = args[2];
        final String recentValidationId = args[3];
        DeployRecentValidationSample sample = new DeployRecentValidationSample(
            USERNAME, PASSWORD, URL);
        sample.deployRecentValidation(recentValidationId);
    }

    public DeployRecentValidationSample(String username, String password, String loginUrl)
        throws ConnectionException {
        createMetadataConnection(username, password, loginUrl);
    }

    public void deployRecentValidation(String recentValidationId)
        throws RemoteException, Exception
```
String asyncResultId = metadataConnection.deployRecentValidation(recentValidationId);

// Wait for the deploy to complete
int poll = 0;
long waitTimeMillisecs = ONE_SECOND;
DeployResult deployResult = null;
boolean fetchDetails;
do {
    Thread.sleep(waitTimeMillisecs);
    // double the wait time for the next iteration
    waitTimeMillisecs *= 2;
    if (poll++ > MAX_NUM_POLL_REQUESTS) {
        throw new Exception("Request timed out. If this is a large set " +
            "of metadata components, check that the time allowed by " +
            "MAX_NUM_POLL_REQUESTS is sufficient.");
    }
    // Fetch in-progress details once for every 3 polls
    fetchDetails = (poll % 3 == 0);
    deployResult = metadataConnection.checkDeployStatus(asyncResultId, fetchDetails);
    System.out.println("Status is: " + deployResult.getStatus());
    if (!deployResult.isDone() && fetchDetails) {
        printErrors(deployResult, "Failures for deployment in progress:
    }
}while (!deployResult.isDone());

if (!deployResult.isSuccess() && deployResult.getErrorStatusCode() != null) {
    throw new Exception(deployResult.getErrorStatusCode() + " msg: " +
        deployResult.getErrorMessage());
}

if (!fetchDetails) {
    // Get the final result with details if we didn't do it in the last attempt.
    deployResult = metadataConnection.checkDeployStatus(asyncResultId, true);
}

if (!deployResult.isSuccess()) {
    printErrors(deployResult, "Final list of failures:
    throw new Exception("The files were not successfully deployed"); 
}

System.out.println("The recent validation " + recentValidationId +
    " was successfully deployed");

/**
 * Print out any errors, if any, related to the deploy.
 * @param result - DeployResult
 */
private void printErrors(DeployResult result, String messageHeader)
DeployDetails deployDetails = result.getDetails();
StringBuilder errorMessageBuilder = new StringBuilder();
if (deployDetails != null) {
    DeployMessage[] componentFailures = deployDetails.getComponentFailures();
    for (DeployMessage message : componentFailures) {
        String loc = (message.getLineNumber() == 0 ? "" :
            "(" + message.getLineNumber() + "," + message.getColumnNumber() + ")");
        if (loc.length() == 0 && !message.getFileName().equals(message.getFullName())) {
            loc = "(" + message.getFullName() + ");"
        }
        errorMessageBuilder.append(message.getFileName() + loc + ":" + message.getProblem()).append('n');
    }
    RunTestsResult rtr = deployDetails.getRunTestResult();
    if (rtr.getFailures() != null) {
        for (RunTestFailure failure : rtr.getFailures()) {
            String n = (failure.getNamespace() == null ? "" :
                (failure.getNamespace() + ",");) + failure.getName();
            errorMessageBuilder.append("Test failure, method: " + n + "," + failure.getMethodName() + " -- " + failure.getMessage() + " stack " + failure.getStackTrace() + 
                "n\n");
        }
    }
    if (rtr.getCodeCoverageWarnings() != null) {
        for (CodeCoverageWarning ccw : rtr.getCodeCoverageWarnings()) {
            errorMessageBuilder.append("Code coverage issue");
            if (ccw.getName() != null) {
                String n = (ccw.getNamespace() == null ? "" :
                    (ccw.getNamespace() + ",");) + ccw.getName();
                errorMessageBuilder.append(" class: " + n);
            }
            errorMessageBuilder.append(" -- " + ccw.getMessage() + "n\n");
        }
    }
}
if (errorMessageBuilder.length() > 0) {
    errorMessageBuilder.insert(0, messageHeader);
    System.out.println(errorMessageBuilder.toString());
}

private void createMetadataConnection(final String username, final String password, final String loginUrl) throws ConnectionException {  
    final ConnectorConfig loginConfig = new ConnectorConfig();
    loginConfig.setUsername(username);
}
loginConfig.setPassword(password);
loginConfig.setAuthEndpoint(loginUrl);

Connector.newConnection(loginConfig);

```java
final ConnectorConfig metadataConfig = new ConnectorConfig();
metadataConfig.setServiceEndpoint(
    loginConfig.getServiceEndpoint().replace("/u/", "/m/"));
metadataConfig.setSessionId(loginConfig.getSessionId());
this.metadataConnection = com.sforce.soap.metadata.Connector.
    newConnection(metadataConfig);
```
1. Issue a `retrieve()` call to start the asynchronous retrieval. An `AsyncResult` object is returned. If the call is completed, the `done` field contains `true`. Most often, the call is not completed quickly enough to be noted in the result. If it is completed, note the value in the `id` field returned and skip the next step.

2. If the call is not complete, issue a `checkStatus()` call in a loop using the value in the `id` field of the `AsyncResult` object, returned by the `retrieve()` call in the previous step. Check the `AsyncResult` object returned until the `done` field contains `true`. The time taken to complete a `retrieve()` call depends on the size of the zip file being deployed, so use a longer wait time between iterations as the size of the zip file increases.

3. Issue a `checkRetrieveStatus()` call to obtain the results of the `retrieve()` call, using the `id` value returned in the first step.

For examples of manifest files, see Sample `package.xml` Manifest Files.

Permissions

Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

Note: If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>retrieveRequest</code></td>
<td><code>RetrieveRequest</code></td>
<td>Encapsulates options for determining which packages or files are retrieved.</td>
</tr>
</tbody>
</table>

Response

`AsyncResult`

Sample Code—Java

This sample shows how to retrieve components into a zip file. See the `deploy()` sample code for details on how to deploy a zip file.

Note: This sample requires API version 34.0 or later.

```java
package com.doc.samples;

import java.io.*;
import java.util.*;
import java.nio.ByteBuffer;
import java.nio.channels.*;
import java.rmi.RemoteException;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;
```
import com.sforce.soap.metadata.AsyncResult;
import com.sforce.soap.metadata.MetadataConnection;
import com.sforce.soap.enterprise.EnterpriseConnection;
import com.sforce.soap.metadata.RetrieveMessage;
import com.sforce.soap.metadata.RetrieveRequest;
import com.sforce.soap.metadata.RetrieveResult;
import com.sforce.soap.metadata.RetrieveStatus;
import com.sforce.soap.enterprise.LoginResult;
import com.sforce.ws.ConnectionException;
import com.sforce.ws.ConnectorConfig;
import com.sforce.soap.metadata.PackageTypeMembers;

public class RetrieveSample {

    // Binding for the metadata WSDL used for making metadata API calls
    private MetadataConnection metadataConnection;

    static BufferedReader rdr = new BufferedReader(new InputStreamReader(System.in));

    // one second in milliseconds
    private static final long ONE_SECOND = 1000;
    // maximum number of attempts to retrieve the results
    private static final int MAX_NUM_POLL_REQUESTS = 50;

    // manifest file that controls which components get retrieved
    private static final String MANIFEST_FILE = "package.xml";

    private static final double API_VERSION = 31.0;

    public static void main(String[] args) throws Exception {
        final String USERNAME = "user@company.com";
        // This is only a sample. Hard coding passwords in source files is a bad practice.
        final String PASSWORD = "password";
        final String URL = "https://login.salesforce.com/services/Soap/c/31.0";

        RetrieveSample sample = new RetrieveSample(USERNAME, PASSWORD, URL);
        sample.retrieveZip();
    }

    public RetrieveSample(String username, String password, String loginUrl)
        throws ConnectionException {
        createMetadataConnection(username, password, loginUrl);
    }

    private void retrieveZip() throws RemoteException, Exception {
        RetrieveRequest retrieveRequest = new RetrieveRequest();
        // The version in package.xml overrides the version in RetrieveRequest
        retrieveRequest.setApiVersion(API_VERSION);
        setUnpackaged(retrieveRequest);
    }
}
// Start the retrieve operation
AsyncResult asyncResult = metadataConnection.retrieve(retrieveRequest);
String asyncResultId = asyncResult.getId();

// Wait for the retrieve to complete
int poll = 0;
long waitTimeMilliSecs = ONE_SECOND;
RetrieveResult result = null;
do {
    Thread.sleep(waitTimeMilliSecs);
    // Double the wait time for the next iteration
    waitTimeMilliSecs *= 2;
    if (poll++ > MAX_NUM_POLL_REQUESTS) {
        throw new Exception("Request timed out. If this is a large set " + 
            "of metadata components, check that the time allowed " + 
            "by MAX_NUM_POLL_REQUESTS is sufficient.");
    }
    result = metadataConnection.checkRetrieveStatus(
        asyncResultId, true);
} while (!result.isDone());

if (result.getStatus() == RetrieveStatus.Failed) {
    throw new Exception(result.getErrorStatusCode() + " msg: " + 
        result.getErrorMessage());
} else if (result.getStatus() == RetrieveStatus.Succeeded) {
    // Print out any warning messages
    StringBuilder buf = new StringBuilder();
    if (result.getMessages() != null) {
        for (RetrieveMessage rm : result.getMessages()) {
            buf.append(rm.getFileName() + " - " + rm.getProblem());
        }
    }
    if (buf.length() > 0) {
        System.out.println("Retrieve warnings:
" + buf);
    }

    // Write the zip to the file system
    System.out.println("Writing results to zip file");
    ByteArrayInputStream bais = new ByteArrayInputStream(result.getZipFile());
    File resultsFile = new File("retrieveResults.zip");
    FileOutputStream os = new FileOutputStream(resultsFile);
    try {
        ReadableByteChannel src = Channels.newChannel(bais);
        FileChannel dest = os.getChannel();
        copy(src, dest);
        System.out.println("Results written to " + resultsFile.getAbsolutePath());
    } finally {
        os.close();
    }
}
/**
 * Helper method to copy from a readable channel to a writable channel,
 * using an in-memory buffer.
 */

private void copy(ReadableByteChannel src, WritableByteChannel dest)
  throws IOException
{
  // Use an in-memory byte buffer
  ByteBuffer buffer = ByteBuffer.allocate(8092);
  while (src.read(buffer) != -1) {
    buffer.flip();
    while(buffer.hasRemaining()) {
      dest.write(buffer);
    }
  }
  buffer.clear();
}

private void setUnpackaged(RetrieveRequest request) throws Exception
{
  // Edit the path, if necessary, if your package.xml file is located elsewhere
  File unpackedManifest = new File(MANIFEST_FILE);
  System.out.println("Manifest file: " + unpackedManifest.getAbsolutePath());
  if (!unpackedManifest.exists() || !unpackedManifest.isFile())
    throw new Exception("Should provide a valid retrieve manifest " +
      "for unpackaged content. " +
      "Looking for " + unpackedManifest.getAbsolutePath());

  // Note that we populate the _package object by parsing a manifest file here.
  // You could populate the _package based on any source for your
  // particular application.
  com.sforce.soap.metadata.Package p = parsePackage(unpackedManifest);
  request.setUnpackaged(p);
}

private com.sforce.soap.metadata.Package parsePackage(File file) throws Exception {
  try {
    InputStream is = new FileInputStream(file);
    List<PackageTypeMembers> pd = new ArrayList<PackageTypeMembers>();
    DocumentBuilder db =
        DocumentBuilderFactory.newInstance().newDocumentBuilder();
    Element d = db.parse(is).getDocumentElement();
    for (Node c = d.getFirstChild(); c != null; c = c.getNextSibling()) {
      if (c instanceof Element) {
        Element ce = (Element)c;
        //
        NodeList namee = ce.getElementsByTagName("name");
        if (namee.getLength() == 0) {
          // not
          continue;
        }
        String name = namee.item(0).getTextContent();
      }
  }
}
NodeList m = ce.getElementsByTagName("members");
List<String> members = new ArrayList<String>();
for (int i = 0; i < m.getLength(); i++) {
    Node mm = m.item(i);
    members.add(mm.getTextContent());
}
PackageTypeMembers pdi = new PackageTypeMembers();
pdi.setName(name);
pdi.setMembers(members.toArray(new String[members.size()]));
pd.add(pdi);
}
com.sforce.soap.metadata.Package r = new com.sforce.soap.metadata.Package();
r.setTypes(pd.toArray(new PackageTypeMembers[pd.size()]));
r.setVersion(API_VERSION + "");
return r;
} catch (ParserConfigurationException pce) {
    throw new Exception("Cannot create XML parser", pce);
} catch (IOException ioe) {
    throw new Exception(ioe);
} catch (SAXException se) {
    throw new Exception(se);
}
}

private void createMetadataConnection(final String username,
        final String password, final String loginUrl)
        throws ConnectionException {

    final ConnectorConfig loginConfig = new ConnectorConfig();
    loginConfig.setAuthEndpoint(loginUrl);
    loginConfig.setServiceEndpoint(loginUrl);
    loginConfig.setManualLogin(true);
    LoginResult loginResult = (new EnterpriseConnection(loginConfig)).login(
            username, password);

    final ConnectorConfig metadataConfig = new ConnectorConfig();
    metadataConfig.setServiceEndpoint(loginResult.getMetadataServerUrl());
    metadataConfig.setSessionId(loginResult.getSessionId());
    this.metadataConnection = new MetadataConnection(metadataConfig);
}

// The sample client application retrieves the user's login credentials.
// Helper function for retrieving user input from the console
String getUserInput(String prompt) {
    System.out.print(prompt);
    try {
        return rdr.readLine();
    } catch (IOException ex) {
        return null;
    }
}
RetrieveRequest

The RetrieveRequest parameter specified on a retrieve() call encapsulates options for determining which packages or files are retrieved.

The RetrieveRequest object consists of the following properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>double</td>
<td>Required. The API version for the retrieve request. The API version determines the fields retrieved for each metadata type. For example, an <code>icon</code> field was added to the CustomTab for API version 14.0. If you retrieve components for version 13.0 or earlier, the components will not include the <code>icon</code> field.</td>
</tr>
<tr>
<td>packageNames</td>
<td>string[]</td>
<td>A list of package names to be retrieved. If you are retrieving only unpackaged components, do not specify a name here. You can retrieve packaged and unpackaged components in the same retrieve.</td>
</tr>
<tr>
<td>singlePackage</td>
<td>boolean</td>
<td>Specifies whether only a single package is being retrieved (true) or not (false). If false, then more than one package is being retrieved.</td>
</tr>
<tr>
<td>specificFiles</td>
<td>string[]</td>
<td>A list of file names to be retrieved. If a value is specified for this property, packageNames must be set to null and singlePackage must be set to true.</td>
</tr>
<tr>
<td>unpackaged</td>
<td>Package</td>
<td>A list of components to retrieve that are not in a package.</td>
</tr>
</tbody>
</table>

checkRetrieveStatus()

Checks the status of the declarative metadata call retrieve() and returns the zip file contents.

Syntax

In API version 34.0 and later:

```java
RetrieveResult = metadatabinding.checkRetrieveStatus(ID id, boolean includeZip);
```
In API version 33.0 and earlier:

```java
RetrieveResult = metadatabinding.checkRetrieveStatus(ID id);
```

**Usage**

Use `checkRetrieveStatus()` to check the progress of the metadata `retrieve()` operation. The `RetrieveResult` object that this method returns indicates when the asynchronous `retrieve()` call is completed. If the retrieval is completed, `RetrieveResult` contains the zip file contents by default. Use the following process to retrieve metadata components with the `retrieve()` call.

1. Issue a `retrieve()` call to start the asynchronous retrieval. An `AsyncResult` object is returned. Note the value in the `id` field and use it for the next step.

2. Issue a `checkRetrieveStatus()` call and pass in the `id` value from the `AsyncResult` object from the first step. Check the value of the `done` field of the returned `RetrieveResult`. If it is `true`, this means that the call is completed and proceed to the next step. Otherwise, repeat this step to call `checkRetrieveStatus()` again until the `done` field is `true`.

3. Retrieve the zip file (`zipFile` field) and other desired fields from `RetrieveResult` that was returned by the final call to `checkRetrieveStatus()` in the previous step.

In API version 31.0 and later, the process of making a `retrieve()` call has been simplified. You no longer have to call `checkStatus()` after a `retrieve()` call to obtain the status of the retrieve operation. Instead, make calls to `checkRetrieveStatus()` only. If the retrieve operation is in progress, call `checkRetrieveStatus()` again until the retrieve operation is completed. The `checkStatus()` call is still supported in versions API version 30.0 or earlier, but is not available in API version 31.0 and later.

**Retrieving the Zip File in a Second Process**

By default, `checkRetrieveStatus()` returns the zip file on the last call to this operation when the retrieval is completed (`RetrieveResult.isDone() == true`) and then deletes the zip file from the server. Subsequent calls to `checkRetrieveStatus()` for the same retrieve operation can’t retrieve the zip file after it has been deleted. Starting with API version 34.0, pass a boolean value for the `includeZip` argument of `checkRetrieveStatus()` to indicate whether to retrieve the zip file. The `includeZip` argument gives you the option to retrieve the file in a separate process after the retrieve operation is completed. For example, a service polls the retrieval status by calling `checkRetrieveStatus(id, false)` in a loop. This call returns the status of the retrieval operation, but doesn’t retrieve the zip file. After the retrieval operation is completed, another process, such as a background file transfer service, calls `checkRetrieveStatus(id, true)` to retrieve the zip file. This last call causes the zip file to be deleted from the server.

```java
// First process: Poll the retrieval but don’t retrieve the zip file.
AsyncResult asyncResult = metadataConnection.retrieve(retrieveRequest);
String asyncResultId = asyncResult.getId();
// Wait for the retrieve to complete
int poll = 0;
long waitTimeMilliSecs = ONE_SECOND;
RetrieveResult result = null;
do {
    Thread.sleep(waitTimeMilliSecs);
    // Check the status but don’t retrieve zip file.
    result = metadataConnection.checkRetrieveStatus(asyncResultId, false);
} while (!result.isDone());

// Second process: Retrieve the zip file.
// For example, this process can be a background file transfer service.
```
// Retrieve the zip file.
result = metadataConnection.checkRetrieveStatus(asyncResultId, true);
// Get the zip file from the RetrieveResult (result) variable
if (result.getStatus() == RetrieveStatus.Succeeded) {
    ByteArrayInputStream bais = new ByteArrayInputStream(result.getZipFile());
    // ...
}

**Sample Code—Java**

See the [retrieve() sample code](#) for sample usage of this call.

**Arguments**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID</td>
<td>ID obtained from anAsyncResult object returned by a retrieve() call or a subsequent RetrieveResult object returned by a checkRetrieveStatus() call.</td>
</tr>
<tr>
<td>includeZip</td>
<td>boolean</td>
<td>Set to true to retrieve the zip file. You can retrieve the zip file only after the retrieval operation is completed. After the zip file is retrieved, it is deleted from the server. Set to false to check the status of the retrieval without attempting to retrieve the zip file. If set to null, this argument defaults to true, which means that the zip file is retrieved on the last call to checkRetrieveStatus() when the retrieval has finished. This argument is available in API version 34.0 and later.</td>
</tr>
</tbody>
</table>

**Response**

RetrieveResult
CHAPTER 8  CRUD-Based Calls

Use CRUD-based calls to work with metadata components in a manner similar to how synchronous API calls in the enterprise WSDL act upon objects.

IN THIS SECTION:

createMetadata()  
Adds one or more new metadata components to your organization synchronously.

readMetadata()  
Returns one or more metadata components from your organization synchronously.

updateMetadata()  
Updates one or more metadata components in your organization synchronously.

upsertMetadata()  
Creates or updates one or more metadata components in your organization synchronously.

deleteMetadata()  
Deletes one or more metadata components from your organization synchronously.

renameMetadata()  
Renames a metadata component in your organization synchronously.

create()  
Deprecated. Adds one or more new metadata components to your organization asynchronously. This call is removed as of API version 31.0 and is available in earlier versions only. Use createMetadata() instead.

delete()  
Deprecated. Deletes one or more components from your organization asynchronously. This call is removed as of API version 31.0 and is available in earlier versions only. Use deleteMetadata() instead.

update()  
Deprecated. Updates one or more components in your organization asynchronously. This call is removed as of API version 31.0 and is available in earlier versions only. Use updateMetadata() or renameMetadata() instead.

createMetadata()  

Adds one or more new metadata components to your organization synchronously.

Syntax

```csharp
SaveResult[] = metadatabinding.createMetadata(Metadata[] metadata);
```
Usage
Use the `createMetadata()` call to create any component that extends Metadata. All components must be of the same type in the same call. For more details, see Metadata Components and Types.

This call executes synchronously, which means that the call returns only when the operation completes.

Starting in API version 34.0, this call supports the `AllOrNoneHeader` header. By default, if `AllOrNoneHeader` isn't used in API version 34.0 and later, this call can save a partial set of records for records with no errors (equivalent to `AllOrNoneHeader=false`). In API version 33.0 and earlier, the default behavior is to only save all records when there are no failures in any record in the call (equivalent to `AllOrNoneHeader=true`).

Version
Available in API version 30.0 and later.

Permissions
Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

Note: If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

Required Fields
Required fields are determined by the metadata components being created. For more information about specific component types, see Metadata Components and Types.

Valid Data Values
You must supply values that are valid for the field's data type, such as integers for integer fields (not alphabetic characters). In your client application, follow the data formatting rules specified for your programming language and development tool. (Your development tool handles the appropriate mapping of data types in SOAP messages.)

String Values
When storing values in string fields, the API trims any leading and trailing whitespace. For example, if the value of a label field is entered as "MyObject ", the value is stored in the database as "MyObject".

Basic Steps for Creating Metadata Components
Use the following process to create metadata components:

1. Design an array and populate it with the components that you want to create. All components must be of the same type.
2. Call `createMetadata()` with the component array passed in as an argument.
3. A `SaveResult` object is returned for each component you tried to create. It contains information about whether the operation was successful, the name of the component created, and any errors returned if the operation wasn't successful.
Sample Code—Java

```java
public void createCustomObjectSync() {
    try {
        CustomObject co = new CustomObject();
        String name = "MyCustomObject1";
        co.setFullName(name + "__c");
        co.setDeploymentStatus(DeploymentStatus.Deployed);
        co.setDescription("Created by the Metadata API");
        co.setEnableActivities(true);
        co.setLabel(name + " Object");
        co.setPluralLabel(co.getLabel() + "s");
        co.setSharingModel(SharingModel.ReadWrite);

        CustomField nf = new CustomField();
        nf.setType(FieldType.Text);
        nf.setLabel(co.getFullName() + " Name");
        co.setNameField(nf);

        SaveResult[] results = metadataConnection
            .createMetadata(new Metadata[] { co });

        for (SaveResult r : results) {
            if (r.isSuccess()) {
                System.out.println("Created component: " + r.getFullName());
            } else {
                System.out.println("Errors were encountered while creating " + r.getFullName());
                for (Error e : r.getErrors()) {
                    System.out.println("Error message: " + e.getMessage());
                    System.out.println("Status code: " + e.getStatusCode());
                }
            }
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>Metadata[]</td>
<td>Array of one or more metadata components. <strong>Limit</strong>: 10. (For CustomMetadata and CustomApplication only, the limit is 200.) You must submit arrays of only one type of component. For example, you can submit an array of 10 custom objects or 10 profiles, but not a mix of both types.</td>
</tr>
</tbody>
</table>
Response

SaveResult[]

**readMetadata()**

Returns one or more metadata components from your organization synchronously.

**Syntax**

```java
ReadResult = metadataConnection.readMetadata(string metadataType, string[] fullNames);
```

**Usage**

Use the `readMetadata()` call to retrieve any component that extends Metadata. All components must be of the same type in the same call. For more details, see Metadata Components and Types.

This call executes synchronously, which means that the call returns only when the operation completes.

**Version**

Available in API version 30.0 and later.

**Permissions**

Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

**Note:** If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

**Basic Steps for Reading Metadata Components**

Use the following process to read metadata components:

1. Determine the metadata type of the components you want to read, and the `fullName` of each component to read. See Metadata for more details on the `fullName` field. You can read only components of the same type in a single call.

2. Invoke the `readMetadata()` call. For the first argument, pass in the name of the metadata type. The metadata type must match one of the values returned by the `describeMetadata()` call. For the second argument, pass in an array of full names corresponding to the components you wish to get. The full names must match one or more full names returned by the `listMetadata()` call.

3. A `ReadResult` is returned that contains an array of `Metadata` components. Cast each returned `Metadata` object to the metadata type you specified in the call to get the component’s properties.

**Sample Code—Java**

```java
public void readCustomObjectSync() {
    try {
```
ReadResult readResult = metadataConnection
    .readMetadata("CustomObject", new String[] {
        "MyCustomObject1__c", "MyCustomObject2__c" });
Metadata[] mdInfo = readResult.getRecords();
System.out.println("Number of component info returned: "+ mdInfo.length);
for (Metadata md : mdInfo) {
    if (md != null) {
        CustomObject obj = (CustomObject) md;
        System.out.println("Custom object full name: "+ obj.getFullName());
        System.out.println("Label: " + obj.getLabel());
        System.out.println("Number of custom fields: "+ obj.getFields().length);
        System.out.println("Sharing model: "+ obj.getSharingModel());
    } else {
        System.out.println("Empty metadata.");
    }
} catch (ConnectionException ce) {
    ce.printStackTrace();
} }

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadataType</td>
<td>string</td>
<td>The metadata type of the components to read.</td>
</tr>
<tr>
<td>fullNames</td>
<td>string[]</td>
<td>Array of full names of the components to read.</td>
</tr>
</tbody>
</table>

Limit: 10. (For CustomMetadata and CustomApplication only, the limit is 200.)
You must submit arrays of only one type of component. For example, you can submit an array
of 10 custom objects or 10 profiles, but not a mix of both types.

Response

ReadResult

updateMetadata() updates one or more metadata components in your organization synchronously.

Syntax

SaveResult[] = metadataConnection.updateMetadata(Metadata[] metadata);
Usage

Use the updateMetadata() call to update any component that extends Metadata. All components must be of the same type in the same call. For more details, see Metadata Components and Types.

This call executes synchronously, which means that the call returns only when the operation completes.

Starting in API version 34.0, this call supports the AllOrNoneHeader header. By default, if AllOrNoneHeader isn’t used in API version 34.0 and later, this call can save a partial set of records for records with no errors (equivalent to AllOrNoneHeader=false). In API version 33.0 and earlier, the default behavior is to only save all records when there are no failures in any record in the call (equivalent to AllOrNoneHeader=true).

Version

Available in API version 30.0 and later.

Permissions

Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

Note: If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

Required Fields

You must supply values for all the required fields in the component.

Valid Field Values

You must supply values that are valid for the field’s data type, such as integers for integer fields (not alphabetic characters). In your client application, follow the data formatting rules specified for your programming language and development tool. (Your development tool handles the appropriate mapping of data types in SOAP messages.)

String Values

When storing values in string fields, the API trims any leading and trailing white space. For example, if the value of a label field is entered as "MyObject " the value is stored in the database as "MyObject".

Basic Steps for Updating Metadata Components

Use this process to update metadata components:

1. Create an array of the components you wish to update. All components must be of the same type.
2. Invoke the updateMetadata() call, passing in the array of metadata components to update.

A SaveResult object is returned for each component you tried to update. It contains information about whether the operation was successful, the name of the component updated, and any errors returned if the operation wasn’t successful.
public void updateCustomObjectSync() {
    try {
        CustomObject co = new CustomObject();
        String name = "MyCustomObject1";
        co.setFullName(name + "__c");
        co.setDeploymentStatus(DeploymentStatus.Deployed);
        co.setDescription("Updated description");
        co.setLabel(name + " Object Update");
        co.setPluralLabel(co.getLabel() + "s");
        co.setSharingModel(SharingModel.ReadWrite);

        // Name field with a type and label is required
        CustomField cf = new CustomField();
        cf.setType(FieldType.Text);
        cf.setLabel(co.getFullName() + " Name");
        co.setNameField(cf);

        SaveResult[] results = metadataConnection.updateMetadata(new Metadata[] { co });

        for (SaveResult r : results) {
            if (r.isSuccess()) {
                System.out.println("Updated component: " + r.getFullName());
            } else {
                System.out.println("Errors were encountered while updating "+ r.getFullName());
                for (Error e : r.getErrors()) {
                    System.out.println("Error message: " + e.getMessage());
                    System.out.println("Status code: " + e.getStatusCode());
                }
            }
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>Metadata[]</td>
<td>Array of one or more metadata components you wish to update.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Limit: 10. (For CustomMetadata and CustomApplication only, the limit is 200.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You must submit arrays of only one type of component. For example, you can</td>
</tr>
<tr>
<td></td>
<td></td>
<td>submit an array of 10 custom objects or 10 profiles, but not a mix of both</td>
</tr>
<tr>
<td></td>
<td></td>
<td>types.</td>
</tr>
</tbody>
</table>
**Response**

`SaveResult[]`

**upsertMetadata()**

Creates or updates one or more metadata components in your organization synchronously.

**Syntax**

```
UpsetResult[] = metadataConnection.upsertMetadata(Metadata[] metadata);
```

**Usage**

Use the `upsertMetadata()` call to create or update any component that extends `Metadata`. All components must be of the same type in the same call. For more details, see `Metadata Components and Types`.

If the specified components already exist in your organization, the `upsertMetadata()` call updates them. Otherwise, `upsertMetadata()` creates these components. Components are matched by the `fullname` field. This call executes synchronously, which means that the call returns only after the operation is completed.

Starting in API version 34.0, this call supports the `AllOrNoneHeader` header. By default, if `AllOrNoneHeader` isn't used in API version 34.0 and later, this call can save a partial set of records for records with no errors (equivalent to `AllOrNoneHeader=false`). In API version 33.0 and earlier, the default behavior is to only save all records when there are no failures in any record in the call (equivalent to `AllOrNoneHeader=true`).

**Version**

Available in API version 31.0 and later.

**Permissions**

Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

**Note:** If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

**Required Fields**

You must supply values for all the required fields in the component.

**Valid Field Values**

You must supply values that are valid for the field’s data type, such as integers (not alphabetic characters) for integer fields. In your client application, follow the data formatting rules that are specified for your programming language and development tool. (Your development tool handles the appropriate mapping of data types in SOAP messages.)
String Values

The API trims any leading and trailing white space when storing values in string fields. For example, if the value of a label field is entered as " MyObject ", the value is stored in the database as "MyObject".

Basic Steps for Upserting Metadata Components

Use this process to upsert metadata components.

1. Create an array of Metadata objects that correspond to the components that you want to create or update. All components must be of the same type.

2. Invoke upsertMetadata(), passing in the array of metadata components that you created in the previous step.

The upsertMetadata() call returns an array of UpsertResult objects. Each returned UpsertResult corresponds to a component that you upserted and contains information about the upsert operation—whether the operation was successful, the name of the component that was upserted, a flag indicating whether the component was created, and any errors that were returned if the operation wasn’t successful.

Sample Code—Java

```java
public void upsertMetadataSample() { 
    try { 
        // Create custom object to upsert
        CustomObject co = new CustomObject();
        String name = "MyCustomObject";
        co.setFullName(name + "__c");
        co.setDeploymentStatus(DeploymentStatus.Deployed);
        co.setDescription("Upserted by the Metadata API");
        co.setEnableActivities(true);
        co.setLabel(name + " Object");
        co.setPluralLabel(co.getLabel() + "s");
        co.setSharingModel(SharingModel.ReadWrite);

        CustomField nf = new CustomField();
        nf.setType(FieldType.Text);
        nf.setLabel("CustomField1");
        co.setNameField(nf);

        // Upsert the custom object
        UpsertResult[] results = metadataConnection.upsertMetadata(new Metadata[]{co});

        for (UpsertResult r : results) {
            if (r.isSuccess()) {
                System.out.println("Success!");
                if (r.isCreated()) {
                    System.out.println("Created component: "
                        + r.getFullName());
                } else {
                    System.out.println("Updated component: "
                        + r.getFullName());
                }
            }
        }
    }
```
} else {
    System.out.println("Errors were encountered while upserting "+ r.getFullName());
    for (Error e : r.getErrors()) {
        System.out.println("Error message: "+ e.getMessage());
        System.out.println("Status code: "+ e.getStatusCode());
    }
}
} catch (ConnectionException ce) {
    ce.printStackTrace();
}

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>Metadata[]</td>
<td>An array of one or more metadata components that you want to create or update</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Limit: 10.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You must submit arrays of only one type of component. For example, you can submit an array of 10 custom objects or 10 profiles, but not a mix of both types.</td>
</tr>
</tbody>
</table>

Response

UpsertResult[]

deleteMetadata()

Deletes one or more metadata components from your organization synchronously.

Syntax

DeleteResult[] = metadataConnection.delete(string metadataType, string[] fullNames);

Usage

Use the deleteMetadata() call to delete any component that extends Metadata. All components must be of the same type in the same call. For more details, see Metadata Components and Types.

This call executes synchronously, which means that the call returns only when the operation completes.
Starting in API version 34.0, this call supports the `AllOrNoneHeader` header. By default, if the `AllOrNoneHeader` isn't used in any API version, this call can delete a partial set of records for records with no errors (equivalent to `AllOrNoneHeader=false`). If `AllOrNoneHeader` is set to `true`, no records are deleted if one or more records cause a failure.

**Version**
Available in API version 30.0 and later.

**Permissions**
Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

**Note:** If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

**Rules and Guidelines**
When deleting components, consider the following rules and guidelines:

- Your client application must be logged in with sufficient access rights to delete individual components within the specified component. For more information, see “Factors that Affect Data Access” in the SOAP API Developer Guide.
- In addition, you might also need permission to access this component’s parent component.
- To ensure referential integrity, this call supports cascading deletions. If you delete a parent component, you delete its children automatically, as long as each child component can be deleted.

**Basic Steps for Deleting Metadata Components**
Use the following process to delete metadata components:

1. Determine the metadata type of the components you want to delete and the `fullName` of each component to delete. You can delete only components of the same type in a single call. The full names must match one or more full names returned by the `listMetadata()` call. See Metadata for more details on the `fullName` field.

2. Invoke the `deleteMetadata()` call. For the first argument, pass in the name of the metadata type. For the second argument, pass in an array of full names corresponding to the components you wish to delete.

A `DeleteResult` object is returned for each component you try to delete. It contains information about whether the operation was successful, the name of the deleted component, and any errors returned if the operation wasn't successful.

**Sample Code—Java**

```java
public void deleteCustomObjectSync() {
    try {
        DeleteResult[] results = metadataConnection.deleteMetadata("CustomObject", new String[] {"MyCustomObject1__c", "MyCustomObject2__c" });
        for (DeleteResult r : results) {
            if (r.isSuccess()) {
                System.out.println("Deleted component: " + r.getFullName());
            } else {

```
System.out
    .println("Errors were encountered while deleting "
        + r.getFullName());
    for (Error e : r.getErrors()) {
        System.out.println("Error message: " + e.getMessage());
        System.out.println("Status code: " + e.getStatusCode());
    }
} catch (ConnectionException ce) {
    ce.printStackTrace();
}
}

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadataType</td>
<td>string</td>
<td>The metadata type of the components to delete.</td>
</tr>
<tr>
<td>fullNames</td>
<td>string[]</td>
<td>Array of full names of the components to delete.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Limit: 10. (For CustomMetadata and CustomApplication only, the limit is 200.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You must submit arrays of only one type of component. For example, you can submit an array of 10 custom objects or 10 profiles, but not a mix of both types.</td>
</tr>
</tbody>
</table>

Response

DeleteResult[]

renameMetadata()

Renames a metadata component in your organization synchronously.

Syntax

SaveResult = metadataConnection.renameMetadata(string metadataType, String oldFullname, String newFullname);

Usage

Use the renameMetadata() call to rename one metadata component in your organization. This call executes synchronously, meaning the call returns only when the operation completes.

You can use this call to rename any of the objects that extend Metadata. For more details, see Metadata Components and Types.
Version

Available in API version 30.0 and later.

Permissions

Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

Note: If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

Basic Steps for Renaming Metadata Components

Use the following process to rename a metadata component:

1. Determine the metadata type of the component you want to rename, its current full name, and the new full name. See Metadata for more details on the fullName field.
2. Invoke the renameMetadata() call. For the first argument, pass in the name of the metadata type. Pass in the old full name as the second argument and the new full name as the last argument.

A SaveResult object is returned that contains information about whether the operation was successful, the name of the renamed component (which is the new name if the renaming was successful), and any errors returned if the operation wasn’t successful.

Sample Code—Java

```java
public void renameCustomObjectSync() {
    try {
        SaveResult[] results = metadataConnection.renameMetadata("CustomObject", "MyCustomObject1__c", "MyCustomObject1New__c");
        for (SaveResult r : results) {
            if (r.isSuccess()) {
                System.out.println("Renamed component: " + r.getName());
            } else {
                System.out.println("Errors were encountered while renaming " + r.getName());
                for (Error e : r.getErrors()) {
                    System.out.println("Error message: " + e.getMessage());
                    System.out.println("Status code: " + e.getStatusCode());
                }
            }
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    } catch (InterruptedException ie) {
        ie.printStackTrace();
    }
}
```
Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadataType</td>
<td>string</td>
<td>The metadata type of the components to rename.</td>
</tr>
<tr>
<td>oldFullName</td>
<td>string</td>
<td>The current component full name.</td>
</tr>
<tr>
<td>newFullName</td>
<td>string</td>
<td>The new component full name.</td>
</tr>
</tbody>
</table>

Response

SaveResult

create()

Deprecated. Adds one or more new metadata components to your organization asynchronously. This call is removed as of API version 31.0 and is available in earlier versions only. Use createMetadata() instead.

Syntax

```
AsyncResult[] = metadatabinding.create(Metadata[] metadata);
```

Usage

Use this call to add one or more metadata components to your organization.

Version

This call is available in API version 30.0 and earlier only. This call is not available in API version 31.0 and later. Use createMetadata() instead.

Permissions

Your client application must be logged in with the “Modify All Data” permission.

Required Fields

Required fields are determined by the metadata components being created. For more information about specific component types, see Metadata Components and Types on page 121.

Valid Data Values

You must supply values that are valid for the field’s data type, such as integers for integer fields (not alphabetic characters). In your client application, follow the data formatting rules specified for your programming language and development tool (your development tool handles the appropriate mapping of data types in SOAP messages).
String Values

When storing values in string fields, the API trims any leading and trailing whitespace. For example, if the value of a label field is entered as "MyObject  " the value is stored in the database as "MyObject".

Basic Steps for Creating Metadata Components

Use the following process to create metadata components:

1. Design an array and populate it with the components you want to create. All components must be of the same type.
2. Call `create()` with the component array passed in as an argument.
3. An `AsyncResult` object is returned for each component you try to create, and is updated with status information as the operation moves from a queue to completed or error state. Call `checkStatus()` in a loop until the status values in `AsyncResult` indicate that all create operations are completed. Start with a wait time of one second between iterations of `checkStatus()` calls, and double the wait time each time you make a subsequent call.

Sample Code—Java

See Step 3: Walk Through the Java Sample Code on page 7 for sample Java code using the `create()` call.

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>Metadata[]</td>
<td>Array of one or more metadata components. limit: 10.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Limit: 10. You must submit arrays of only one type of component. For example, you could submit an array of 10 custom objects or 10 profiles, but not a mix of both types.</td>
</tr>
</tbody>
</table>

Response

`AsyncResult[]`

SEE ALSO:
- `createMetadata()`
- `update()`
- `delete()`
- `checkStatus()`

`delete()`

Deprecated. Deletes one or more components from your organization asynchronously. This call is removed as of API version 31.0 and is available in earlier versions only. Use `deleteMetadata()` instead.
You can use this call to delete any of the objects that extend Metadata. For more details, see Metadata Components and Types on page 121.

**Syntax**

```java
AsyncResult[] = metadataConnection.delete(Metadata[] metadata);
```

**Usage**

Use this call to delete one or more components from your organization.

**Version**

This call is available in API version 30.0 and earlier only. This call is not available in API version 31.0 and later. Use `deleteMetadata()` instead.

**Permissions**

Your client application must be logged in with the “Modify All Data” permission.

**Rules and Guidelines**

When deleting components, consider the following rules and guidelines:

- Your client application must be logged in with sufficient access rights to delete individual components within the specified component. For more information, see “Factors that Affect Data Access” in the SOAP API Developer Guide.
- In addition, you might also need permission to access this component’s parent component.
- To ensure referential integrity, this call supports cascading deletions. If you delete a parent component, you delete its children automatically, as long as each child component can be deleted.

**Basic Steps for Deleting Metadata Components**

Use the following process to delete metadata components:

1. Determine the `fullName` of each component you want to delete. See Metadata for more details on the `fullName` field. You can only delete components of the same type in a single call.

2. Invoke the `delete()` call, passing in the array of metadata components with `fullName` specified.

3. An `AsyncResult` object is returned for each component you try to delete, and is updated with status information as the operation moves from a queue to completed or error state. Call `checkStatus()` in a loop until the status values in `AsyncResult` indicate that all the delete operations are completed. Start with a wait time of one second between iterations of `checkStatus()` calls, and double the wait time each time you make a subsequent call.

**Sample Code—Java**

```java
public void deleteCustomObject() {
    try {
```
CustomObject co = new CustomObject();
co.setFullName("MyCustomObject__c");
AsyncResult[] ars = metadataConnection.create(new Metadata[
    {co}]);
AsyncResult asyncResult = ars[0];
long waitTimeMillisecs = 1000;
while (!asyncResult.isDone()) {
    Thread.sleep(waitTimeMillisecs);
    // double the wait time for the next iteration
    waitTimeMillisecs *= 2;
    asyncResult = mdConnection.checkStatus(
        new String[] {asyncResult.getId()})[0];
    System.out.println("Status is: " + asyncResult.getState());
}
} catch (ConnectionException ce) {
    ce.printStackTrace();
} catch (InterruptedException ie) {
    ie.printStackTrace();
}

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>Metadata[]</td>
<td>Array of one or more metadata components. You only need to set the fullName field in the Metadata object. Limit: 10. You must submit arrays of only one type of component. For example, you could submit an array of 10 custom objects or 10 profiles, but not a mix of both types.</td>
</tr>
</tbody>
</table>

Response

AsyncResult[]

SEE ALSO:
- deleteMetadata()
- create()
- update()
- checkStatus()

update()

Deprecated. Updates one or more components in your organization asynchronously. This call is removed as of API version 31.0 and is available in earlier versions only. Use updateMetadata() or renameMetadata() instead.
This call can be used to update any of the objects that extend Metadata. For more details, see Metadata Components and Types on page 121.

**Syntax**

```java
AsyncResult[] = metadataConnection.update(UpdateMetadata[] metadata);
```

**Usage**

Use this call to update one or more components. This call is analogous to the `ALTER TABLE` statement in SQL.

**Version**

This call is available in API version 30.0 and earlier only. This call is not available in API version 31.0 and later. Use `updateMetadata()` instead to update metadata components or `renameMetadata()` to rename a metadata component.

**Permissions**

Your client application must be logged in with the “Modify All Data” permission.

**Required Fields**

You must supply values for all the required fields in the component.

**Valid Field Values**

You must supply values that are valid for the field’s data type, such as integers for integer fields (not alphabetic characters). In your client application, follow the data formatting rules specified for your programming language and development tool (your development tool handles the appropriate mapping of data types in SOAP messages).

**String Values**

When storing values in string fields, the API trims any leading and trailing white space. For example, if the value of a `label` field is entered as "MyObject " the value is stored in the database as "MyObject".

**Basic Steps for Updating Metadata Components**

Use this process to update metadata components:

1. Create an array of `UpdateMetadata` components and populate it with the components you wish to update. All components must be of the same type.
2. Invoke the `update()` call, passing in the array of metadata components to update.
3. An `AsyncResult` object is returned for each component you try to update, and is updated with status information as the operation moves from a queue to completed or error state. In a loop, call `checkStatus()` until the status values in `AsyncResult` indicate that all the update operations are completed. Start with a wait time of one second between iterations of `checkStatus()` calls, and double the wait time each time you make a subsequent call.
public void updateCustomObject() {
    try {
        CustomObject co = new CustomObject();
        String name = "MyCustomObject";
        co.setFullName(name + "__c");
        co.setDeploymentStatus(DeploymentStatus.Deployed);
        co.setDescription("Created by the Metadata API");
        co.setEnableActivities(true);
        co.setLabel(name + " Object");
        co.setPluralLabel(co.getLabel() + "s");
        co.setSharingModel(SharingModel.ReadWrite);

        CustomField nf = new CustomField();
        nf.setType(FieldType.Text);
        nf.setLabel(co.getFullName() + " Name");
        co.setNameField(nf);

        UpdateMetadata updateMetadata = new UpdateMetadata();
        updateMetadata.setMetadata(co);
        updateMetadata.setCurrentName("TheCurrentName");

        AsyncResult[] ars = metadataConnection.update(new UpdateMetadata[]{ updateMetadata });
        AsyncResult asyncResult = ars[0];
        // set initial wait time to one second in milliseconds
        long waitTimeMillis = 1000;
        while (!asyncResult.isDone()) {
            Thread.sleep(waitTimeMillis);
            // double the wait time for the next iteration
            waitTimeMillis *= 2;
            asyncResult = metadataConnection.checkStatus(
                new String[]{ asyncResult.getId() })[0];
            System.out.println("Status is: " + asyncResult.getState());
        }

        if (asyncResult.getState() != AsyncRequestState.Completed) {
            System.out.println(asyncResult.getStatusCode() + " msg: " + asyncResult.getMessage());
        }
    } catch (InterruptedException ie) {
        ie.printStackTrace();
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>UpdateMetadata[]</td>
<td>Array of one or more UpdateMetadata data structures that represent the components you wish to update. Limit: 10. You must submit arrays of only one type of component. For example, you could submit an array of 10 custom objects or 10 profiles, but not a mix of both types.</td>
</tr>
</tbody>
</table>

UpdateMetadata

One or more UpdateMetadata objects are defined in the metadata argument. This object can be used to update any of the objects that extend Metadata. For more details, see Metadata Components and Types on page 121. Each UpdateMetadata object has the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>currentName</td>
<td>string</td>
<td>The API name of the component or field before the update. For example, if you wanted to update a CustomObject named Foo, the value of this field would be \texttt{Foo__c}. This value is supplied because this call may change the name, and the value here provides mapping.</td>
</tr>
<tr>
<td>metadata</td>
<td>Metadata</td>
<td>Full specification of the component or field you wish to update.</td>
</tr>
</tbody>
</table>

Response

AsyncResult[]

SEE ALSO:
- updateMetadata()
- create()
- delete()
- checkStatus()
CHAPTER 9 Utility Calls

Use utility calls to gather information that is useful for working with the file-based or CRUD-based calls.

- (Deprecated) `checkStatus()`
- `describeMetadata()`
- `describeValueType()`
- `listMetadata()`

### checkStatus()

Deprecated. Checks the status of asynchronous metadata calls `create()`, `update()`, or `delete()`, or the declarative metadata call `retrieve()`. This call is removed as of API version 31.0 and is available only in earlier versions.

**Note:** Starting in API version 29.0, you no longer have to call `checkStatus()` after a `deploy()` call to get information about deployments. Similarly, starting in API version 31.0, you no longer have to call `checkStatus()` after a `retrieve()` call. The `checkStatus()` call has been replaced by `checkDeployStatus()` and `checkRetrieveStatus()` for deploy and retrieve operations respectively.

### Syntax

```java
AsyncResult[] = metadatabinding.checkStatus(ID[] ids);
```

### Usage

Use this call to check whether or not an asynchronous metadata call or declarative metadata call has completed.

### Version

This call is available only in API version 30.0 and earlier. This call is not available in API version 31.0 and later.

### Sample Code—Java

See Step 3: Walk Through the Java Sample Code on page 7 for sample Java code using this call.
Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ids</td>
<td>ID[]</td>
<td>Array of one or more IDs. Each ID is returned in anAsyncResult and corresponds to a component being created, updated, deleted, deployed, or retrieved.</td>
</tr>
</tbody>
</table>

Response

AsyncResult[]

describeMetadata() (Utility Calls)

This call retrieves the metadata that describes your organization. This information includes Apex classes and triggers, custom objects, custom fields on standard objects, tab sets that define an app, and many other metadata types.

Syntax

```java
DescribeMetadataResult = metadataConnection.describeMetadata(double apiVersion);
```

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>double</td>
<td>The API version for which you want metadata; for example, 46.0.</td>
</tr>
</tbody>
</table>

Permissions

Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

⚠️ Note: If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

Sample Code—Java

```java
public void describeMetadata() {
    try {
        double apiVersion = 21.0;
        // Assuming that the SOAP binding has already been established.
        DescribeMetadataResult res =
            metadataConnection.describeMetadata(apiVersion);
        StringBuffer sb = new StringBuffer();
        if (res != null && res.getMetadataObjects().length > 0) {
            for (DescribeMetadataObject obj : res.getMetadataObjects()) {
                sb.append("*******************************************
                XMLName: "+ obj.getXmlName() + "\n");
                sb.append("XMLName: " + obj.getXmlName() + "\n");
```
Response

DescribeMetadataResult

When to Use **describeMetadata()** and **describeValueType()**?

Use the **describeMetadata()** call to get high-level information about all the metadata types that are available for your organization, such as type names and file suffixes. Use the **describeValueType()** call to get granular information about a specific metadata type, such as fields contained within the type.

**describeValueType()**

Retrieves the metadata describing a given metadata type (value type).

describeValueType() accepts a namespace and a type name, and returns a DescribeValueTypeResult object. This call is available in API version 33.0 and later.

**Syntax**

```
DescribeValueTypeResult = connection.describeValueType("{namespace}type_name");
```

**Example**

Describe Apex class metadata in the Metadata namespace:

```
DescribeValueTypeResult = metadataConnection.describeValueType("{http://soap.sforce.com/2006/04/metadata}ApexClass");
```

Describe Apex class metadata in the Tooling namespace:

```
DescribeValueTypeResult = toolingConnection.describeValueType("{urn:metadata.tooling.soap.sforce.com}ApexClass");
```
Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>string</td>
<td>The name of the metadata type for which you want metadata; for example, ApexClass. Include the namespace.</td>
</tr>
</tbody>
</table>

Permissions

Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

Note: If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

Sample Code—Java

The following example describes several metadata types by specifying the Metadata namespace. Each metadata type is described using the helper method, doDescribe(), which calls the describeValueType() Metadata API call. The sample retrieves information from the returned DescribeValueTypeResult: a property, the parent field (if any), and the fields. Next, the sample iterates through the fields and outputs information about each field.

```java
public void describeValueType() throws ConnectionException {
    doDescribe("{http://soap.sforce.com/2006/04/metadata}CustomObject");
    doDescribe("{http://soap.sforce.com/2006/04/metadata}CustomField");
    doDescribe("{http://soap.sforce.com/2006/04/metadata}EmailTemplate");
}

public void doDescribe(String type) throws ConnectionException {
    DescribeValueTypeResult result = metadataConnection.describeValueType(type);
    StringBuffer sb = new StringBuffer();
    sb.append("Describing " + type + " ...
");
    if (result.getApiCreatable() == true) {
        sb.append("Is API creatable.
");
    } else {
        sb.append("Is not API creatable.
");
    }
    ValueTypeField parentField = result.getParentField();
    if (parentField != null) {
        sb.append("** Parent type fields **
");
        if (parentField.getIsForeignKey()) {
            sb.append("This field is a foreign key.
");
            for (String fkDomain : parentField.getForeignKeyDomain()) {
                sb.append("Foreign key domain: " + fkDomain + "
");
            }
        }
    }
    sb.append("** Value type fields **
");
}
for (ValueTypeField field : result.getValueTypeFields()) {
    sb.append("*********************************************\n");
    sb.append("Name: " + field.getName() + "\n");
    sb.append("SoapType: " + field.getSoapType() + "\n");
    if (field.getIsForeignKey()) {
        sb.append("This field is a foreign key.\n");
        for (String fkDomain : field.getForeignKeyDomain()) {
            sb.append("Foreign key domain: " + fkDomain + "\n");
        }
    }
    sb.append("*********************************************\n");
} System.out.println(sb.toString());

To run the previous example with the Tooling WSDL, replace the namespace with the Tooling namespace in the helper function call as follows. Also, use the Tooling connection instead of the Metadata connection to make the describeValueType() call.

doDescribe("{urn:metadata.tooling.soap.sforce.com}CustomObject");
doDescribe("{urn:metadata.tooling.soap.sforce.com}CustomField");
doDescribe("{urn:metadata.tooling.soap.sforce.com}EmailTemplate");

After you run the sample, the output looks similar to the following.

Describing {http://soap.sforce.com/2006/04/metadata}CustomObject ...
Is API creatable.
** Value type fields **
*********************************************
Name: actionOverrides
SoapType: ActionOverride
*********************************************
*********************************************
Name: allowInChatterGroups
SoapType: boolean
*********************************************
*********************************************
Name: articleTypeChannelDisplay
SoapType: ArticleTypeChannelDisplay
*********************************************
*********************************************
Name: businessProcesses
SoapType: BusinessProcess
*********************************************
*********************************************
Name: compactLayoutAssignment
SoapType: string
*********************************************
*********************************************
Name: compactLayouts
SoapType: CompactLayout
*********************************************
*********************************************
Name: customHelp
SoapType: string
This field is a foreign key.
Foreign key domain: ApexPage
Foreign key domain: Scontrol
*******************************************************************************
<The rest of the output for CustomObject has been omitted for brevity.>

Describing {http://soap.sforce.com/2006/04/metadata}CustomField ...
Is API creatable.
** Parent type fields **
This field is a foreign key.
Foreign key domain: CustomObject
** Value type fields **
*******************************************************************************
Name: caseSensitive
SoapType: boolean
*******************************************************************************
*******************************************************************************
Name: defaultValue
SoapType: string
*******************************************************************************
<The rest of the output has been omitted for brevity.>

Response
DescribeValueTypeResult

**listMetadata()**

This call retrieves property information about metadata components in your organization. Data is returned for the components that match the criteria specified in the queries parameter. The queries array can contain up to three ListMetadataQuery queries for each call. This call supports every metadata type: both top-level, such as CustomObject and ApexClass, and child types, such as CustomField and RecordType.

**Syntax**

```java
FileProperties[] = metadataConnection.listMetadata(ListMetadataQuery[] queries, double asOfVersion);
```

**Usage**

This call is useful when you want to identify individual components in package.xml for a retrieve() call or if you want a high-level view of particular metadata types in your organization. For example, you could use this call to return a list of names of all the CustomObject or Layout components in your organization, and use this information to make a subsequent retrieve() call to return a subset of these components. For more information about working with package.xml, see Deploying and Retrieving Metadata on page 16.
Note: This is a synchronous call so the results are returned in one call. This differs from asynchronous calls, such as `retrieve()`, where at least one subsequent call is needed to get the results.

Permissions

Your client application must be logged in with the Modify All Data or Modify Metadata Through Metadata API Functions permission.

Note: If a user requires access to metadata but not to data, you can enable the Modify Metadata Through Metadata API Functions permission to give the access the user needs without providing access to org data.

Sample Code—Java

The sample code below lists information about your custom objects. The code assumes that the SOAP binding has already been established.

```java
public void listMetadata() {
    try {
        ListMetadataQuery query = new ListMetadataQuery();
        query.setType("CustomObject");
        //query.setFolder(null);
        double asOfVersion = 46.0;
        // Assuming that the SOAP binding has already been established.
        FileProperties[] lmr = metadataConnection.listMetadata(
            new ListMetadataQuery[]{query}, asOfVersion);
        if (lmr != null) {
            for (FileProperties n : lmr) {
                System.out.println("Component fullName: " + n.getFullName());
                System.out.println("Component type: " + n.getType());
            }
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queries</td>
<td>ListMetadataQuery[]</td>
<td>A list of objects that specify which components you are interested in.</td>
</tr>
<tr>
<td>asOfVersion</td>
<td>double</td>
<td>The API version for the metadata listing request. If you don’t specify a value in this field, it defaults to the API version specified when you logged in. This field allows you to override the default and set another API version so that, for example, you could list the metadata for a metadata type that was added in a later version than the API version specified when you logged in. This field is available in API version 18.0 and later.</td>
</tr>
</tbody>
</table>

Response

FileProperties
ListMetadataQuery

The `ListMetadataQuery` parameter represents a list of objects that specify which components you are interested in.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>folder</td>
<td>string</td>
<td>The folder associated with the component. This field is required for components that use folders, such as Dashboard, Document, EmailTemplate, or Report.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Required. The metadata type, such as CustomObject, CustomField, or ApexClass.</td>
</tr>
</tbody>
</table>
CHAPTER 10  Result Objects

Use the following objects to get the results of your file-based or CRUD-based calls.

IN THIS SECTION:

- **AsyncResult**
  Contains the ID of a deployment or retrieval. In API version 28.0 and earlier, contains status information of any asynchronous metadata call.

- **CancelDeployResult**
  Contains information about a deployment cancellation—whether the cancellation completed and the deployment ID.

- **DeployResult**
  Contains information about the success or failure of the associated deploy() call.

- **DescribeMetadataResult**
  Contains information about the organization that is useful for developers working with declarative metadata.

- **DescribeValueTypeResult**
  Contains information about a value type that is useful for developers working with declarative metadata.

- **ReadResult**
  Contains result information for the readMetadata call.

- **RetrieveResult**
  Contains information about the success or failure of the associated retrieve() call.

- **SaveResult**
  Contains result information for the createMetadata, updateMetadata, or renameMetadata call.

- **DeleteResult**
  Contains result information for the deleteMetadata call.

- **UpsertResult**
  Contains information about the result of the associated upsertMetadata() call.

- **Error**
  Represents an error that occurred during a synchronous CRUD (createMetadata(), updateMetadata(), or deleteMetadata() operation.

**AsyncResult**

Contains the ID of a deployment or retrieval. In API version 28.0 and earlier, contains status information of any asynchronous metadata call.
**API Version 31.0 and Later**

In API version 31.0, the process of retrieving metadata has been simplified and retrieval properties have been moved to RetrieveResult. Also, the asynchronous create(), update(), and delete() calls have been removed. Therefore, only the id field in AsyncResult is used. The id field is the ID of a deployment or retrieval.

AsyncResult is returned by the following asynchronous calls.

- deploy()
- retrieve()

AsyncResult has the following field that is in use.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID</td>
<td>Required. The ID of the component that’s being deployed or retrieved.</td>
</tr>
</tbody>
</table>

All fields in AsyncResult other than id are deprecated as of API version 31.0. These fields exist but are no longer in use.

- done
- message
- state
- statusCode

**API Versions 29.0 and 30.0**

In API version 29.0, Salesforce moved several properties from the AsyncResult object to the DeployResult object and added several new ones, to improve the process for getting information about deployments. For more information about these changes, see deploy().

In API versions 29.0 and 30.0, AsyncResult is returned by the same asynchronous calls as in API version 28.0 and earlier, but it has different fields.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>done</td>
<td>boolean</td>
<td>Required. Indicates whether the call has been completed (true) or not (false).</td>
</tr>
<tr>
<td>id</td>
<td>ID</td>
<td>Required. The ID of the component that’s being created, updated, deleted, deployed, or retrieved.</td>
</tr>
<tr>
<td>message</td>
<td>string</td>
<td>The message that corresponds to the returned statusCode field, if any.</td>
</tr>
<tr>
<td>state</td>
<td>AsyncRequestState</td>
<td>Required. The AsyncRequestState object has one of four possible values.</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>Queued: This call has not started. It is waiting in a queue.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InProgress: This call has started but has not been completed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Completed: This call has been completed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Error: An error occurred. See the statusCode for more information.</td>
</tr>
</tbody>
</table>
If an error occurred during the `create()`, `update()`, or `delete()` call, a status code is returned, and the message that corresponds to the status code is returned in the `message` field.

For a description of each StatusCode value, see “StatusCode” in the SOAP API Developer Guide.

### API Version 28.0 and Earlier

In API version 28.0 and earlier, AsyncResult is returned by the following asynchronous calls.

- `deploy()`
- `retrieve()`
- `create()`
- `update()`
- `delete()`

Use the `checkStatus()` call against each object to discover when the call is completed for that object. Salesforce updates each AsyncResult object as the call is completed or when errors occur.

Similarly, the `deploy()` and `retrieve()` calls use AsyncResult, though you must subsequently use `checkDeployStatus()` or `checkRetrieveStatus()` respectively to get more status information for the deployment or retrieval.

AsyncResult has the following fields.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>checkOnly</code></td>
<td>boolean</td>
<td>Indicates whether this deployment is being used to check the validity of the deployed files without making any changes in the organization (true) or not (false). A check-only deployment does not deploy any components or change the organization in any way. This field is available in API version 16.0 and later and is relevant only for the <code>deploy()</code> call.</td>
</tr>
<tr>
<td><code>done</code></td>
<td>boolean</td>
<td>Required. Indicates whether the call has been completed (true) or not (false).</td>
</tr>
<tr>
<td><code>id</code></td>
<td>ID</td>
<td>Required. The ID of the component that’s being created, updated, deleted, deployed, or retrieved.</td>
</tr>
<tr>
<td><code>message</code></td>
<td>string</td>
<td>The message that corresponds to the returned <code>statusCode</code> field, if any.</td>
</tr>
<tr>
<td><code>numberComponentErrors</code></td>
<td>int</td>
<td>The number of components that generated errors during this deployment. This field is available in API version 16.0 and later and is relevant only for the <code>deploy()</code> call.</td>
</tr>
<tr>
<td><code>numberComponentsDeployed</code></td>
<td>int</td>
<td>The number of components that have been deployed for this deployment. This field in conjunction with the <code>numberComponentsTotal</code> field gives you an indication of the progress of the deployment. This field is available in API version 16.0 and later and is relevant only for the <code>deploy()</code> call.</td>
</tr>
</tbody>
</table>
### Result Objects

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numberComponentsTotal</td>
<td>int</td>
<td>The total number of components in the deployment. This field in conjunction with the numberComponentsDeployed field gives you an indication of the progress of the deployment. This field is available in API version 16.0 and later and is relevant only for the deploy() call.</td>
</tr>
<tr>
<td>numberTestErrors</td>
<td>int</td>
<td>The number of Apex tests that generated errors during this deployment. This field is available in API version 16.0 and later and is relevant only for the deploy() call.</td>
</tr>
<tr>
<td>numberTestsCompleted</td>
<td>int</td>
<td>The number of Apex tests that have been completed for this deployment. This field in conjunction with the numberTestsTotal field gives you an indication of the progress of tests for the deployment. This field is available in API version 16.0 and later and is relevant only for the deploy() call.</td>
</tr>
<tr>
<td>numberTestsTotal</td>
<td>int</td>
<td>The total number of Apex tests in the deployment. This field in conjunction with the numberTestsCompleted field gives you an indication of the progress of tests for the deployment. The value in this field is not accurate until the deployment has started running tests for the components that are being deployed. This field is available in API version 16.0 and later and is relevant only for the deploy() call.</td>
</tr>
<tr>
<td>secondsToWait</td>
<td>int</td>
<td>This field is no longer supported for API version 13.0 and later and is provided only for backward compatibility. The field was removed in API version 17.0. Indicates the number of seconds before the call is likely to be completed. This is an estimate only. A reasonable approach is to wait one second before calling checkStatus() to see if the operation is complete. Double your wait time for each successive iteration of checkStatus() calls until the operation is complete.</td>
</tr>
</tbody>
</table>
| state                 | AsyncRequestState (enumeration of type string) | Required. The AsyncRequestState object has one of four possible values.  
- Queued: This call has not started. It is waiting in a queue.  
- InProgress: This call has started but has not been completed.  
- Completed: This call has been completed.  
- Error: An error occurred. See the statusCode for more information. |
| stateDetail           | string       | Indicates which component is being deployed or which Apex test class is running. This field is available in API version 16.0 and later and is relevant only for the deploy() call. |
| stateDetailLastModifiedDate | dateTime | The date and time when the stateDetail field was last modified. This field is available in API version 16.0 and later and is relevant only for the deploy() call. |
| statusCode            | StatusCode (enumeration of type string) | If an error occurred during the create(), update(), delete(), or deploy() call, a status code is returned, and the message that corresponds to the status code is returned in the message field. 
For a description of each StatusCode value, see “StatusCode” in the SOAP API Developer Guide. |
CancelDeployResult

Contains information about a deployment cancellation—whether the cancellation completed and the deployment ID.

The asynchronous metadata call `cancelDeploy()` returns a `CancelDeployResult` object.

Version

Available in API version 30.0 and later.

`CancelDeployResult` has the following properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>done</td>
<td>boolean</td>
<td>Indicates whether the deployment cancellation, which is started through <code>cancelDeploy()</code>, has completed (<code>true</code>) or not (<code>false</code>). When a deployment hasn’t started yet and is still in the queue, the deployment is canceled immediately with the <code>cancelDeploy()</code> call and this field returns <code>true</code>. Otherwise, this field returns <code>false</code> when the cancellation is in progress.</td>
</tr>
<tr>
<td>id</td>
<td>ID</td>
<td>ID of the deployment being canceled.</td>
</tr>
</tbody>
</table>

DeployResult

Contains information about the success or failure of the associated `deploy()` call.

The asynchronous metadata call `checkDeployStatus()` returns a `DeployResult` object.

In API version 29.0, Salesforce moved several properties from the `AsyncResult` object to the `DeployResult` object to improve the process for getting information about deployments. For more information about these changes, see `deploy()` on page 46.

For API version 29.0 and later, the `DeployResult` object has the following properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID</td>
<td>ID of the component being deployed.</td>
</tr>
<tr>
<td>canceledBy</td>
<td>ID</td>
<td>The ID of the user who canceled the deployment.</td>
</tr>
<tr>
<td>canceledByName</td>
<td>string</td>
<td>The full name of the user who canceled the deployment.</td>
</tr>
<tr>
<td>checkOnly</td>
<td>boolean</td>
<td>Indicates whether this deployment is being used to check the validity of the deployed files without making any changes in the organization (<code>true</code>) or not (<code>false</code>). A check-only deployment does not deploy any components or change the organization in any way.</td>
</tr>
<tr>
<td>completedDate</td>
<td>dateTime</td>
<td>Timestamp for when the deployment process ended.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>createdBy</td>
<td>ID</td>
<td>The ID of the user who created the deployment. This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>createdByName</td>
<td>string</td>
<td>The full name of the user who created the deployment. This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>createdDate</td>
<td>dateTime</td>
<td>Timestamp for when the deploy() call was received.</td>
</tr>
<tr>
<td>details</td>
<td>DeployDetails[]</td>
<td>Provides the details of a deployment that is in-progress or ended, if the includeDetails parameter is set to true in the checkDeployStatus() call.</td>
</tr>
<tr>
<td>done</td>
<td>boolean</td>
<td>Indicates whether the server finished processing the deploy() call for the specified id.</td>
</tr>
<tr>
<td>errorMessage</td>
<td>string</td>
<td>Message corresponding to the values in the errorStatusCode field, if any.</td>
</tr>
<tr>
<td>errorStatusCode</td>
<td>string</td>
<td>If an error occurred during the deploy() call, a status code is returned, and the message corresponding to the status code is returned in the errorMessage field. For a description of each StatusCode value, see “StatusCode” in the SOAP API Developer Guide.</td>
</tr>
<tr>
<td>ignoreWarnings</td>
<td>boolean</td>
<td>Optional. Defaults to false. Specifies whether a deployment should continue even if the deployment generates warnings. Do not set this argument to true for deployments to production organizations.</td>
</tr>
<tr>
<td>lastModifiedDate</td>
<td>dateTime</td>
<td>Timestamp of the last update for the deployment process.</td>
</tr>
<tr>
<td>numberComponentErrors</td>
<td>int</td>
<td>The number of components that generated errors during this deployment.</td>
</tr>
<tr>
<td>numberComponentsDeployed</td>
<td>int</td>
<td>The number of components deployed in the deployment process. Use this value with the numberComponentsTotal value to get an estimate of the deployment’s progress.</td>
</tr>
<tr>
<td>numberComponentsTotal</td>
<td>int</td>
<td>The total number of components in the deployment. Use this value with the numberComponentsDeployed value to get an estimate of the deployment’s progress.</td>
</tr>
<tr>
<td>numberTestErrors</td>
<td>int</td>
<td>The number of Apex tests that have generated errors during this deployment.</td>
</tr>
<tr>
<td>numberTestsCompleted</td>
<td>int</td>
<td>The number of completed Apex tests for this deployment. Use this value with the numberTestsTotal value to get an estimate of the deployment’s test progress.</td>
</tr>
<tr>
<td>numberTestsTotal</td>
<td>int</td>
<td>The total number of Apex tests for this deployment. Use this value with the numberTestsCompleted value to get an estimate of the deployment’s test progress. The value in this field is not accurate until the deployment has started running tests for the components being deployed.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>runTestsEnabled</td>
<td>boolean</td>
<td>Indicates whether Apex tests were run as part of this deployment (true) or not (false). Tests are either automatically run as part of a deployment or can be set to run in <code>DeployOptions</code> for the <code>deploy()</code> call. For information on when tests are automatically run, see Running Tests in a Deployment. This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>rollbackOnError</td>
<td>boolean</td>
<td>Optional. Defaults to true. Indicates whether any failure causes a complete rollback (true) or not (false). If false, whatever set of actions can be performed without errors are performed, and errors are returned for the remaining actions. This parameter must be set to true if you are deploying to a production organization.</td>
</tr>
<tr>
<td>startDate</td>
<td>dateTime</td>
<td>Timestamp for when the deployment process began.</td>
</tr>
<tr>
<td>stateDetail</td>
<td>string</td>
<td>Indicates which component is being deployed or which Apex test class is running.</td>
</tr>
</tbody>
</table>
| status            | DeployStatus (enumeration of type string) | Indicates the current state of the deployment. The valid values are:  
- Pending  
- InProgress  
- Succeeded  
- SucceededPartial  
- Failed  
- Canceling  
- Canceled |
| success           | boolean      | Indicates whether the deployment was successful (true) or not (false).                                                                     |

**DeployDetails**

These fields provide more information for the `details` field of the `DeployResult` object, if the `includeDetails` parameter is set to (true in the `deploy()` call.

**Note:** While a deployment is still in-progress, the `DeployDetails` object only contains `componentFailures` data. After the deployment process finishes, the other fields populate with the data for the entire deployment.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentFailures</td>
<td>DeployMessage[]</td>
<td>One or more DeployMessage objects containing deployment errors for each component.</td>
</tr>
<tr>
<td>componentSuccesses</td>
<td>DeployMessage[]</td>
<td>One or more DeployMessage objects containing successful deployment details for each component.</td>
</tr>
<tr>
<td>retrieveResult</td>
<td>RetrieveResult</td>
<td>If the <code>performRetrieve</code> parameter was specified for the <code>deploy()</code> call, a <code>retrieve()</code> call is performed immediately after the <code>deploy()</code> process completes. This field contains the results of that retrieval.</td>
</tr>
</tbody>
</table>
### DeployResult

For API version 28.0 and earlier, the DeployResult object has the following properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID</td>
<td>ID of the component being deployed.</td>
</tr>
<tr>
<td>messages</td>
<td>DeployMessage[]</td>
<td>Contains information about the success or failure of a <code>deploy()</code> call.</td>
</tr>
<tr>
<td>retrieveResult</td>
<td>RetrieveResult</td>
<td>If the <code>performRetrieve</code> parameter was specified for the <code>deploy()</code> call, a <code>retrieve()</code> call is performed immediately after the <code>deploy()</code> process completes. This field contains the results of that retrieval.</td>
</tr>
<tr>
<td>runTestResult</td>
<td>RunTestsResult</td>
<td>If tests were run for the <code>deploy()</code> call, this field contains the test results.</td>
</tr>
<tr>
<td>success</td>
<td>boolean</td>
<td>Indicates whether the deployment was successful (<code>true</code>) or not (<code>false</code>).</td>
</tr>
</tbody>
</table>

**DeployMessage**

Each DeployResult object contains one or more DeployMessage objects. Each DeployMessage object contains information about the deployment success or failure of a component in the deployment `.zip` file:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>changed</td>
<td>boolean</td>
<td>If <code>true</code>, the component was changed as a result of this deployment. If <code>false</code>, the deployed component was the same as the corresponding component already in the organization.</td>
</tr>
<tr>
<td>componentName</td>
<td>string</td>
<td>The metadata type of the component in this deployment. This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>created</td>
<td>boolean</td>
<td>If <code>true</code>, the component was created as a result of this deployment. If <code>false</code>, the component was either deleted or modified as a result of the deployment.</td>
</tr>
<tr>
<td>createdDate</td>
<td>dateTime</td>
<td>The date and time when the component was created as a result of this deployment. This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>deleted</td>
<td>boolean</td>
<td>If <code>true</code>, the component was deleted as a result of this deployment. If <code>false</code>, the component was either new or modified as result of the deployment.</td>
</tr>
<tr>
<td>fileName</td>
<td>string</td>
<td>The name of the file in the <code>.zip</code> file used to deploy this component.</td>
</tr>
</tbody>
</table>
**Result Objects**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fullName</td>
<td>string</td>
<td>The full name of the component. Inherited from <code>Metadata</code>, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See <code>createMetadata()</code> to see an example of this field specified for a call.</td>
</tr>
<tr>
<td>id</td>
<td>ID</td>
<td>ID of the component being deployed.</td>
</tr>
<tr>
<td>lineNumber</td>
<td>int</td>
<td>Each component is represented by a text file. If an error occurred during deployment, this field represents the line number of the text file where the error occurred.</td>
</tr>
<tr>
<td>problem</td>
<td>string</td>
<td>If an error or warning occurred, this field contains a description of the problem that caused the compile to fail.</td>
</tr>
</tbody>
</table>
| problemType   | `DeployProblemType` (enumeration of type string) | Indicates the problem type. The problem details are tracked in the `problem` field. The valid values are:
  * Warning
  * Error
This field is available in API version 18.0 and later. Prior to version 18.0, there was no distinction between warnings and errors. All problems were treated as errors and prevented a successful deployment. |
| success       | boolean     | Indicates whether the component was successfully deployed (true) or not (false).                                                            |

**RunTestsResult**

Contains information about the execution of unit tests, including whether unit tests were completed successfully, code coverage results, and failures.

A `RunTestsResult` object has the following properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apexLogId</td>
<td>string</td>
<td>The ID of an ApexLog object that is created at the end of a test run. The ApexLog object is created if there is an active trace flag on the user running an Apex test, or on a class or trigger being executed. This field is available in API version 35.0 and later.</td>
</tr>
<tr>
<td>codeCoverage</td>
<td><code>CodeCoverageResult[]</code></td>
<td>An array of one or more <code>CodeCoverageResult</code> objects that contains the details of the code coverage for the specified unit tests.</td>
</tr>
<tr>
<td>codeCoverageWarnings</td>
<td><code>CodeCoverageWarning[]</code></td>
<td>An array of one or more code coverage warnings for the test run. The results include both the total number of lines that could have been executed, as well as the number, line, and column positions of code that was not executed.</td>
</tr>
<tr>
<td>failures</td>
<td><code>RunTestFailure[]</code></td>
<td>An array of one or more <code>RunTestFailure</code> objects that contain information about the unit test failures, if there are any.</td>
</tr>
</tbody>
</table>
### Result Objects

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flowCoverage</td>
<td>FlowCoverageResult[]</td>
<td>An array of results from test runs that executed flows. This field is available in API version 44.0 and later.</td>
</tr>
<tr>
<td>flowCoverageWarnings</td>
<td>FlowCoverageWarning[]</td>
<td>An array of warnings generated by test runs that executed flows. This field is available in API version 44.0 and later.</td>
</tr>
<tr>
<td>numFailures</td>
<td>int</td>
<td>The number of failures for the unit tests.</td>
</tr>
<tr>
<td>numTestsRun</td>
<td>int</td>
<td>The number of unit tests that were run.</td>
</tr>
<tr>
<td>successes</td>
<td>RunTestSuccess[]</td>
<td>An array of one or more RunTestSuccess objects that contain information about successes, if there are any.</td>
</tr>
<tr>
<td>totalTime</td>
<td>double</td>
<td>The total cumulative time spent running tests. This can be helpful for performance monitoring.</td>
</tr>
</tbody>
</table>

### CodeCoverageResult

The RunTestsResult object contains this object. It contains information about whether or not the compile of the specified Apex and run of the unit tests was successful.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dmlInfo</td>
<td>CodeLocation[]</td>
<td>For each class or trigger tested, for each portion of code tested, this property contains the DML statement locations, the number of times the code was executed, and the total cumulative time spent in these calls. This can be helpful for performance monitoring.</td>
</tr>
<tr>
<td>id</td>
<td>ID</td>
<td>The ID of the CodeLocation. The ID is unique within an organization.</td>
</tr>
<tr>
<td>locationsNotCovered</td>
<td>CodeLocation[]</td>
<td>For each class or trigger tested, if any code is not covered, the line and column of the code not tested, and the number of times the code was executed.</td>
</tr>
<tr>
<td>methodInfo</td>
<td>CodeLocation[]</td>
<td>For each class or trigger tested, the method invocation locations, the number of times the code was executed, and the total cumulative time spent in these calls. This can be helpful for performance monitoring.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the class or trigger covered.</td>
</tr>
<tr>
<td>namespace</td>
<td>string</td>
<td>The namespace that contained the unit tests, if one is specified.</td>
</tr>
<tr>
<td>numLocations</td>
<td>int</td>
<td>The total number of code locations.</td>
</tr>
</tbody>
</table>
For each class or trigger tested, the location of SOQL statements in the code, the number of times this code was executed, and the total cumulative time spent in these calls. This can be helpful for performance monitoring.

Do not use. In early, unsupported releases, used to specify class or package.

**CodeCoverageWarning**

The `RunTestsResult` object contains this object. It contains information about the Apex class which generated warnings.

This object has the following properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID</td>
<td>The ID of the <code>CodeLocation</code>. The ID is unique within an organization.</td>
</tr>
<tr>
<td>message</td>
<td>string</td>
<td>The message of the warning generated.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The namespace that contained the unit tests, if one is specified.</td>
</tr>
<tr>
<td>namespace</td>
<td>string</td>
<td>The namespace that contained the unit tests, if one is specified.</td>
</tr>
</tbody>
</table>

**RunTestFailure**

The `RunTestsResult` object returns information about failures during the unit test run.

This object has the following properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID</td>
<td>The ID of the class which generated failures.</td>
</tr>
<tr>
<td>message</td>
<td>string</td>
<td>The failure message.</td>
</tr>
<tr>
<td>methodName</td>
<td>string</td>
<td>The name of the method that failed.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the class that failed.</td>
</tr>
<tr>
<td>namespace</td>
<td>string</td>
<td>The namespace that contained the class, if one was specified.</td>
</tr>
<tr>
<td>seeAllData</td>
<td>boolean</td>
<td>Indicates whether the test method has access to organization data (true) or not (false).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 33.0 and later.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>stackTrace</td>
<td>string</td>
<td>The stack trace for the failure.</td>
</tr>
<tr>
<td>time</td>
<td>double</td>
<td>The time spent running tests for this failed operation. This can be helpful for performance monitoring.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Do not use. In early, unsupported releases, used to specify class or package.</td>
</tr>
</tbody>
</table>

**FlowCoverageResult**

This object contains information about the flow version and the number of elements executed by the test run. This object is available in API version 44.0 and later.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>elementsNotCovered</td>
<td>string</td>
<td>List of elements in the flow version that weren’t executed by the test run.</td>
</tr>
<tr>
<td>flowId</td>
<td>string</td>
<td>The ID of the flow version. The ID is unique within an org.</td>
</tr>
<tr>
<td>flowName</td>
<td>string</td>
<td>The name of the flow that was executed by the test run.</td>
</tr>
<tr>
<td>flowNamespace</td>
<td>string</td>
<td>The namespace that contains the flow, if one is specified.</td>
</tr>
<tr>
<td>numElements</td>
<td>int</td>
<td>The total number of elements in the flow version.</td>
</tr>
<tr>
<td>runElementsNotCovered</td>
<td>int</td>
<td>The number of elements in the flow version that weren’t executed by the test run</td>
</tr>
<tr>
<td>processType</td>
<td>FlowProcessType (enumeration of type string)</td>
<td>The process type of the flow version.</td>
</tr>
</tbody>
</table>

**FlowCoverageWarning**

This object contains information about the flow version that generated warnings. This object is available in API version 44.0 and later.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flowId</td>
<td>string</td>
<td>The ID of the flow version that generated the warning.</td>
</tr>
<tr>
<td>flowName</td>
<td>string</td>
<td>The name of the flow that generated the warning. If the warning applies to the overall test coverage of flows within your org, this value is null.</td>
</tr>
<tr>
<td>flowNamespace</td>
<td>string</td>
<td>The namespace that contains the flow, if one was specified.</td>
</tr>
<tr>
<td>message</td>
<td>string</td>
<td>The message of the warning that was generated.</td>
</tr>
</tbody>
</table>

**RunTestSuccess**

The RunTestsResult object returns information about successes during the unit test run.
This object has the following properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID</td>
<td>The ID of the class which generated the success.</td>
</tr>
<tr>
<td>methodName</td>
<td>string</td>
<td>The name of the method that succeeded.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the class that succeeded.</td>
</tr>
<tr>
<td>namespace</td>
<td>string</td>
<td>The namespace that contained the unit tests, if one is specified.</td>
</tr>
<tr>
<td>seeAllData</td>
<td>boolean</td>
<td>Indicates whether the test method has access to organization data (true) or not (false). This field is available in API version 33.0 and later.</td>
</tr>
<tr>
<td>time</td>
<td>double</td>
<td>The time spent running tests for this operation. This can be helpful for performance monitoring.</td>
</tr>
</tbody>
</table>

**CodeLocation**

The RunTestsResult object contains this object in a number of fields.

This object has the following properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>column</td>
<td>int</td>
<td>The column location of the Apex tested.</td>
</tr>
<tr>
<td>line</td>
<td>int</td>
<td>The line location of the Apex tested.</td>
</tr>
<tr>
<td>numExecutions</td>
<td>int</td>
<td>The number of times the Apex was executed in the test run.</td>
</tr>
<tr>
<td>time</td>
<td>double</td>
<td>The total cumulative time spent at this location. This can be helpful for performance monitoring.</td>
</tr>
</tbody>
</table>

**DescribeMetadataResult**

Contains information about the organization that is useful for developers working with declarative metadata.

The describeMetadata() call returns a DescribeMetadataResult object.

Each DescribeMetadataResult object has the following properties:
DescribeMetadataObject

This object is returned as part of the DescribeMetadataResult. Each DescribeMetadataObject has the following properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>childXmlNames</td>
<td>string[]</td>
<td>List of child sub-components for this component.</td>
</tr>
<tr>
<td>directoryName</td>
<td>string</td>
<td>The name of the directory in the .zip file that contains this component.</td>
</tr>
<tr>
<td>inFolder</td>
<td>boolean</td>
<td>Indicates whether the component is in a folder (true) or not (false). For example, documents, email templates and reports are stored in folders.</td>
</tr>
<tr>
<td>metaFile</td>
<td>boolean</td>
<td>Indicates whether the component requires an accompanying metadata file. For example, documents, classes, and s-controls are components that require an additional metadata file.</td>
</tr>
<tr>
<td>suffix</td>
<td>string</td>
<td>The file suffix for this component.</td>
</tr>
<tr>
<td>xmlName</td>
<td>string</td>
<td>The name of the root element in the metadata file for this component. This name also appears in the Packages &gt; types &gt; name field in the manifest file package.xml.</td>
</tr>
</tbody>
</table>

DescribeValueTypeResult

Contains information about a value type that is useful for developers working with declarative metadata.

The describeValueType() call returns a DescribeValueTypeResult object.

Each DescribeValueTypeResult object has the following properties.
### apiCreatable
- **Type:** boolean
- **Description:** Indicates whether this value type can be created through the `createMetadata()` call (true) or not (false). This field is available in API version 36.0 and later.

### apiDeletable
- **Type:** boolean
- **Description:** Indicates whether this value type can be created through the `deleteMetadata()` call (true) or not (false). This field is available in API version 36.0 and later.

### apiReadable
- **Type:** boolean
- **Description:** Indicates whether this value type can be created through the `readMetadata()` call (true) or not (false). This field is available in API version 36.0 and later.

### apiUpdatable
- **Type:** boolean
- **Description:** Indicates whether this value type can be created through the `updateMetadata()` call (true) or not (false). This field is available in API version 36.0 and later.

### parentField
- **Type:** `ValueTypeField`
- **Description:** Information about the parent of this value type. Parent field information is useful for metadata types that are specified with the parent in their name, such as custom fields, email templates, workflow rules, and reports. For example, the full name of a custom field includes the sObject that contains it (for example, `Account.field1__c`). Similarly, the full name of an email template includes the folder where the template is stored (for example, `MyFolder/EmailTemplate1`). If the value type has no parent, this field is null. This field is available in API version 36.0 and later.

### valueTypeFields
- **Type:** `ValueTypeField[]`
- **Description:** One or more metadata components and their attributes.

---

### ValueTypeField
This object is returned as part of the `DescribeValueTypeResult` and represents the metadata for one field. Each `ValueTypeField` has the following properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td><code>ValueTypeField</code></td>
<td>The <code>ValueTypeField</code> object for the next field, if any.</td>
</tr>
<tr>
<td>foreignKeyDomain</td>
<td><code>string</code></td>
<td>If <code>isForeignKey</code> is True, <code>foreignKeyDomain</code> is the type of object, such as <code>Account</code> or <code>Opportunity</code>.</td>
</tr>
<tr>
<td>isForeignKey</td>
<td><code>boolean</code></td>
<td>True if the field is a foreign key. That means this field is the primary key in a different database table.</td>
</tr>
<tr>
<td>isNameField</td>
<td><code>boolean</code></td>
<td>True if this value type field is a <code>fullName</code> field, otherwise False.</td>
</tr>
<tr>
<td>minOccurs</td>
<td><code>int</code></td>
<td>1 if this field is required, 0 otherwise.</td>
</tr>
</tbody>
</table>
## Description

### TypeName

The name of this value type field. The name is null for parent fields.

### picklistValues

The individual picklist values if the field is a picklist.

### soapType

The data type of the field, such as boolean or double.

### valueRequired

Required. Indicates whether this value type field must have a value (true) or can be null (false).

## ReadResult

Contains result information for the `readMetadata` call.

## Version

Available in API version 30.0 and later.

## Properties

### records

An array of metadata components returned from `readMetadata()`.

## RetrieveResult

Contains information about the success or failure of the associated `retrieve()` call.

The metadata `retrieve()` call returns a RetrieveResult object.

Each RetrieveResult object has the following fields:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>done</td>
<td>boolean</td>
<td>Required. Indicates whether the <code>retrieve()</code> call is completed (true) or not (false). This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>errorMessage</td>
<td>string</td>
<td>If an error occurs during the <code>retrieve()</code> call, this field contains a descriptive message about this error. This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>errorStatusCode</td>
<td>StatusCode</td>
<td>If an error occurs during the <code>retrieve()</code> call, this field contains the status code for this error. This field is available in API version 31.0 and later. For a description of each StatusCode value, see “StatusCode” in the SOAP API Developer Guide.</td>
</tr>
<tr>
<td>fileProperties</td>
<td>FileProperties[]</td>
<td>Contains information about the properties of each component in the .zip file, and the manifest file package.xml. One object per component is returned.</td>
</tr>
</tbody>
</table>
RetrieveResult

Result Objects

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID</td>
<td>ID of the component being retrieved.</td>
</tr>
<tr>
<td>messages</td>
<td>RetrieveMessage[]</td>
<td>Contains information about the success or failure of the retrieve() call.</td>
</tr>
<tr>
<td>status</td>
<td>RetrieveStatus</td>
<td>The status of the retrieve() call. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>Pending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InProgress</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Succeeded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Failed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>success</td>
<td>boolean</td>
<td>Indicates whether the retrieve() call was successful (true) or not (false). This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>zipFile</td>
<td>base64Binary</td>
<td>The zip file returned by the retrieve request. Base 64-encoded binary data. Prior to making an API call, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to binary. This conversion is usually handled for you by a SOAP client.</td>
</tr>
</tbody>
</table>

FileProperties

This component contains information about the properties of each component in the .zip file, and the manifest file package.xml. One object per component is returned. Note that this component does not contain information about any associated metadata files in the .zip file, only the component files and manifest file. FileProperties contains the following properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>createdById</td>
<td>string</td>
<td>Required. ID of the user who created the file.</td>
</tr>
<tr>
<td>createdByName</td>
<td>string</td>
<td>Required. Name of the user who created the file.</td>
</tr>
<tr>
<td>createdDate</td>
<td>dateTime</td>
<td>Required. Date and time when the file was created.</td>
</tr>
<tr>
<td>fileName</td>
<td>string</td>
<td>Required. Name of the file.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The file developer name used as a unique identifier for API access. The value is based on the fileName but the characters allowed are more restrictive. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.</td>
</tr>
<tr>
<td>id</td>
<td>string</td>
<td>Required. ID of the file.</td>
</tr>
<tr>
<td>lastModifiedById</td>
<td>string</td>
<td>Required. ID of the user who last modified the file.</td>
</tr>
<tr>
<td>lastModifiedByName</td>
<td>string</td>
<td>Required. Name of the user who last modified the file.</td>
</tr>
<tr>
<td>lastModifiedDate</td>
<td>dateTime</td>
<td>Required. Date and time that the file was last modified.</td>
</tr>
</tbody>
</table>
Indicates the manageable state of the specified component if it is contained in a package:
- beta
- deleted
- deprecated
- installed
- released
- unmanaged

For more information about states of manageability for components in Lightning Platform AppExchange packages, see “Planning the Release of Managed Packages” in the Salesforce online help.

If any, the namespace prefix of the component.

Required. The metadata type, such as CustomObject, CustomField, or ApexClass.

### RetrieveMessage

RetrieveResult returns this object, which contains information about the success or failure of the retrieve() call. One object per problem is returned:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileName</td>
<td>string</td>
<td>The name of the file in the retrieved .zip file where a problem occurred.</td>
</tr>
<tr>
<td>problem</td>
<td>string</td>
<td>A description of the problem that occurred.</td>
</tr>
</tbody>
</table>

SEE ALSO:
- retrieve()

### SaveResult

Contains result information for the createMetadata, updateMetadata, or renameMetadata call.

### Version

Available in API version 30.0 and later.
Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errors</td>
<td>Error[]</td>
<td>An array of errors returned if the operation wasn’t successful.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The full name of the component processed.</td>
</tr>
<tr>
<td>success</td>
<td>boolean</td>
<td>Indicates whether the operation was successful (true) or not (false).</td>
</tr>
</tbody>
</table>

DeleteResult

Contains result information for the deleteMetadata call.

Version

Available in API version 30.0 and later.

Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errors</td>
<td>Error[]</td>
<td>An array of errors returned if the operation wasn’t successful.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The full name of the deleted component.</td>
</tr>
<tr>
<td>success</td>
<td>boolean</td>
<td>Indicates whether the deletion was successful (true) or not (false).</td>
</tr>
</tbody>
</table>

UpsertResult

Contains information about the result of the associated upsertMetadata() call.

Version

Available in API version 31.0 and later.

Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>created</td>
<td>boolean</td>
<td>Indicates whether the upsert operation resulted in the creation of the component (true) or not (false). If false and the upsert operation was successful, this means that the component was updated.</td>
</tr>
<tr>
<td>errors</td>
<td>Error[]</td>
<td>An array of errors that were returned if the operation wasn’t successful.</td>
</tr>
</tbody>
</table>
### Error Result Objects

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fullName</td>
<td>string</td>
<td>The full name of the component that was created or updated if the operation was successful.</td>
</tr>
<tr>
<td>success</td>
<td>boolean</td>
<td>Indicates whether the operation was successful (true) or not (false).</td>
</tr>
</tbody>
</table>

### Error

Represents an error that occurred during a synchronous CRUD (createMetadata(), updateMetadata(), or deleteMetadata()) operation.

### Version

Available in API version 30.0 and later.

### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>extendedErrorDetails</td>
<td>ExtendedErrorDetails</td>
<td>More details about the error, including an extended error code and extra error properties, when available. Reserved for future use. For a description of the ExtendedErrorDetails element, see “ExtendedErrorDetails” in the SOAP API Developer Guide.</td>
</tr>
<tr>
<td>fields</td>
<td>string[]</td>
<td>An array containing names of fields that affected the error condition.</td>
</tr>
<tr>
<td>message</td>
<td>string</td>
<td>The error message text.</td>
</tr>
<tr>
<td>statusCode</td>
<td>StatusCode</td>
<td>A status code corresponding to the error. For a description of each StatusCode value, see “StatusCode” in the SOAP API Developer Guide.</td>
</tr>
</tbody>
</table>
CHAPTER 11  Metadata Types

Metadata API enables you to access some entities and feature settings that you can customize in the user interface.

Note:

- Metadata type names are case-sensitive. Specifying a type name with an invalid case results in a deployment error.
- Metadata types don't always correspond directly to their related data types. In some cases, the information is accessible but not organized as expected. For example, dependent picklists are exposed as a type of picklist, not a separate metadata type.
- The wildcard character doesn't apply to metadata types for feature settings, like AccountSettings. The wildcard applies only when retrieving all settings and not an individual setting. See Settings.

Metadata Components and Types

Metadata components are not based on sObjects, like objects in the API. Instead, they are based on metadata types, such as ApexClass and CustomObject, which extend Metadata, the base class for all metadata types. A component is an instance of a metadata type.

For example, CustomObject is a metadata type for custom objects, and the MyCustomObject__c component is an instance of a custom object.

A metadata type can be identified in the metadata WSDL as any complexType that extends the Metadata complexType. A complexType that is a metadata type includes the following element in its WSDL definition:

```xml
<xsd:extension base="tns:Metadata">
```

CustomObject and BusinessProcess extend Metadata so they are metadata types; ActionOverride doesn't extend Metadata so it's not a metadata type.

You can individually deploy or retrieve a component for a metadata type. For example, you can retrieve an individual BusinessProcess component, but you can't retrieve an individual ActionOverride component. You can only retrieve an ActionOverride component by retrieving its encompassing CustomObject component.

Metadata components can be manipulated by asynchronous Metadata API calls or declarative (or file-based) Metadata API calls.

Most of the components can be accessed using Force.com IDE. Exceptions are noted in the description of the object.

Field Data Types

Each component field has a specific field type. These field types can correspond to other components defined in the WSDL, or primitive data types, like string, that are commonly used in strongly typed programming languages.

These field data types are used in the SOAP messages that are exchanged between your client application and the API. When writing your client application, follow the data typing rules defined for your programming language and development environment. Your development tool handles the mapping of typed data in your programming language with these SOAP data types.

For more information about primitive data types, see the SOAP API Developer Guide.
Enumeration Fields

Some component fields have a data type that is an enumeration. An enumeration is the API equivalent of a picklist. The valid values of the field are restricted to a strict set of possible values, all having the same data type. These values are listed in the field description column for each enumeration field. See `sortBy` for an example of an enumeration field of type string. The XML below shows a sample definition of an enumeration of type string in the WSDL.

```xml
<xsd:simpleType name="DashboardComponentFilter">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="RowLabelAscending"/>
        <xsd:enumeration value="RowLabelDescending"/>
        <xsd:enumeration value="RowValueAscending"/>
        <xsd:enumeration value="RowValueDescending"/>
    </xsd:restriction>
</xsd:simpleType>
```

Supported Calls

All of the metadata types are supported by the main calls, unless it is stated otherwise in the individual component sections. The main Metadata API calls are:

- CRUD calls, such as `createMetadata()` and `deleteMetadata()`
- File-based calls, such as `deploy()` and `retrieve()`
- Utility calls, such as `listMetadata()` and `describeMetadata()`

Metadata Coverage Report

Launch the Metadata Coverage report to determine supported metadata components. The Metadata Coverage report is the ultimate source of truth for metadata coverage across several channels. These channels include Metadata API, scratch org source tracking, unlocked packages, second-generation managed packages, classic managed packages, and more.

To view the Metadata Coverage report, you don’t have to be logged into an org.

Unsupported Metadata Types

Some things you can customize in a Salesforce org aren’t available in Metadata API.

The following components can’t be retrieved or deployed with Metadata API, and changes to them must be made manually in each of your organizations:

- Account Teams
- Activity Button Overrides
- Analytic Settings
- Automated Case User Settings
- Auto-number on Customizable Standard Fields
- Calendars
- Campaign Influences
- Case Contact Roles
- Case Feed Layouts
- Case Team Roles
- Console Layouts
- Multiline layout fields for contract line items
- Currency Exchange Rates
- Data Category Visibility Settings
- Delegated Administration
- Divisions
- Fiscal Year
- File Upload and Download Security Settings
- Lead Settings
- Chats routed with Omni-Channel
- Mail Merge Templates
- Mobile Administration
- Mobile Users and Devices
- Multiline layout fields for opportunity teams
- Offline Briefcase Configurations
- Opportunity Big Deal Alerts
- Opportunity Update Reminders
- Organization Wide Email Addresses
- Outlook Configurations
- Partner Management
- The following standard picklists: IdeaTheme.Categories, Order.Status, Question.Origin. (All other standard picklists are supported.)
- Predefined Case Teams
- Product Schedule Setup
- Quote Templates
- Salesforce to Salesforce
- Self-Service Portal Font and Colors
- Self-Service Portal Settings
- Self-Service Portal Users
- Self-Service Public Solutions
- Self-Service Web-to-Case
- Service report templates
- Site.com
- Social Account/Contact Settings
- Social Business Rules
- Social Customer Service Settings
- SoftPhone Layout
- Solution Categories
- Solution Settings

Unsupported Metadata Types
Metadata Types

• Standard fields that aren’t customizable, such as autonumber fields or system fields
• Tag Settings
• Territory Assignment Rules
• User Interface Settings (except calendar features, which are supported in ActivitiesSettings on page 750)
• Web Links on Person Account Page Layouts
• Web-to-Lead

Special Behavior in Metadata API Deployments

Important considerations for specific types and contents of a deployment.

When deploying changes to a Salesforce org, consider how individual components in your deployment behave so you’re including all the necessary changes. Use the following information to determine what to include in your deployment, and how the changes appear in the destination org.

Apex Classes and Apex Triggers

By default, changes to Apex code that has Apex jobs pending or in progress can’t be deployed. To deploy these changes, do one of the following.

• Cancel Apex jobs before deploying changes to Apex code. Reschedule the jobs after the deployment.
• Enable deployments with Apex jobs in the Salesforce user interface in the Deployment Settings page.

Approval Processes

• To use approval processes on Salesforce Knowledge articles with the Metadata API, the article type must be deployed. For article version (_kav) in approval processes, the supported action types are: Knowledge Action, Email Alert, Field Update, and Outbound Message.
• If the approval process references any post templates that contain custom fields, then you need to resave those post templates in the originating organization before adding them to the change set. From Setup, enter Post Templates in the Quick Find box, then select Post Templates. For each post template, click Edit and then Save.
• The metadata doesn’t include the order of active approval processes. You might need to reorder the approval processes in the destination org after deployment.
• If you change the Unique Name of an approval process that was previously included in a change set and deployed in another organization, and you resend the approval process via a change set, a new approval process will be created upon deployment in the other organization. The previously deployed approval process will not be modified.

Custom Fields

Starting in API version 30.0, when deploying a new custom field, the default values for the editable and readable fields in profile field permissions are false. To override the default values, include field permissions for the new field in your profiles.

Custom Objects

Using API version 29.0, you can’t change the sharingModel of an object using Metadata API. You must manually make this change to the target org through the user interface.

Starting with API version 30.0, you can change the sharingModel of an object for internal users using Metadata API and the user interface.

Connected App

• You cannot set the consumerKey in Metadata API. It is included in a retrieve operation for informational purposes. If you try to move the connected app to another org, you must remove the consumerKey from the .zip file before the deployment to an org. A new key will be generated in the destination org.
Mobile settings of connected apps are not supported in change sets and must be manually migrated.

Groups
Members of the public group are not migrated when you deploy the group type.

Master-Detail Relationships
A Metadata API deployment that includes Master-Detail relationships deletes all detail records in the Recycle Bin in the following cases.

1. For a deployment with a new Master-Detail field, soft delete (send to the Recycle Bin) all detail records before proceeding to deploy the Master-Detail field, or the deployment fails. During the deployment, detail records are permanently deleted from the Recycle Bin and cannot be recovered.

2. For a deployment that converts a Lookup field relationship to a Master-Detail relationship, detail records must reference a master record or be soft-deleted (sent to the Recycle Bin) for the deployment to succeed. However, a successful deployment permanently deletes any detail records in the Recycle Bin.

Page Layout
A deployment containing page layout assignments replaces all existing page layout assignments in the destination org with those specified in the .zip file. Existing page layouts in the org disappear if they’re not included in the .zip file. Always include all page layouts for all required record types in the .zip file.

Picklist Values
Values for a picklist field in a target org that are not included in the metadata are set to inactive.

For example, if the target org has a picklist that includes an active value of 1, and the metadata does not include 1 as a value for the picklist, 1 changes from active to inactive in the target org.

Profiles
If a package includes a profile with a name that doesn’t exist in the target org, a new profile is created with that name. If the deployed profile doesn’t specify any permissions or settings, the resulting profile consists of all the permissions and settings in the Standard Profile.

Sharing
- Simultaneously updating the sharingModel field for an object and adding a new sharing rule isn’t supported in Metadata API, regardless of which object you’re updating. For example, you can add a sharing rule when the org-wide default is public, and subsequently update the sharingModel. This would result in a single sharing recalculation.
- You might encounter an error if you’re deploying a change set with a custom object that has a parent-child relationship without the master/detail field in the same change set. To resolve this error, include the master/detail custom field in the change set, even if you haven’t changed the org-wide default.

Workflow
Test mode for flow triggers isn’t supported in the Metadata API. If you want a flow trigger to run the latest flow version when an administrator causes the workflow rule to fire, enable test mode via the user interface after deployment.

AccountRelationshipShareRule

Represents the rule that determines which object records are shared, how they are shared, the account relationship type that shares the records, and the level of access granted to the records. This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.
File Suffix and Directory Location

AccountRelationshipShareRule components have the suffix `.accountRelationshipShareRule` and are stored in the `.accountRelationshipShareRules` folder.

Version

AccountRelationshipShareRule components are available in API version and later.

Special Access Rules

Access to the Accountrelationshipsharerule type requires orgs to enable the Account Relationships permission. The Manage Communities permission is required for user access.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessLevel</td>
<td>string</td>
<td>Type of access granted by the share rule. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td>accountToCriteriaField</td>
<td>string</td>
<td>Criteria that must be met for the data to be shared. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Account.Ownerld</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Account.ParentId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Campaign.OwnerId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Case.AccountId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Case.OwnerId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Contact.AccountId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Contact.OwnerId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Lead ConvertedAccountId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Lead.OwnerId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Lead.PartnerAccountId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Opportunity.AccountId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Opportunity.OwnerId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Opportunity.PartnerAccountId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Order.AccountId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Order.OwnerId</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To get the full list for your org, do a Describe on the ARSR entity.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A meaningful explanation of the sharing rule.</td>
</tr>
<tr>
<td>entityType</td>
<td>string</td>
<td>The type of data being shared by this share rule. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Account</td>
</tr>
</tbody>
</table>
### Declarative Metadata Sample Definition

The following is an example of an `AccountRelationshipShareRule` component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<AccountRelationshipShareRule xmlns="http://soap.sforce.com/2006/04/metadata">
  <accessLevel>Edit</accessLevel>
  <accountToCriteriaField>Account.OwnerId</accountToCriteriaField>
  <description>TestDescription</description>
  <entityType>Account</entityType>
  <masterLabel>TestName</masterLabel>
  <staticFormulaCriteria>YearStarted = "1980"</staticFormulaCriteria>
  <type>Dealer</type>
</AccountRelationshipShareRule>
```

The following is an example `package.xml` that references the previous definition.

```xml
<Package>
  <?xml version="1.0" encoding="UTF-8"?>
  <Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
      <members>ArsrDevName</members>
      <name>AccountRelationshipShareRule</name>
    </types>
    <version>45.0</version>
  </Package>
</Package>
```

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file.
ActionLinkGroupTemplate

Represents the action link group template. Action link templates let you reuse action link definitions and package and distribute action links. An action link is a button on a feed element. Clicking on an action link can take a user to another Web page, initiate a file download, or invoke an API call to an external server or Salesforce. Use action links to integrate Salesforce and third-party services into the feed. Every action link belongs to an action link group and action links within the group are mutually exclusive. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

ActionLinkGroupTemplate components have the suffix .actionLinkGroupTemplate and are stored in the actionLinkGroupTemplates folder.

Version

ActionLinkGroupTemplate components are available in API version 33.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionLinkTemplates</td>
<td>ActionLinkTemplate on page 129]</td>
<td>Action link templates that are associated with the action link group template.</td>
</tr>
</tbody>
</table>
| category            | PlatformAction GroupCategory (enumeration of type string) | Required. The location of the action link group within the feed element. Values are: 
  • Primary—The action link group is displayed in the body of the feed element. 
  • Overflow—The action link group is displayed in the overflow menu of the feed element. |
| executionsAllowed   | ActionLink ExecutionsAllowed (enumeration of type string) | Required. The number of times an action link can be executed. Values are: 
  • Once—An action link can be executed only once across all users. 
  • OncePerUser—An action link can be executed only once for each user. 
  • Unlimited—An action link can be executed an unlimited number of times by each user. If the action link’s actionType is Api or ApiAsync, you can’t use this value. |
| hoursUntilExpiration | int        | Required. The number of hours from when the action link group is created until it’s removed from associated feed elements and can no longer be executed. The maximum value is 8,760. |
| isPublished         | boolean    | Required. If true, the action link group template is published. Action link group templates shouldn’t be published until at least one action link template is associated with it. |
**ActionLinkTemplate**

ActionLinkTemplate components are used to create multiple action links that share properties.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the action link group template to use in code.</td>
</tr>
<tr>
<td>actionUrl</td>
<td>string</td>
<td>Required. The action link URL. For example, a <strong>Ui</strong> action link URL is a Web page. A <strong>Download</strong> action link URL is a link to the file to download. <strong>Ui</strong> and <strong>Download</strong> action link URLs are provided to clients. An <strong>Api</strong> or <strong>ApiAsync</strong> action link URL is a REST resource. <strong>Api</strong> and <strong>ApiAsync</strong> action link URLs aren't provided to clients. Links to Salesforce can be relative. All other links must be absolute and start with <code>https://</code>.</td>
</tr>
<tr>
<td>headers</td>
<td>string</td>
<td>Template for the HTTP headers sent when corresponding action links are invoked. This field can be used only for <strong>Api</strong> and <strong>ApiAsync</strong> action links. This field can contain context variables and binding variables in the form <code>{!Bindings.key}</code>.</td>
</tr>
<tr>
<td>isConfirmationRequired</td>
<td>boolean</td>
<td>Required. If <code>true</code>, a confirmation dialog appears before the action is executed.</td>
</tr>
<tr>
<td>isGroupDefault</td>
<td>boolean</td>
<td>Required. If <code>true</code>, action links derived from this template are the default or primary action in their action groups. There can be only one default action per action group.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>A custom label to display on the action link button. If none of the <strong>LabelKey</strong> values make sense for an action link, use a custom label. Set the <strong>LabelKey</strong> field to <strong>None</strong> and enter a label name in the <strong>Label</strong> field.</td>
</tr>
<tr>
<td>labelKey</td>
<td>string</td>
<td>Required. Key for the set of labels to display for these action link states: new, pending, success, failed. For example, the Approve set contains these labels: Approve, Pending, Approved, Failed. For a complete list of keys and labels, see Action Links Labels in the Chatter REST API Developer Guide or the Apex Developer Guide.</td>
</tr>
<tr>
<td>linkType</td>
<td>ActionLinkType</td>
<td>Required. The type of action link. One of these values:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• <strong>Api</strong>—The action link calls a synchronous API at the action URL. Salesforce sets the status to <strong>SuccessfulStatus</strong> or <strong>FailedStatus</strong> based on the HTTP status code returned by your server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>ApiAsync</strong>—The action link calls an asynchronous API at the action URL. The action remains in a <strong>PendingStatus</strong> state until a third party makes a request to <code>/connect/action-links/actionLinkId</code> to set the status to <strong>SuccessfulStatus</strong> or <strong>FailedStatus</strong> when the asynchronous operation is complete.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Download</strong>—The action link downloads a file from the action URL.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Ui</strong>—The action link takes the user to a web page at the action URL.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>method</td>
<td>ActionLink</td>
<td>Required. HTTP method for the action URL. One of these values:</td>
</tr>
<tr>
<td></td>
<td>HttpMethod</td>
<td>• HttpMethodDelete—Returns HTTP 204 on success. Response body or output class is empty.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• HttpMethodGet—Returns HTTP 200 on success.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• HttpMethodRead—Returns HTTP 200 on success. Response body or output class is empty.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• HttpMethodPatch—Returns HTTP 200 on success or HTTP 204 if the response body or output class is empty.</td>
</tr>
<tr>
<td>position</td>
<td>int</td>
<td>Required. An integer specifying the position of the action link template relative to other action links in the group. 0 is the first position.</td>
</tr>
<tr>
<td>requestBody</td>
<td>string</td>
<td>Template for the HTTP request body sent when corresponding action links are invoked. This field can be used only for Api and ApiAsync action links. This field can contain context variables and binding variables in the form {!Bindings.key}.</td>
</tr>
<tr>
<td>userAlias</td>
<td>string</td>
<td>If you selected CustomUser or CustomExcludedUser for UserVisibility, this field is the alias for the custom user. Use the alias in a template binding to specify the custom user when an action link group is created using the template.</td>
</tr>
<tr>
<td>userVisibility</td>
<td>ActionLink</td>
<td>Required. Who can see the action link. This value is set per action link, not per action link group. Values are:</td>
</tr>
<tr>
<td></td>
<td>UserVisibility</td>
<td>• Creator—Only the creator of the action link can see the action link.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Everyone—Everyone can see the action link.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• EveryoneButCreator—Everyone but the creator of the action link can see the action link.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Manager—Only the manager of the creator of the action link can see the action link.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CustomUser—Only the custom user can see the action link.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CustomExcludedUser—Everyone but the custom user can see the action link.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example of an ActionLinkGroupTemplate component.

```xml
<?xml version="1.0" encoding="UTF-8"?><ActionLinkGroupTemplate xmlns="http://soap.sforce.com/2006/04/metadata"
  <actionLinkTemplates>
    <actionUrl>/services/data/{!Bindings.word}/chatter/feed-elements</actionUrl>
    <headers>Content-Type:{!Bindings.word3}</headers>
    <isConfirmationRequired>true</isConfirmationRequired>
    <isGroupDefault>true</isGroupDefault>
    <labelKey>Add</labelKey>
    <linkType>API</linkType>
    <method>httpPost</method>
    <position>0</position>
    <requestBody>{"body":{"messageSegments":["{"type": "Text",
      "text": "{!Bindings.word1}\"},"subjectId": "{!Bindings.word2}\",
      "feedElementType": "feedItem"],"feedElementType": "feedItem"}</requestBody>
    <userAlias>customExcludedUser</userAlias>
    <userVisibility>CustomExcludedUser</userVisibility>
  </actionLinkTemplates>
  <category>Primary</category>
  <executionsAllowed>OncePerUser</executionsAllowed>
  <hoursUntilExpiration>10</hoursUntilExpiration>
  <isPublished>true</isPublished>
  <name>MyPackage</name>
</ActionLinkGroupTemplate>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>ActionLinkGroupTemplate</name>
  </types>
  <version>33.0</version>
</Package>
```

Usage

If you modify action link group templates, you overwrite the related action link templates.

If you delete a published action link group template, you delete all related action link information which includes deleting all action links that were instantiated using the template from feed items.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
AnalyticSnapshot

Represents a reporting snapshot. A reporting snapshot lets you report on historical data. Authorized users can save tabular or summary report results to fields on a custom object, then map those fields to corresponding fields on a target object. They can then schedule when to run the report to load the custom object’s fields with the report’s data. Reporting snapshots enable you to work with report data similarly to how you work with other records in Salesforce.

Declarative Metadata File Suffix and Directory Location

Lightning Platform AnalyticSnapshot components are stored in the `analyticSnapshots` directory of the corresponding package directory. The file name matches the unique name of the reporting snapshot, and the extension is `.analyticsnapshot`.

Version

Lightning Platform AnalyticSnapshot components are available in API version 16.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the reporting snapshot.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The reporting snapshot name used for API access. The name can only contain</td>
</tr>
<tr>
<td></td>
<td></td>
<td>characters, letters, and the underscore <code>_</code> character, must start with a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>letter, and cannot end with an underscore or contain two consecutive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>underscore characters. This field is inherited from the Metadata component.</td>
</tr>
<tr>
<td>groupColumn</td>
<td>string</td>
<td>A column that specifies which level to extract data from the source report.</td>
</tr>
<tr>
<td>mappings</td>
<td>AnalyticSnapshotMapping[]</td>
<td>A list of reporting snapshot mappings. For valid values, see AnalyticSnapshotMapping.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The display name of the reporting snapshot.</td>
</tr>
<tr>
<td>runningUser</td>
<td>string</td>
<td>The username of the user whose role and sharing settings are used to run</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the reporting snapshot.</td>
</tr>
<tr>
<td>sourceReport</td>
<td>string</td>
<td>Required. The report where data will be extracted from.</td>
</tr>
<tr>
<td>targetObject</td>
<td>string</td>
<td>Required. The custom object where data will be inserted into.</td>
</tr>
</tbody>
</table>

AnalyticSnapshotMapping

AnalyticSnapshotMapping defines the mapping for the reporting snapshot. Valid values are:
**Field** | **Field Type** | **Description**
--- | --- | ---
aggregateType | ReportSummaryType[] (enumeration of type string) | List that defines if and how each report field is summarized. For valid values, see ReportSummaryType.
sourceField | string | The sourceField can be one of the following:
  - The field on the sourceReport that you want to map to the targetField in the targetObject
  - A summary of a field on the sourceReport (for Summary reports only)
  - A field on the reporting snapshot, such as JobName, RunningUser, or ExecutionTime (set through the user interface)
  
  **Note:** The sourceField must correspond to the sourceType you specify.
sourceType | ReportJobSourceTypes[] (enumeration of type string) | List that defines the report format for the reporting snapshot. For valid values, see ReportJobSourceTypes.
targetField | string | A field on the targetObject into which this particular sourceField will be inserted.

**ReportJobSourceTypes**

An enumeration of type string that defines the report format for the reporting snapshot. Valid values are:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>snapshot</td>
<td>Use this option if the sourceField contains snapshot-specific information such as JobName, RunningUser, or ExecutionTime.</td>
</tr>
<tr>
<td>summary</td>
<td>Use this option if referencing a summary (Sum, Average, Minimum, Maximum) of a field from the sourceReport.</td>
</tr>
<tr>
<td>tabular</td>
<td>Use this option if referencing an available column from the sourceReport.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

Here’s a sample XML definition of a reporting snapshot.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<AnalyticSnapshot xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>my description</description>
  <groupColumn>INDUSTRY</groupColumn>
  <mappings>
    <aggregateType>Average</aggregateType>
    <sourceField>SALES</sourceField>
    <sourceType>summary</sourceType>
    <targetField> myObject __c.Name</targetField>
  </mappings>
  <mappings>
    <sourceField>ExecutionTime</sourceField>
    <sourceType>snapshot</sourceType>
  </mappings>
</AnalyticSnapshot>
```
<mappings>
  <sourceField>INDUSTRY</sourceField>
  <sourceType>tabular</sourceType>
  <targetField>testObject__c.Name</targetField>
</mappings>

<name>my snapshot</name>
<runningUser>user@salesforce.com</runningUser>
<sourceReport>myFolder/mytSummaryReport</sourceReport>
<targetObject>myObject__c</targetObject>
</AnalyticSnapshot>

**Wildcard Support in the Manifest File**

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- Report

---

### AnimationRule

Represents criteria for determining when an animation is displayed to Path users. This type extends the Metadata metadata type and inherits its fullName field.

### File Suffix and Directory Location

AnimationRule components have the suffix `animationRule` and are stored in the `animationRules` folder.

### Version

AnimationRule components are available in API version 46.0 and later.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>animationFrequency</td>
<td>picklist</td>
<td>Required. The frequency with which an animation is displayed when a user selects the designated picklist values in a path. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- always</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- often</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- sometimes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- rarely</td>
</tr>
</tbody>
</table>

---

134
Metadata Types

### Field Name | Field Type | Description
--- | --- | ---
always | string | A value of always triggers an animation every time. The values often, sometimes, and rarely trigger an animation progressively less frequently.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
developerName | string | Required. The developer name for the animation rule. |
isActive | boolean | Required. Indicates whether the animation rule is active (true) or not (false). |
masterLabel | string | Required. The label for the animation rule. |
recordTypeContext | picklist | Required. An enum to track whether this AnimationRule applies to all record types for the associated sObject, or only to a single or master record type. Valid values are All, Master, or Custom. |
recordTypeName | reference | The record type selected for the sObject in which the animation is displayed. |
sobjectType | string | The object on which the animation rule is run. |
targetField | string | Required. Name of the field used to determine when to display an animation. |
targetFieldChangeToValues | string | Required. Values used to determine when to display an animation. When a user selects a value in targetField that matches a value stored in targetFieldChangeToValues, the animation is displayed. |

### Declarative Metadata Sample Definition

The following is an example of an AnimationRule component.

```xml
<AnimationRule xmlns="http://soap.sforce.com/2006/04/metadata">
    <animationFrequency>Always</animationFrequency>
    <developerName>AnimationRule_DeveloperName</developerName>
    <isActive>true</isActive>
    <masterLabel>AnimationRule Master Label</masterLabel>
    <recordTypeContext>All</recordTypeContext>
    <recordTypeName>__MASTER__</recordTypeName>
    <sobjectType>Opportunity</sobjectType>
    <targetField>StageName</targetField>
    <targetFieldChangeToValues>Delivered, Negotiating, Closed Won</targetFieldChangeToValues>
</AnimationRule>
```

The following is an example package.xml that references the AnimationRule component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>PathAssistant</members>
        <name>Settings</name>
    </types>
</Package>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**ArticleType**

Represents the metadata associated with an article type.

All articles in Salesforce Knowledge are assigned to an article type. An article’s type determines the type of content it contains, its appearance, and which users can access it. For example, a simple FAQ article type can have two custom fields, Question and Answer, where article managers enter data when creating or updating FAQ articles. A more complex article type can have dozens of fields organized into several sections. Using layouts and templates, administrators can structure the article type in the most effective way for its particular content. User access to article types is controlled by permissions. For each article type, an administrator can grant “Create,” “Read,” “Edit,” or “Delete” permissions to users. For example, the article manager can allow internal users to read, create, and edit FAQ article types, but let partner users only read FAQs. See “Knowledge Article Types” in the Salesforce online help and “Articles” in the SOAP API Developer Guide.

Declarative Metadata File Suffix and Directory Location

An ArticleType is defined as a custom object and is stored in the objects folder. ArticleTypes have a suffix __kav (instead of __c for custom objects). ArticleType field names have a suffix of __c like other custom objects, and must be dot-qualified with the name of the article type to which they belong. This is shown in the following sample package.xml file:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
   <fullName>articlefilemetadata</fullName>
   <apiAccessLevel>Unrestricted</apiAccessLevel>
   <types>
      <members>newarticle__kav.description__c</members>
      <name>CustomField</name>
   </types>
   <types>
      <members>newarticle__kav</members>
      <name>CustomObject</name>
   </types>
</Package>
```

Version

ArticleTypes are available in API version 19.0 and later.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>articleTypeChannelDisplay</td>
<td>articleTypeChannelDisplay</td>
<td>Represents the article-type templates used to display an article in the various channels. See “Article Type Templates” in the Salesforce online help.</td>
</tr>
<tr>
<td>deploymentStatus</td>
<td>DeploymentStatus (enumeration of type string)</td>
<td>A string which represents the deployment status of a custom object or field. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• InDevelopment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Deployed</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the article type. Maximum of 1000 characters.</td>
</tr>
<tr>
<td>fields</td>
<td>CustomField[]</td>
<td>Represents one or more fields in the article type.</td>
</tr>
<tr>
<td>gender</td>
<td>Gender</td>
<td>Indicates the gender of the noun that represents the object. This is used for languages where words need different treatment depending on their gender.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Label that represents the object throughout the Salesforce user interface.</td>
</tr>
<tr>
<td>pluralLabel</td>
<td>string</td>
<td>Plural version of the label value.</td>
</tr>
<tr>
<td>startsWith</td>
<td>StartsWith (enumeration of type string)</td>
<td>Indicates whether the noun starts with a vowel, consonant, or is a special character. This is used for languages where words need different treatment depending on the first character. Valid values are listed in StartsWith.</td>
</tr>
</tbody>
</table>

**articleTypeChannelDisplay**

Determines the article-type templates that are used to display an article in its channels. Unless otherwise noted, all fields are createable, filterable, and nillable.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>articleTypeTemplates</td>
<td>articleTypeTemplates</td>
<td>Indicates which article-type template applies in the specified channel.</td>
</tr>
</tbody>
</table>

**articleTypeTemplates**

Sets the article-type template for a specific channel. If not specified, the default article-type template applies.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>string</td>
<td>Specifies the channel where the article-type template applies:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AllChannels: all the available channels.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• App: the Articles tab in Salesforce Knowledge.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• P.kb: the public knowledge base.</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
Csp | the Customer Portal. |  
Prm | the partner portal. |  
page | string | Represents the name of the custom Visualforce page used as a custom article-type template. Use this field when you select Page in the template field.

template | string | Indicates the article-type template used for the specified channel:
- **Page**: custom Visualforce page. When specifying this value, you must also set the page field with the Visualforce page name.
- **Tab**: display the sections you defined in the layout as tabs.
- **Toc**: display the sections you defined in the layout as table of content.

### Declarative Metadata Sample Definitions

A sample article type definition follows:

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
    <articleTypeChannelDisplay>
        <articleTypeTemplates>
            <channel>App</channel>
            <template>Tab</template>
        </articleTypeTemplates>
        <articleTypeTemplates>
            <channel>Prm</channel>
            <template>Tab</template>
        </articleTypeTemplates>
        <articleTypeTemplates>
            <channel>Csp</channel>
            <template>Tab</template>
        </articleTypeTemplates>
        <articleTypeTemplates>
            <channel>Pkb</channel>
            <template>Toc</template>
        </articleTypeTemplates>
    </articleTypeChannelDisplay>
    <deploymentStatus>Deployed</deploymentStatus>
    <description>Article type with custom fields</description>
    <fields>
        <fullName>description__c</fullName>
        <label>Description</label>
        <length>48</length>
        <type>Text</type>
    </fields>
</CustomObject>
```
**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- ArticleType Layout
- ArticleType CustomField

**ArticleType Layout**

Represents the metadata associated with an article type page layout. Article type layouts determine which fields users can view and edit when entering data for an article, they also determine which sections appear when users view articles.

The format of the article, for example whether layout sections display as subtabs or as a single page with links, is defined by the `article-type template`. Each article type has only one layout, but you can choose a different template for each of the article type's four channels. For more information, see “Knowledge Article Types” in the Salesforce online help and “Articles” in the SOAP API Developer Guide.

**File Suffix and Directory Location**

ArticleType layouts are stored in the `layouts` directory of the corresponding package directory. The prefix must match with the article type API name. The extension is `.layout`.

**Version**

ArticleType layouts are available in API version 19.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layoutSections</td>
<td>LayoutSection</td>
<td>The main sections of the layout containing the article fields. The order here determines the layout order.</td>
</tr>
</tbody>
</table>

**LayoutSection**

LayoutSection represents a section of an ArticleType layout.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customLabel</td>
<td>boolean</td>
<td>Indicates if this section's label is custom or standard (built-in). Custom labels can be any text, but must be translated. Standard labels have a predefined set of valid values, for example 'System Information', which are automatically translated.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label; either standard or custom, based on the <code>customLabel</code> flag.</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
layoutColumns | LayoutColumn[] | The columns of the layout, depending on the style. Salesforce Knowledge only supports one column in article type layouts.
style | LayoutSectionStyle (enumeration of type string) | The style of the layout. Salesforce Knowledge only supports the value OneColumn which displays a one column page.

### LayoutColumn
LayoutColumn represents the items in a column within a layout section.

| Field Name | Field Type | Description |
--- | --- | ---
layoutItems | LayoutItem[] | The individual items within a column (ordered from top to bottom).

### LayoutItem
LayoutItem represents the valid values that define a layout item.

| Field Name | Field Type | Description |
--- | --- | ---
field | string | The field name reference, for example MyField__c.

### Declarative Metadata Sample Definition
The following is the definition of an ArticleType page layout:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Layout xmlns="http://soap.sforce.com/2006/04/metadata">
  <layoutSections>
    <customLabel>true</customLabel>
    <label>Description</label>
    <layoutColumns>
      <layoutItems>
        <field>description__c</field>
      </layoutItems>
    </layoutColumns>
  </layoutSections>
  <style>OneColumn</style>
</Layout>
```

### ArticleType Layout Metadata Types
SEE ALSO:
- ArticleType
- ArticleType CustomField

ChannelLayout

Represents the metadata associated with an communication channel layout. Communication channel layouts allow admins to share article contents inline into communication channels (e.g., email publisher, portal publisher, social publisher etc.). Admins can create a list of fields of an article type that they want to share for each communication channels and customize its order.

File Suffix and Directory Location

Channel layout components have the suffix `.channelLayout` and are stored in the `channelLayouts` folder of the corresponding package directory. The prefix must match with the article type API name.

Version

Channel layout components are available in API version 32.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>The label</td>
</tr>
<tr>
<td>layoutItems</td>
<td>layoutItem</td>
<td>The article fields contained in the layout. The order here determines the field order.</td>
</tr>
<tr>
<td>enabledChannels</td>
<td>enabledChannel</td>
<td>The channels where this layout applies.</td>
</tr>
</tbody>
</table>

layoutItem

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>Name of the field. The format is <code>&lt;ArticleType name&gt;.&lt;Field name&gt;</code></td>
</tr>
<tr>
<td>enabledChannel</td>
<td>enum</td>
<td>Enum name of the communication channel (e.g., email, portal etc)</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example of a Channel Layout component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ChannelLayout xmlns="http://soap.sforce.com/2006/04/metadata">
  <label>layoutaaaa</label>
  <layoutItems>
    <field>ArticleTypeA.FieldA</field>
  </layoutItems>
  <layoutItems>
    <field>ArticleTypeA.FieldC</field>
  </layoutItems>
  <enabledChannels>email</enabledChannels>
  <enabledChannels>portal</enabledChannels>
</ChannelLayout>
```

The following is an example `package.xml` that references the previous definition.

ArticleType CustomField

Represents the metadata associated with an article type custom field. Use this metadata type to create, update, or delete article type custom field definitions.

This type extends the `Metadata` metadata type and inherits its `fullName` field.

Always specify the full name whenever you create or update a custom field. For example, a custom field on a custom object:

```
MyArticleType__kav.MyCustomField__c
```

Declarative Metadata File Suffix and Directory Location

Custom fields are defined as part of the article type. ArticleType field names have a suffix of `__c` like other custom objects, and must be dot-qualified with the name of the article type to which they belong. See `ArticleType` for more information.

Retrieving Custom Fields on Custom or Standard Objects

When you retrieve a custom or standard object, you return everything associated with the object. However, you can also retrieve only the custom fields for an object by explicitly naming the object and fields in `package.xml`. The following definition in `package.xml` retrieves the files `objects/MyCustomObject__c.object`, `objects/Account.object__c.object`, and `objects/MyArticleType__kav.object`, each containing one custom field definition.

```xml
<types>
  <members>MyCustomObject__c.MyCustomField__c</members>
  <members>Account.MyCustomAccountField__c</members>
  <members>MyArticleType__kav.MyOtherCustomField__c</members>
</types>
```
Version

ArticleTypes custom fields are available in API version 19.0 and later.

Fields for ArticleType

Unless otherwise noted, all fields are createable, filterable, and nillable.

![Note:](image)

If you create a knowledge validation rule, the errors always display at the top of the page, even if you add it beside the field. Therefore, write the errors descriptively so authors know how to satisfy the validation rule. For example, identify which field is causing the error. The Salesforce Classic user interface does not support field level error messages for articles.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultValue</td>
<td>string</td>
<td>If specified, represents the default value of the field.</td>
</tr>
<tr>
<td>deleteConstraint</td>
<td>DeleteConstraint</td>
<td>Provides deletion options for lookup relationships. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td><img src="image" alt="SetNull" /> This is the default. If the lookup record is deleted, the lookup field is cleared.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><img src="image" alt="Restrict" /> Prevents the record from being deleted if it's in a lookup relationship.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><img src="image" alt="Cascade" /> Deletes the lookup record as well as associated lookup fields.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For more information on lookup relationships, see “Object Relationships” in the Salesforce Help.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the field.</td>
</tr>
<tr>
<td>formula</td>
<td>string</td>
<td>If specified, represents a formula on the field.</td>
</tr>
<tr>
<td>formulaTreatBlankAs</td>
<td>TreatBlanksAs</td>
<td>Indicates how to treat blanks in a formula. Valid values are BlankAsBlank and BlankAsZero.</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td><img src="image" alt="TreatBlanksAsBlank" /> and <img src="image" alt="TreatBlanksAsZero" />.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See <code>createMetadata()</code> to see an example of this field specified for a call.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value cannot be null.</td>
</tr>
<tr>
<td>inlineHelpText</td>
<td>string</td>
<td>Represents the content of field-level help. For more information, see “Define Field-Level Help” in the Salesforce Help.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Label for the field. You cannot update the label for standard fields in Article Type such as Title, UriName, Summary, etc.</td>
</tr>
<tr>
<td>length</td>
<td>int</td>
<td>Length of the field.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>picklist</td>
<td>Picklist (Including</td>
<td>(Deprecated. Use this field in API version 37.0 and earlier only. In later versions, use <code>valueSet</code> instead.) If specified, the field is a picklist, and this field enumerates the picklist values and labels.</td>
</tr>
<tr>
<td></td>
<td>Dependent Picklist)</td>
<td></td>
</tr>
<tr>
<td>referenceTo</td>
<td>string</td>
<td>If specified, indicates a reference this field has to another object.</td>
</tr>
<tr>
<td>relationshipLabel</td>
<td>string</td>
<td>Label for the relationship.</td>
</tr>
<tr>
<td>relationshipName</td>
<td>string</td>
<td>If specified, indicates the value for one-to-many relationships. For example, in the object MyObject that had a relationship to YourObject, the relationship name might be YourObjects.</td>
</tr>
<tr>
<td>required</td>
<td>boolean</td>
<td>Indicates whether the field requires a value on creation (<code>true</code>) or not (<code>false</code>).</td>
</tr>
<tr>
<td>type</td>
<td>FieldType</td>
<td>Required. Indicates the field type for the field. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Checkbox available in version 30.0 and later</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Currency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ArticleCurrency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Date</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DateTime</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Email</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• File</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Formula</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Html</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Lookup</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Percent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Phone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Picklist</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DependentPicklist</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MultiselectPicklist</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Text</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TextArea</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LongTextArea</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• URL</td>
</tr>
<tr>
<td>visibleLines</td>
<td>int</td>
<td>Indicates the number of lines displayed for the field.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  ....
  <fields>
    <fullName>Comments__c</fullName>
    <description>add your comments about this object here</description>
    <label>Comments</label>
    <length>32000</length>
    <type>LongTextArea</type>
    <visibleLines>30</visibleLines>
  </fields>
  ....
</CustomObject>

SEE ALSO:
  ArticleType
  ArticleType Layout

ApexClass

Represents an Apex class. An Apex class is a template or blueprint from which Apex objects are created. Classes consist of other classes, user-defined methods, variables, exception types, and static initialization code. For more information, see the Lightning Platform Apex Code Developer’s Guide. This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.

Note: By default, you can’t deploy updates to an Apex class if there are one or more active jobs for that class. To deploy updates in this case, do one of the following.

- Cancel Apex jobs before deploying changes to Apex code. Reschedule the jobs after the deployment.
- Enable deployments with Apex jobs in the Salesforce user interface in the Deployment Settings page.

Supported Calls

All Metadata API calls except CRUD-Based Calls, which prevents deployment outside of proper deployment lifecycle and test-execution constraints.

Declarative Metadata File Suffix and Directory Location

The file suffix is .cls for the class file. The accompanying metadata file is named ClassName-meta.xml. Apex classes are stored in the classes folder in the corresponding package directory.

Version

Apex classes are available in API version 10.0 and later.
Fields

This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>double</td>
<td>The API version for this class. Every class has an API version specified at creation.</td>
</tr>
<tr>
<td>content</td>
<td>base64</td>
<td>The Apex class definition. Base 64-encoded binary data. Prior to making an API call, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to binary. This conversion is usually handled for you by a SOAP client. This field is inherited from the MetadataWithContent component.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The Apex class name. The name can only contain characters, letters, and the underscore (_) character, must start with a letter, and cannot end with an underscore or contain two consecutive underscore characters. This field is inherited from the Metadata component.</td>
</tr>
<tr>
<td>packageVersions</td>
<td>PackageVersion[]</td>
<td>The list of installed managed package versions that are referenced by this Apex class. For more information about managed packages, see the Lightning Platform Quick Reference for Developing Packages. For more information about package versions, see “About Package Versions” in the Salesforce online help. This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>status</td>
<td>ApexCodeUnitStatus</td>
<td>The current status of the Apex class. The following string values are valid:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• Active - The class is active.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Deleted - The class is marked for deletion. This is useful for managed packages, because it allows a class to be deleted when a managed package is updated.</td>
</tr>
</tbody>
</table>

Note: ApexCodeUnitStatus includes an Inactive option, but it is only supported for ApexTrigger; it is not supported for ApexClass.

PackageVersion

PackageVersion identifies a version of a managed package. A package version is a number that identifies the set of components uploaded in a package. The version number has the format majorNumber.minorNumber.patchNumber (for example, 2.1.3). The major and minor numbers increase to a chosen value during every major release. The patchNumber is generated and updated only for a patch release. It is available in API version 16.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>namespace</td>
<td>string</td>
<td>Required. In a packaging context, a namespace prefix is a one to 15-character alphanumeric identifier that distinguishes your package and its contents from packages of other developers on AppExchange. Namespace prefixes are case-insensitive. For example, ABC and abc are not recognized as unique. Your</td>
</tr>
</tbody>
</table>
namespace prefix must be globally unique across all Salesforce organizations. It keeps your managed package under your control exclusively. Salesforce automatically prepends your namespace prefix, followed by two underscores ("__"), to all unique component names in your Salesforce organization. A unique package component is one that requires a name that no other component has within Salesforce, such as custom objects, custom fields, custom links, s-controls, and validation rules. For more information about namespaces, see “Register a Namespace Prefix” in the Salesforce online help.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>majorNumber</td>
<td>int</td>
<td>Required. The major number of the package version. A package version number has a <code>majorNumber.minorNumber</code> format.</td>
</tr>
<tr>
<td>minorNumber</td>
<td>int</td>
<td>Required. The minor number of the package version. A package version number has a <code>majorNumber.minorNumber</code> format.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following sample creates the `MyHelloWorld.cls` class, and the corresponding `MyHelloWorld.cls-meta.xml` metadata file.

**MyHelloWorld.cls file:**

```java
public class MyHelloWorld {
    // This method updates the Hello field on a list of accounts.
    public static void addHelloWorld(Account[] accs){
        for (Account a:accs){
            if (a.Hello__c != 'World')
                a.Hello__c = 'World';
        }
    }
}
```

**MyHelloWorld.cls-meta.xml**:

```xml
<?xml version="1.0" encoding="UTF-8"?
<ApexClass xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>46.0</apiVersion>
</ApexClass>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:

- ApexTrigger
ApexComponent

Represents a Visualforce component.

For more information, see “Visualforce” in the Salesforce online help. This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.

Declarative Metadata File Suffix and Directory Location

The file suffix is .component for the page file. The accompanying metadata file is named ComponentName-meta.xml. Visualforce components are stored in the components folder in the corresponding package directory.

Version

Visualforce components are available in API version 12.0 and later.

Fields

This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>double</td>
<td>The API version for this Visualforce component. Every component has an API version specified at creation. This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>content</td>
<td>base64Binary</td>
<td>The component content. Base 64-encoded binary data. Prior to making an API call, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to binary. This conversion is usually handled for you by a SOAP client. This field is inherited from the MetadataWithContent component.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of what the component does.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The component developer name used as a unique identifier for API access. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the Metadata component.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The label for this component.</td>
</tr>
<tr>
<td>packageVersions</td>
<td>PackageVersion[]</td>
<td>The list of installed managed package versions that are referenced by this Visualforce component.</td>
</tr>
</tbody>
</table>

Note: Package components and Visualforce custom component are distinct concepts. A package is comprised of many elements, such as custom objects, Apex classes and triggers, and custom pages and components.

For more information about managed packages, see the Lightning Platform Quick Reference for Developing Packages. For more information about package versions, see the Lightning Platform Quick Reference for Installing Packages.
### Wildcard Support in the Manifest File

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](#).

**SEE ALSO:**

- ApexPage

## ApexPage

Represents a Visualforce page.

For more information, see “Visualforce” in the Salesforce online help. This type extends the `MetadataWithContent` metadata type and inherits its `content` and `fullName` fields.

### Declarative Metadata File Suffix and Directory Location

The file suffix is `.page` for the page file. The accompanying metadata file is named `PageName-meta.xml`.

Visualforce pages are stored in the `pages` folder in the corresponding package directory.

### Version

Visualforce pages are available in API version 11.0 and later.

### Fields

This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>double</td>
<td>Required. The API version for this page. Every page has an API version specified at creation. This field is available in API version 15.0 and later. If you set this field to a number lower than 15.0, it will be changed to 15.0.</td>
</tr>
<tr>
<td>content</td>
<td>base64Binary</td>
<td>The page content. Base 64-encoded binary data. Prior to making an API call, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to binary. This conversion is usually handled for you by a SOAP client. This field is inherited from the <code>MetadataWithContent</code> component.</td>
</tr>
</tbody>
</table>
### Field Name
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>A description of what the page does.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The page developer name used as a unique identifier for API access. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the Metadata component.</td>
</tr>
<tr>
<td>availableInTouch</td>
<td>boolean</td>
<td>Indicates if Visualforce tabs associated with the Visualforce page can be used in the Salesforce app. (Use of this field for Salesforce Touch is deprecated). This field is available in API version 27.0 and later. Standard object tabs that are overridden with a Visualforce page aren’t supported in the Salesforce app, even if you set this field for the page. The default page for the object is displayed instead of the Visualforce page.</td>
</tr>
<tr>
<td>confirmationTokenRequired</td>
<td>boolean</td>
<td>Indicates whether GET requests for the page require a CSRF confirmation token. This field is available in API version 28.0 and later. If you change this field’s value from false to true, links to the page require a CSRF token to be added to them, or the page will be inaccessible.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The label for this page.</td>
</tr>
<tr>
<td>packageVersions</td>
<td>PackageVersion[]</td>
<td>The list of installed managed package versions that are referenced by this Visualforce page. For more information about managed packages, see the Lightning Platform Quick Reference for Developing Packages. For more information about package versions, see “About Package Versions” in the Salesforce online help. This field is available in API version 16.0 and later.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following sample creates the MyPage.page page, and the corresponding MyPage.page-meta.xml metadata file.

**SampleApexPage.page file:**

```xml
<apex:page>
  <h1>Congratulations</h1>
  This is your new Page.
</apex:page>
```

**SampleApexPage.page-meta.xml:**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ApexPage xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>This is a sample Visualforce page.</description>
</ApexPage>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:

ApexComponent

ApexTestSuite

Represents a suite of Apex test classes to include in a test run.

File Suffix and Directory Location

ApexTestSuite components have the suffix .testSuite and are stored in the testSuites folder.

Version

ApexTestSuite components are available in API version 38.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>testClassName</td>
<td>string[]</td>
<td>A list of Apex test classes, specified by name, to include in this test suite.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

To include namespaced tests in an Apex test suite, specify each namespace individually. Local Apex tests consist of all tests in the org that don’t originate from managed packages.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ApexTestSuite xmlns="http://soap.sforce.com/2006/04/metadata">
    <testClassName>LocalTestClass</testClassName>
    <!-- LocalTestClass adds the test class named LocalTestClass. -->
    <testClassName>A*Class</testClassName>
    <!-- A*Class adds AClass, AnotherClass, AwesomeClass, and so on. -->
    <testClassName>Namespace1.NamspacedTestClass</testClassName>
    <testClassName>Namespace2.*</testClassName>
    <!-- Adds all tests in Namespace2. -->
</ApexTestSuite>
```
These syntaxes are supported in `package.xml`. If the test classes in your suites are already present in the target org, you can omit the `ApexClass` type in `package.xml`.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>ApexClass</name>
  </types>
  <types>
    <members>*</members>
    <name>ApexTestSuite</name>
  </types>
  <version>38.0</version>
</Package>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>ApexClass</name>
  </types>
  <types>
    <members>
      <Suite1/>
      <Suite2/>
    </members>
    <name>ApexTestSuite</name>
  </types>
  <version>38.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](#).

**ApexTrigger**

Represents an Apex trigger. A trigger is Apex code that executes before or after specific data manipulation language (DML) events occur, such as before object records are inserted into the database, or after records have been deleted.

For more information, see "Manage Apex Triggers" in the Salesforce online help. This type extends the `MetadataWithContent` metadata type and inherits its `content` and `fullName` fields.

**Supported Calls**

All Metadata API calls except [CRUD-Based Calls](#), which prevents deployment outside of proper deployment lifecycle and test-execution constraints.

**Declarative Metadata File Suffix and Directory Location**

The file suffix is `.trigger` for the trigger file. The accompanying metadata file is named `TriggerName-meta.xml`. 
Apex triggers are stored in the `triggers` folder in the corresponding package directory.

**Version**

Triggers are available in API version 10.0 and later.

**Fields**

This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>double</td>
<td>Required. The API version for this trigger. Every trigger has an API version specified at creation.</td>
</tr>
<tr>
<td>content</td>
<td>base64</td>
<td>The Apex trigger definition. This field is inherited from the <code>MetadataWithContent</code> component.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The Apex trigger name. The name can only contain characters, letters, and the underscore (_) character, must start with a letter, and cannot end with an underscore or contain two consecutive underscore characters. This field is inherited from the <code>Metadata</code> component.</td>
</tr>
<tr>
<td>packageVersions</td>
<td>PackageVersion[]}</td>
<td>The list of installed managed package versions that are referenced by this Apex trigger. For more information about managed packages, see the Lightning Platform Quick Reference for Developing Packages. For more information about package versions, see “About Package Versions” in the Salesforce online help. This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>status</td>
<td>ApexCodeUnitStatus (enumeration of type string)</td>
<td>Required. The current status of the Apex trigger. The following string values are valid:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Active - The trigger is active.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Inactive - The trigger is inactive, but not deleted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Deleted - The trigger is marked for deletion. This is useful for managed packages, because it allows a trigger to be deleted when a managed package is updated.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following sample creates the `MyhelloWorld.trigger` trigger, and the corresponding `MyHelloWorld.trigger-meta.xml` metadata file.

**MyHelloWorld.trigger file:**

```apex
trigger helloWorldAccountTrigger on Account (before insert) {
    Account[] accs = Trigger.new;
```
AppMenu

Represents the app menu or the Salesforce mobile navigation menu.

File Suffix and Directory Location

Each AppMenu component gets stored in a single file in the folder of the corresponding package directory. The filename uses the format Feature.appMenu.

- There's one app switcher app menu file stored in a file named AppSwitcher.appMenu.
- There's one Salesforce app menu file stored in a file named Salesforce1.appMenu.

These two files are located in the appMenus folder. The .appMenu files are different from other named components, as there's only one file for each AppMenu component. App menu files can't be created or deleted.

Version

AppMenu components are available in API version 30.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appMenuItems</td>
<td>AppMenuItem[]</td>
<td>A list of menu items in the app menu.</td>
</tr>
</tbody>
</table>

AppMenuItem

Represents a menu item in the app menu.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The API name of the item.</td>
</tr>
</tbody>
</table>
| type       | string     | The type of application represented by this item. Acceptable values for `AppSwitcher.appMenu` are:  
  - `ConnectedApp`  
  - `CustomApplication`  
  - `ServiceProvider`  
  Acceptable values for `Salesforce1.appMenu` are:  
  - `CustomApplication`  
  - `CustomTab`  
  - `StandardAppMenuItem`  
  The name for this item can be:  
  - `MyDay`  
  - `Feed`  
  - `Tasks`  
  - `Dashboards`  
  - `Search`  
  - `People` (available only when Chatter is enabled)  
  - `Groups` (available only when Chatter is enabled) |

### Declarative Metadata Sample Definition

The following is an example of an `AppSwitcher.appMenu` file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<AppMenu xmlns="http://soap.sforce.com/2006/04/metadata">
  <appMenuItems>
    <appMenuItem>
      <name>standard__Sales</name>
      <type>CustomApplication</type>
    </appMenuItem>
    <appMenuItem>
      <name>standard__Support</name>
      <type>CustomApplication</type>
    </appMenuItem>
    <appMenuItem>
      <name>CustomApp1</name>
      <type>CustomApplication</type>
    </appMenuItem>
    <appMenuItem>
      <name>CustomApp2</name>
      <type>CustomApplication</type>
    </appMenuItem>
  </appMenuItems>
</AppMenu>
```
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>AppSwitcher</members>
        <name>AppMenu</name>
    </types>
</Package>
```

The following is an example of a Salesforce1.appMenu component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<AppMenu xmlns="http://soap.sforce.com/2006/04/metadata">
    <appMenuItems>
        <appMenuItem>
            <name>StandardItem1</name>
            <type>StandardAppMenuItem</type>
        </appMenuItem>
        <appMenuItem>
            <name>StandardItem2</name>
            <type>StandardAppMenuItem</type>
        </appMenuItem>
        <appMenuItem>
            <name>StandardItem3</name>
            <type>StandardAppMenuItem</type>
        </appMenuItem>
        <appMenuItem>
            <name>CustomTab1</name>
            <type>CustomTab</type>
        </appMenuItem>
    </appMenuItems>
</AppMenu>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>Salesforce1</members>
        <name>AppMenu</name>
    </types>
</Package>
```

The following is an example of a package manifest used to deploy or retrieve all the available app menu metadata for an organization, using a wildcard:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>*</members>
        <name>AppMenu</name>
    </types>
</Package>
```
Usage

Use `AppSwitcher.appMenu` to reorder the list of menu items that appears in the app menu. You can’t add app menu items to or remove app menu items from `AppSwitcher.appMenu`.

Use `Salesforce1.appMenu` to customize the list of menu items that appears in the Salesforce navigation menu by reordering, adding, or removing the app menu items.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

ApprovalProcess

Represents the metadata associated with an approval process. An approval process automates how records are approved in Salesforce. An approval process specifies each step of approval, including who to request approval from and what to do at each point of the process. This type extends the `Metadata` metadata type and inherits its `fullName` field.

Note:

- To use approval processes on Salesforce Knowledge articles with the Metadata API, the article type must be deployed. For article version (_kav) in approval processes, the supported action types are: Knowledge Action, Email Alert, Field Update, and Outbound Message.
- Send actions and approval processes for email drafts aren’t supported in the Metadata API.
- The metadata doesn’t include the order of active approval processes. You might need to reorder the approval processes in the destination org after deployment.
- Before you implement an approval process for your organization, see “Considerations for Approvals” in the Salesforce Help.

File Suffix and Directory Location

ApprovalProcess components have the suffix `.approvalProcess` and are stored in the `approvalProcesses` folder.

Version

ApprovalProcess components are available in API version 28.0 and later.
## Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Whether the approval process is active. After an approval process is activated, you can’t add, delete, or change the order of the steps or change its reject or skip behavior, even if the process is inactive.</td>
</tr>
<tr>
<td>allowRecall</td>
<td>boolean</td>
<td>Whether to allow submitters to recall approval requests. If set to false, only administrators can recall approval requests.</td>
</tr>
<tr>
<td>allowedSubmitters</td>
<td>ApprovalSubmitter[]</td>
<td>Required. An array of users who are allowed to submit records for approval.</td>
</tr>
<tr>
<td>approvalPageFields</td>
<td>ApprovalPageField</td>
<td>Specifies which fields to display on the approval page, where the approver goes to approve or reject the record. By default, the approval page displays the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Name field</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Owner field (except for child objects)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If you enable notifications in the Salesforce app, keep in mind that approvers may view this list of fields on a mobile device. Select only the fields necessary for users to decide whether to approve or reject records.</td>
</tr>
<tr>
<td>approvalStep</td>
<td>ApprovalStep[]</td>
<td>An array of approval step definitions.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Describes the approval process.</td>
</tr>
<tr>
<td>emailTemplate</td>
<td>string</td>
<td>Specifies which Classic email template to use for approval requests. If not specified, the default email template is used. Lightning email templates aren’t packageable. We recommend using a Classic email template. When an approval process assigns an approval request to a user, Salesforce sends the user an approval request email.</td>
</tr>
<tr>
<td>enableMobileDeviceAccess</td>
<td>boolean</td>
<td>Whether users can access an external version of the approval page from any browser, including those on mobile devices, without logging in to Salesforce. Corresponds to Security Settings in the user interface. If set to true, approval steps can’t have approvers of type adhoc. If set to false, approvers must log in to Salesforce to access the approval page.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>entryCriteria</td>
<td>ApprovalEntryCriteria</td>
<td>Determines which records can enter the approval process. Exclude this field to allow all records to enter the approval process.</td>
</tr>
<tr>
<td>finalApprovalActions</td>
<td>ApprovalAction</td>
<td>Specifies which workflow actions to execute when all required approvals have been given for a record.</td>
</tr>
<tr>
<td>finalApprovalRecordLock</td>
<td>boolean</td>
<td>Whether to keep the record locked after it receives all necessary approvals. Default: false.</td>
</tr>
<tr>
<td>finalRejectionActions</td>
<td>ApprovalAction</td>
<td>Specifies which workflow actions to execute after a record enters the final rejection state.</td>
</tr>
<tr>
<td>finalRejectionRecordLock</td>
<td>boolean</td>
<td>Whether to keep the record locked after it's finally rejected. Default: false.</td>
</tr>
<tr>
<td>initialSubmissionActions</td>
<td>ApprovalAction</td>
<td>Specifies which workflow actions to execute when a record is initially submitted for approval.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Name of the approval process.</td>
</tr>
<tr>
<td>nextAutomatedApprover</td>
<td>NextAutomatedApprover</td>
<td>Specifies a standard or custom user hierarchy field that can be used to automatically assign the approver for an approval step. If you exclude this field, then no approval step can use a user hierarchy field to automatically assign the approver.</td>
</tr>
<tr>
<td>postTemplate</td>
<td>string</td>
<td>Post template to use for Approvals in Chatter. Chatter post approval notifications are only available for approval processes associated with an object that has been enabled for feed tracking.</td>
</tr>
<tr>
<td>recallActions</td>
<td>ApprovalAction</td>
<td>Specifies which workflow actions to execute when a pending approval request is withdrawn.</td>
</tr>
</tbody>
</table>
| recordEditability       | RecordEditabilityType (enumeration of type string) | Specifies which users can edit records that are pending approval. When a record is submitted for approval, it is automatically locked to prevent other users from editing it during the approval process. Valid values are:  
  - AdminOnly—Records pending approval can be edited by:  
    - Users with the “Modify All Data” permission  
    - Users with the “Modify All” object-level permission for the given object  
  - AdminOrCurrentApprover—Records pending approval can be edited by:  
    - Users with the “Modify All Data” permission  
    - Users with the “Modify All” object-level permission for the given object |
The assigned approver, who must have edit access to the record through user permissions and the organization-wide sharing defaults for the given object.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showApprovalHistory</td>
<td>boolean</td>
<td>Whether to add the Approval History related list to the approval page, which is where the approver can view the approval request details and approve or reject the record. The Approval History related list tracks a record through the approval process. If you also want to add the Approval History related list to record detail and edit pages, use the Salesforce user interface to customize the page layouts for the given object.</td>
</tr>
</tbody>
</table>

**ApprovalSubmitter**

Represents a user or set of users who can submit records for approval.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| submitter  | string     | Identifies a specific user or set of users who can submit records for approval. This field is required, except when the following types are specified and the submitter field is ignored:  
- owner  
- creator  
- allInternalUsers  

Example:

```xml
<allowedSubmitters>  
  <type>allInternalUsers</type>  
</allowedSubmitters>  
<allowedSubmitters>  
  <submitter>myGroup</submitter>  
  <type>group</type>  
</allowedSubmitters>
```

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| type       | ProcessSubmitterType (enumeration of type string) | Required. Type of user or set of users who can submit records for approval. Valid values are:  
- group  
- role  
- user  
- roleSubordinates  
- roleSubordinatesInternal  
- owner  
- creator |
### ApprovalPageField

Represents the selection of fields to display on the approval page, where an approver can view the approval request details and approve or reject the record.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string[]</td>
<td>An array of fields that are displayed on the page for the approver to approve or reject the record.</td>
</tr>
</tbody>
</table>

### ApprovalStep

Represents a step in the approval process. Approval steps define the chain of approval for a particular approval process. Each step determines which records can advance to that step, who to assign approval requests to, and whether to let each approver’s delegate respond to the requests. The first step specifies what to do if a record doesn’t advance to that step. Later steps specify what happens if an approver rejects the request.

**Note:**
- The order of the ApprovalStep entries in the approval process definition determines the order in which the approval steps are executed.
- After an approval process is activated, you can’t add, delete, or change the order of the steps or change its reject or skip behavior, even if the process is inactive.
- Each approval process supports up to 30 steps.
### ApprovalProcess

Represents the actions that occur as a result of an approval process.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>WorkflowActionReference[]</td>
<td>An array of workflow actions to execute.</td>
</tr>
</tbody>
</table>

### ApprovalStepApprover

Represents the assigned approvers for an approval step. Each step supports up to 25 approvers.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>approver</td>
<td>Approver[]</td>
<td>An array of assigned approvers for this step of the approval process.</td>
</tr>
</tbody>
</table>
**whenMultipleApprovers**

Specifies how to handle approval or rejection when multiple approvers are assigned to the step. Valid values are:

- **Unanimous**—(Default) Require unanimous approval from all approvers for this step. The approval request for this step is rejected if any of the approvers reject the request.
- **FirstResponse**—Approve or reject based on the first response.

**Approver**

Represents an assigned approver for an approval step. Check out “Considerations for Setting Approvers” in the Salesforce Help.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Identifies an assigned approver. This field is required, except when the type is one of the following and the name is ignored:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• adhoc</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• userHierarchyField</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>type</th>
<th>NextOwnerType</th>
<th>Combined with the specified name, this identifies an assigned approver. Valid values are:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• adhoc—The approver for the step must be selected manually. For the first step, the submitter selects the approver. For the second and later steps, the approver for the previous step selects the approver. For this value, exclude the name field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• user—A user in your organization. For this value, enter a username for the name field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• userHierarchyField—A user specified in a standard or custom user hierarchy field, such as the standard Manager field. For this value, exclude the name field. The user hierarchy field must be defined in the nextAutomatedApprovers for the approval process.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• relatedUserField—A user specified in a user lookup field on the submitted record, such as the Last Modified By field. For this value, enter the name of the user lookup field for the name field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• queue—Automatically assign to a queue. For this value, enter the name of the queue for the name field.</td>
</tr>
</tbody>
</table>

**ApprovalEntryCriteria**

Represents the criteria that records must meet to enter the approval process or an approval step. Specify either filter criteria or a formula, but not both.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Filter logic for criteriaItems. Exclude this field if you enter a formula.</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteriaItems</td>
<td>FilterItem[]</td>
<td>Filter criteria that a record must meet to enter the approval process or approval step. Approval processes don’t support valueField entries in filter criteria.</td>
</tr>
<tr>
<td>formula</td>
<td>string</td>
<td>Formula that must evaluate to true for a record to enter the approval process or approval step.</td>
</tr>
</tbody>
</table>

### ApprovalStepRejectBehavior

Represents what happens if the approver rejects the request during this approval step, unless it’s the first step in the approval process. For the first step in the approval process, the reject behavior is determined by the approval process’s final rejection actions.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>StepRejectBehaviorType (enumeration of type string)</td>
<td>Not allowed in the first step of the approval process. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RejectRequest—Rejects the request even if previous steps were approved. Salesforce performs all rejection actions specified for this step and all final rejection actions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BackToPrevious—Rejects the request, and returns the approval request to the previous approver. Salesforce performs all rejection actions specified for this step.</td>
</tr>
</tbody>
</table>

### NextAutomatedApprover

Represents the user hierarchy field to use as the next automated approver for the approval process. If defined, the user specified in the hierarchy field can be automatically assigned as the approver in one or more approval steps.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>useApproverFieldOfRecordOwner</td>
<td>boolean</td>
<td>Required. Whether the first executed approval step should use the specified userHierarchyField in the record owner’s user record—instead of the submittler’s user record—as the approver. All remaining steps use the specified userHierarchyField in the user record of the preceding step’s approver.</td>
</tr>
<tr>
<td>userHierarchyField</td>
<td>string</td>
<td>Required. Standard or custom user hierarchy field whose value specifies which user to assign as the approver. For example, the standard Manager hierarchy field can be used to assign approvers for employee PTO (paid time off) requests.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example of an ApprovalProcess component:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ApprovalProcess xmlns="http://soap.sforce.com/2006/04/metadata">
  <active>false</active>
  <allowRecall>false</allowRecall>
  <allowedSubmitters>
    <type>owner</type>
  </allowedSubmitters>
  <allowedSubmitters>
    <submitter>USSalesRep</submitter>
    <type>role</type>
  </allowedSubmitters>
  <allowedSubmitters>
    <submitter>MarketingGroup</submitter>
    <type>group</type>
  </allowedSubmitters>
  <allowedSubmitters>
    <submitter>kcooper@example.com</submitter>
    <type>user</type>
  </allowedSubmitters>
  <approvalPageFields>
    <field>Name</field>
    <field>Owner</field>
    <field>MyLeadCustomField__c</field>
    <field>Address</field>
  </approvalPageFields>
  <approvalStep>
    <allowDelegate>false</allowDelegate>
    <approvalActions>
      <action>
        <name>LeadApprovedTask1</name>
        <type>Task</type>
      </action>
      <action>
        <name>LeadApprovedTask2</name>
        <type>Task</type>
      </action>
    </approvalActions>
    <assignedApprover>
      <approver>
        <type>adhoc</type>
      </approver>
    </assignedApprover>
    <label>Step1</label>
    <name>Step1</name>
    <rejectionActions>
      <action>
        <name>LeadRejectedTask</name>
        <type>Task</type>
      </action>
    </rejectionActions>
  </approvalStep>
</ApprovalProcess>
```
<approvalStep>
  <allowDelegate>false</allowDelegate>
  <assignedApprover>
    <approver>
      <type>userHierarchyField</type>
    </approver>
  </assignedApprover>
  <entryCriteria>
    <criteriaItems>
      <field>Lead.CreatedDate</field>
      <operation>equals</operation>
      <value>3/25/2013</value>
    </criteriaItems>
    <criteriaItems>
      <field>User.IsActive</field>
      <operation>equals</operation>
      <value>true</value>
    </criteriaItems>
  </entryCriteria>
  <ifCriteriaNotMet>ApproveRecord</ifCriteriaNotMet>
  <label>Step2</label>
  <name>Step2</name>
  <rejectBehavior>
    <type>RejectRequest</type>
  </rejectBehavior>
</approvalStep>

<approvalStep>
  <allowDelegate>true</allowDelegate>
  <assignedApprover>
    <approver>
      <name>MarketingTeamQueue</name>
      <type>queue</type>
    </approver>
    <approver>
      <name>LastModifiedBy</name>
      <type>relatedUserField</type>
    </approver>
    <approver>
      <name>awheeler@example.com</name>
      <type>user</type>
    </approver>
    <whenMultipleApprovers>FirstResponse</whenMultipleApprovers>
  </assignedApprover>
  <entryCriteria>
    <formula>CONTAINS( MyLeadCustomField__c , 'Salesforce')</formula>
  </entryCriteria>
  <label>Step3</label>
  <name>Step3</name>
  <rejectBehavior>
    <type>BackToPrevious</type>
  </rejectBehavior>
</approvalStep>

<emailTemplate>MyFolder/LeadsNewassignmentnotification</emailTemplate>
<enableMobileDeviceAccess>false</enableMobileDeviceAccess>
<entryCriteria>
  <criteriaItems>
    <field>Lead.AnnualRevenue</field>
    <operation>greaterThan</operation>
    <value>10500</value>
  </criteriaItems>
  <criteriaItems>
    <field>Lead.MyLeadCustomField__c</field>
    <operation>equals</operation>
    <value>Salesforce</value>
  </criteriaItems>
</entryCriteria>

<finalApprovalActions>
  <action>
    <name>LeadEmailContacted</name>
    <type>Alert</type>
  </action>
</finalApprovalActions>

<finalApprovalRecordLock>true</finalApprovalRecordLock>

<finalRejectionActions>
  <action>
    <name>ProcessRejectedMessageAction</name>
    <type>OutboundMessage</type>
  </action>
</finalRejectionActions>

<finalRejectionRecordLock>false</finalRejectionRecordLock>

<initialSubmissionActions>
  <action>
    <name>LeadFieldUpdate</name>
    <type>FieldUpdate</type>
  </action>
  <action>
    <name>NewLeadEmail</name>
    <type>Alert</type>
  </action>
</initialSubmissionActions>

<label>SampleProcess</label>

<nextAutomatedApprover>
  <useApproverFieldOfRecordOwner>false</useApproverFieldOfRecordOwner>
  <userHierarchyField>customlookupuserfield__c</userHierarchyField>
</nextAutomatedApprover>

<postTemplate>MyPostTemplate</postTemplate>

<recallActions>
  <action>
    <name>ProcessRecalledMessageAction</name>
    <type>OutboundMessage</type>
  </action>
</recallActions>

<recordEditability>AdminOnly</recordEditability>

<showApprovalHistory>false</showApprovalHistory>

</ApprovalProcess>
**Wildcard Support in the Manifest File**

Use the wildcard character * (asterisk) in the `package.xml` manifest file to retrieve all approval processes for all objects. You can’t use it to retrieve a subset of approval processes. Syntax such as `Lead.*` is not supported. For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](#).

**AssignmentRules**

Represents assignment rules that allow you to automatically route cases to the appropriate users or queues. You can access rules metadata for all applicable objects, for a specific object, or for a specific rule on a specific object.

The `package.xml` syntax for accessing all assignment rules for all objects is:

```xml
<types>
  <members>*</members>
  <name>AssignmentRules</name>
</types>
```

All rules for a specific object uses a similar syntax without the wildcard. For example, all assignment rules for the Case object would use this syntax:

```xml
<types>
  <members>Case</members>
  <name>AssignmentRules</name>
</types>
```

You can also access specific assignment rules for an object. The following example only accesses the “samplerule” and “newrule” assignment rules on the Case object. Notice that for this example the type name syntax is `AssignmentRule` and not `AssignmentRules`.

```xml
<types>
  <members>Case.samplerule</members>
  <members>Case.newrule</members>
  <name>AssignmentRule</name>
</types>
```

**File Suffix and Directory Location**

Assignment rules for an object have the suffix `.assignmentRules` and are stored in the `assignmentRules` folder. For example, all Case assignment rules are stored in the `Case.assignmentRules` file.

**Version**

AssignmentRules components are available in API version 27.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignmentRule</td>
<td>AssignmentRule[]</td>
<td>Represents the definitions of the named assignment rules.</td>
</tr>
</tbody>
</table>
**AssignmentRule**

Specifies whether the rule is active or not and its definition. Rules are processed in the order they appear within the AssignmentRules container.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Indicates whether the assignment rule is active (true) or not (false).</td>
</tr>
<tr>
<td>fullname</td>
<td>string</td>
<td>Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See createMetadata() to see an example of this field specified for a call. This value cannot be null.</td>
</tr>
<tr>
<td>ruleEntry</td>
<td>RuleEntry[]</td>
<td>Represents the type and description for the assignment rule.</td>
</tr>
</tbody>
</table>

**RuleEntry**

Represents the fields used by the rule.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignedTo</td>
<td>string</td>
<td>The name of the user or queue the item is assigned to.</td>
</tr>
<tr>
<td>assignedToType</td>
<td>AssignToLookupValueTypetreferencetypes</td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• User</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Queue</td>
</tr>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Advanced filter conditions that were specified for the rule.</td>
</tr>
<tr>
<td>criteriaItems</td>
<td>FilterItem[]</td>
<td>The items in the list that define the assignment criteria.</td>
</tr>
<tr>
<td>formula</td>
<td>string</td>
<td>The validation formula.</td>
</tr>
<tr>
<td>notifyCcRecipients</td>
<td>boolean</td>
<td>Specifies whether email addresses included on the Cc line of an incoming Email-to-Case or Web-to-Lead message should be included on the Cc line of the auto-response to that message (true) or not (false). Available in API version 32.0 and later.</td>
</tr>
<tr>
<td>overrideExistingTeams</td>
<td>boolean</td>
<td>Specifies whether the case team should be reset when the assignment is done (true) or if the current team is added to the case instead of replacing the previous team (false).</td>
</tr>
</tbody>
</table>
**Metadata Types**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>team</td>
<td>string[]</td>
<td>The name of the case team. It may occur 0 or more times.</td>
</tr>
<tr>
<td>template</td>
<td>string</td>
<td>Specifies the template to use for the email that is automatically sent to the designated recipient. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example file showing two assignment rules on the Case object:

```xml
  <assignmentRule>
    <fullName>samplerule</fullName>
    <active>false</active>
    <ruleEntry>
      <assignedTo>testUser@org.com</assignedTo>
      <assignedToType>User</assignedToType>
      <criteriaItems>
        <field>Case.IsEscalated</field>
        <operation>equals</operation>
        <value>True</value>
      </criteriaItems>
      <template>emailtemplate</template>
    </ruleEntry>
  </assignmentRule>
  <assignmentRule>
    <fullName>Another samplerule</fullName>
    <active>false</active>
    <ruleEntry>
      <assignedTo>otherUser@org.com</assignedTo>
      <assignedToType>User</assignedToType>
      <criteriaItems>
        <field>Case.IsEscalated</field>
        <operation>equals</operation>
        <value>False</value>
      </criteriaItems>
      <template>emailtemplate</template>
    </ruleEntry>
  </assignmentRule>
</AssignmentRules>

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
Audience

Represents the audience in a community. An audience consists of different types of criteria, where the audience can be assigned and used for targeting in a community. This type extends the Metadata metadata type and inherits its `fullName` field.

File Suffix and Directory Location

Audience components have the suffix `.audience` and are stored in the `audience` folder.

Version

Audience components are available in API version 44.0 and later.

Special Access Rules

Access to the Audience type requires the AudienceMetadata permission. This permission is on by default for orgs that have Networks enabled.

Access to permission-based criteria for the Audience type requires the AudiencePermissionCriteria permission. This permission is available in API version 45.0 and later and is on by default for orgs that have Networks enabled.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>audienceName</td>
<td>string</td>
<td>Required. The name of the audience.</td>
</tr>
<tr>
<td>container</td>
<td>string</td>
<td>Required. The name of the community or org that contains the audience.</td>
</tr>
<tr>
<td>criterion</td>
<td>AudienceCriterion[]</td>
<td>List of criteria in an audience.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the audience.</td>
</tr>
<tr>
<td>formula</td>
<td>string</td>
<td>Formula used to determine the audience. This field is available in API version 45.0 and later.</td>
</tr>
</tbody>
</table>
| formulaFilterType | FormulaFilterType (enumeration of type string) | Indicates the audience’s formula type. Valid values are:
  - `AllCriteriaMatch`
  - `AnyCriterionMatches`
  - `CustomLogicMatches` (available in API version 45.0 and later) |

AudienceCriterion

Represents a criterion for an audience.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteriaNumber</td>
<td>int</td>
<td>The number associated with the criterion in a formula, for example, (1 AND 2) OR 3. This field is available in API version 45.0 and later.</td>
</tr>
<tr>
<td>criterionValue</td>
<td>AudienceCriteriaValue</td>
<td>The value of the criterion.</td>
</tr>
</tbody>
</table>
| operator         | AudienceCriterionOperator(enumeration of type string) | The operator associated with this criterion. Valid values are:  
  - Equal  
  - NotEqual  
  - GreaterThan  
  - GreaterThanOrEqual  
  - LessThan  
  - LessThanOrEqual  
  - Contains  
  - StartsWith  
  - Includes (available in API version 45.0 and later)  
  - NotIncludes (available in API version 45.0 and later) |
| type             | AudienceCriterionType(enumeration of type string) | Valid values are:  
  - GeoLocation  
  - Domain  
  - Profile  
  - FieldBased  
  - Permission (available in API version 45.0 and later)  
For a list of AudienceCriteriaValue fields that you can use with each AudienceCriterion type field value, see this table. |

### AudienceCriteriaValue

Represents the value of a criterion in an audience. For a list of AudienceCriteriaValue fields that you can use with each AudienceCriterion type field value, see this table.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>city</td>
<td>string</td>
<td>City of a user. This field can be used only when the value of the AudienceCriterion type field is GeoLocation.</td>
</tr>
<tr>
<td>country</td>
<td>string</td>
<td>Country of a user. This field can be used only when the value of the AudienceCriterion type field is GeoLocation.</td>
</tr>
<tr>
<td>domain</td>
<td>string</td>
<td>Domain of a user. This field can be used only when the value of the AudienceCriterion type field is Domain.</td>
</tr>
<tr>
<td>entityField</td>
<td>string</td>
<td>Field of an object. This field can be used only when the value of the AudienceCriterion type field is FieldBased.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>entityType</td>
<td>string</td>
<td>Type of object. This field can be used only when the value of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AudienceCriterion type field is FieldBased.</td>
</tr>
<tr>
<td>fieldValue</td>
<td>string</td>
<td>Value of a field. This field can be used only when the value of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AudienceCriterion type field is FieldBased.</td>
</tr>
<tr>
<td>isEnabled</td>
<td>string</td>
<td>Indicates whether the permission is enabled (true) or not (false) for a user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 45.0 and later and can be used only when the value of the AudienceCriterion type field is Permission.</td>
</tr>
<tr>
<td>permissionName</td>
<td>string</td>
<td>Valid API name of a standard user or custom permission. This field is available in API version 45.0 and later and can be used only when the value of the AudienceCriterion type field is Permission.</td>
</tr>
<tr>
<td>permissionType</td>
<td>string</td>
<td>Type of permission. Valid values are Standard and Custom. This field is available in API version 45.0 and later and can be used only when the value of the AudienceCriterion type field is Permission.</td>
</tr>
<tr>
<td>profile</td>
<td>string</td>
<td>Profile of a user. This field can be used only when the value of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AudienceCriterion type field is Profile.</td>
</tr>
<tr>
<td>subdivision</td>
<td>string</td>
<td>Subdivision of a user. This field can be used only when the value of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AudienceCriterion type field is GeoLocation.</td>
</tr>
</tbody>
</table>

This table summarizes which AudienceCriteriaValue fields you can use with the different AudienceCriterion type field values.

<table>
<thead>
<tr>
<th>AudienceCriterion Type</th>
<th>AudienceCriteriaValue Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>GeoLocation</td>
<td>city</td>
</tr>
<tr>
<td></td>
<td>country</td>
</tr>
<tr>
<td></td>
<td>subdivision</td>
</tr>
<tr>
<td>Domain</td>
<td></td>
</tr>
<tr>
<td></td>
<td>domain</td>
</tr>
<tr>
<td>Profile</td>
<td></td>
</tr>
<tr>
<td></td>
<td>profile</td>
</tr>
<tr>
<td>FieldBased</td>
<td></td>
</tr>
<tr>
<td></td>
<td>entityField</td>
</tr>
<tr>
<td></td>
<td>entityType</td>
</tr>
<tr>
<td></td>
<td>fieldValue</td>
</tr>
<tr>
<td>Permission</td>
<td></td>
</tr>
<tr>
<td></td>
<td>isEnabled</td>
</tr>
<tr>
<td></td>
<td>permissionName</td>
</tr>
<tr>
<td></td>
<td>permissionType</td>
</tr>
</tbody>
</table>

173
Declarative Metadata Sample Definition

The following is an example of an Audience component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Audience xmlns="http://soap.sforce.com/2006/04/metadata">
  <audienceName>Audience Metadata</audienceName>
  <container>Customer</container>
  <criterion>
    <criteriaNumber>1</criteriaNumber>
    <criterionValue>
      <country>United States</country>
      <subdivision>Nevada</subdivision>
    </criterionValue>
    <operator>Equal</operator>
    <type>GeoLocation</type>
  </criterion>
  <criterion>
    <criteriaNumber>2</criteriaNumber>
    <criterionValue>
      <profile>customer community user</profile>
    </criterionValue>
    <operator>Equal</operator>
    <type>Profile</type>
  </criterion>
  <criterion>
    <criteriaNumber>3</criteriaNumber>
    <criterionValue>
      <domain>sampledomain.example.com</domain>
    </criterionValue>
    <operator>Equal</operator>
    <type>Domain</type>
  </criterion>
  <criterion>
    <criteriaNumber>4</criteriaNumber>
    <criterionValue>
      <entityField>Manager.Profile.CreatedBy.Contact.MailingCountry</entityField>
      <entityType>User</entityType>
      <fieldValue>USA</fieldValue>
    </criterionValue>
    <operator>StartsWith</operator>
    <type>FieldBased</type>
  </criterion>
  <criterion>
    <criteriaNumber>5</criteriaNumber>
    <criterionValue>
      <entityField>RecordTypeId</entityField>
      <entityType>CollaborationGroup</entityType>
      <fieldValue>CollaborationGroup.Group_RT2</fieldValue>
    </criterionValue>
    <operator>Equal</operator>
    <type>FieldBased</type>
  </criterion>
  <criterion>
    <criteriaNumber>6</criteriaNumber>
  </criterion>
</Audience>
```
Usage

You can't delete an audience using Metadata API.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

AuraDefinitionBundle

Represents an Aura definition bundle. A bundle contains an Aura definition, such as an Aura component, and its related resources, such as a JavaScript controller. The definition can be a component, application, event, interface, or a tokens collection.

File Suffix and Directory Location

An AuraDefinitionBundle component is a folder that contains definition files. Unlike most other metadata components, an AuraDefinitionBundle component isn’t a single file, it's a collection of files. Each file represents a resource in a bundle, such as markup, applications, code files (including controllers and helpers), events, documentation, and interfaces. For example, this directory structure shows the hierarchy of the folders and files for two bundles: bundle1 and bundle2.

```
aura
  bundle1
    bundle1.cmp
    bundle1Controller.js
  bundle2
    bundle2.app
```
Aura definition bundles must be under a top-level folder named `aura`. Each bundle must have its own subfolder under the `aura` folder. The name of each definition file must start with the bundle name.

A bundle doesn’t have a suffix. Definition files can have one of these suffixes:

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Component Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>.app</td>
<td>Application</td>
</tr>
<tr>
<td>.cmp</td>
<td>Component</td>
</tr>
<tr>
<td>.design</td>
<td>Design</td>
</tr>
<tr>
<td>.evt</td>
<td>Event</td>
</tr>
<tr>
<td>.intf</td>
<td>Interface</td>
</tr>
<tr>
<td>.js</td>
<td>Controller, Helper, or Renderer</td>
</tr>
<tr>
<td>.svg</td>
<td>SVG image</td>
</tr>
<tr>
<td>.css</td>
<td>Style</td>
</tr>
<tr>
<td>.auradoc</td>
<td>Documentation</td>
</tr>
<tr>
<td>.tokens</td>
<td>Tokens collection</td>
</tr>
</tbody>
</table>

Each bundle can have only one file each with a suffix of `.app`, `.cmp`, `.design`, `.evt`, `.intf`, or `.tokens`.

**Version**

AuraDefinitionBundle components are available in API version 32.0 and later.

Design and SVG components are available in API version 33.0 and later.

In API version 45.0 and later, there are two types of Lightning component: Aura components and Lightning web components. This metadata type describes an Aura component.

**Special Access Rules**

Definitions can be created only in organizations with defined namespaces.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>double</td>
<td>The API version for this definition bundle. When you create an Aura bundle, you can specify the API version to save it with. Available in API version 35.0 and later.</td>
</tr>
<tr>
<td>controllerContent</td>
<td>base64Binary</td>
<td>The content of a JavaScript client-side controller.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The specification of the Aura bundle. Available in API version 35.0 and later.</td>
</tr>
<tr>
<td>designContent</td>
<td>base64Binary</td>
<td>The content of a design definition. Only valid inside a component bundle.</td>
</tr>
<tr>
<td>documentationContent</td>
<td>base64Binary</td>
<td>The content of a documentation definition.</td>
</tr>
<tr>
<td>helperContent</td>
<td>base64Binary</td>
<td>The content of a JavaScript helper.</td>
</tr>
<tr>
<td>markup</td>
<td>base64Binary</td>
<td>The content of the markup for a definition.</td>
</tr>
<tr>
<td>modelContent</td>
<td>base64Binary</td>
<td>Deprecated. Do not use.</td>
</tr>
<tr>
<td>packageVersions</td>
<td>PackageVersion[]</td>
<td>The list of installed managed package versions that this Aura definition bundle references. Available in API version 35.0 and later.</td>
</tr>
<tr>
<td>rendererContent</td>
<td>base64Binary</td>
<td>The content of a JavaScript client-side renderer.</td>
</tr>
<tr>
<td>styleContent</td>
<td>base64Binary</td>
<td>The CSS for the definition.</td>
</tr>
<tr>
<td>SVGContent</td>
<td>base64Binary</td>
<td>The SVG image for the definition.</td>
</tr>
<tr>
<td>testsuiteContent</td>
<td>base64Binary</td>
<td>Reserved for future use.</td>
</tr>
</tbody>
</table>
| type             | AuraBundleType (enumeration of type string) | The definition type. Valid values are:  
• Application
• Component
• Event
• Interface
• Tokens

**Declarative Metadata Sample Definition**

This example shows the directory structure of an AuraDefinitionBundle component.

```xml
<aura:component>
  <aura:attribute name="val1" type="String" default="Value"/>
  <aura:attribute name="val2" type="String" />
  <aura:handler name="init" value="{!this}" action="{!c.myAction}"/>
  <ui:outputText value='Hello world!'/>
  <ui:outputText value='{{v.val1}}'/>
  <ui:outputText value='{{v.val2}}'/>
</aura:component>
```

The following samples show the contents of the metadata definition files that correspond to the sample `aura` directory.

Content of `sampleCmp.cmp`:

```xml
<aura:component>
  <aura:attribute name="val1" type="String" default="Value"/>
  <aura:attribute name="val2" type="String" />
  <aura:handler name="init" value="{!this}" action="{!c.myAction}"/>
  <ui:outputText value='Hello world!'/>
  <ui:outputText value='{{v.val1}}'/>
  <ui:outputText value='{{v.val2}}'/>
</aura:component>
```
Content of `sampleCmpController.js`:

```javascript
({
  myAction : function(component) {
    component.set('v.val1','Value1');
    component.set('v.val2','Value2');
  }
})
```

This `package.xml` references the definitions of all Lightning components that are present in the `sampleCmp` bundle.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>sampleCmp</members>
    <name>AuraDefinitionBundle</name>
  </types>
  <version>46.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**AuthProvider**

Represents an authentication provider (or auth provider) in your organization. An auth provider enables users to log in to your Salesforce organization using their login credentials from an external service provider such as Facebook© or Janrain©. This type extends the Metadata metadata type and inherits its `fullName` field.

**File Suffix and Directory Location**

Authentication providers are stored in the `authproviders` directory. The file name matches the URL suffix and the extension is `.authprovider`. For example, an auth provider with URL suffix `FacebookProvider` is stored in `authproviders/FacebookProvider.authprovider`.

**Version**

Authentication providers are available in API version 27.0 and later.

**Special Access Rules**

Only users with the “Customize Application” and “Manage AuthProviders” permissions can access this object.
### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authorizeUrl</td>
<td>string</td>
<td>Required, but only if providerType is OpenIdConnect. The OAuth authorization endpoint URL. Used only with OpenId Connect authentication providers. Available in API version 29.0 and later. In API version 33.0 and later, the behavior of this field changed to support the Salesforce-managed auth provider configuration, which allows Salesforce to manage the value for Facebook, Salesforce, LinkedIn, Twitter, or Google authentication. For more information, see the Usage section.</td>
</tr>
<tr>
<td>consumerKey</td>
<td>string</td>
<td>Required. The app’s key that is registered at the third-party single sign-on provider. In API version 33.0 and later, the behavior of this field changed to support the Salesforce-managed auth provider configuration, which allows Salesforce to manage the value for Facebook, Salesforce, LinkedIn, Twitter, or Google authentication. For more information, see the Usage section.</td>
</tr>
</tbody>
</table>
| consumerSecret        | string     | Required. The consumer secret of the app that is registered at the third-party single sign-on provider. This field cannot be updated. When using create(), this field must be encrypted. To create an encrypted form of the consumer secret from plain text:

1. Create an auth provider with the consumerSecret plain text value.
2. Save the auth provider.
3. Create an outbound change set that includes the auth provider component.

The new change set .xml file has an entry in the form

```xml
<consumerSecret>++XYZ++</consumerSecret>
```

where ```++XYZ++``` is the encrypted secret. In API version 33.0 and later, the behavior of this field changed to support the Salesforce-managed auth provider configuration, which allows Salesforce to manage the value for Facebook, Salesforce, LinkedIn, Twitter, or Google authentication. For more information, see the Usage section. |
<p>| customMetadataTypeRecord | string | Required, but only with custom authentication provider plug-ins, when ProviderType is Custom. The API name of the authentication provider. Available in API version 36.0 and later.                                                                                                                                  |
| defaultScopes         | string     | Required, but only if providerType is OpenIdConnect. The scopes to be sent with the authorization request, if not specified when a flow is started. Used only with OpenId Connect authentication providers. Available in API version 29.0 and later. In API version 33.0 and later, the behavior of this field changed to support the Salesforce-managed auth provider configuration, which allows Salesforce to manage the value for Facebook, Salesforce, LinkedIn, Twitter, or Google authentication. For more information, see the Usage section. |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeveloperName</td>
<td>string</td>
<td>Required. Used when referring to the auth provider from a program.</td>
</tr>
<tr>
<td>errorUrl</td>
<td>string</td>
<td>A custom error URL for the provider to use to report any errors.</td>
</tr>
<tr>
<td>executionUser</td>
<td>string</td>
<td>The username of the user who runs the Apex handler, which provides the context in which the Apex handler runs. For example, if the Apex handler creates a contact, the creation can be easily traced back to the registration process. A user is required when you specify a registration handler class. The user must have the Manage Users permission. Available in API version 27.0 and later.</td>
</tr>
<tr>
<td>executionUserId</td>
<td>string</td>
<td>The ID of the user who runs the Apex handler, which provides the context in which the Apex handler runs. For example, if the Apex handler creates a contact, the creation can be easily traced back to the registration process. A user is required when you specify a registration handler class. The user must have the Manage Users permission. Available in API version 27.0 and later.</td>
</tr>
<tr>
<td>friendlyName</td>
<td>string</td>
<td>Required. A user-friendly name for the provider.</td>
</tr>
<tr>
<td>iconUrl</td>
<td>string</td>
<td>The path to an icon to use as a button on the login page for a community. Users click the button to log in to a community with the associated authentication provider, such as Twitter® or LinkedIn®. Available in API version 32.0 and later.</td>
</tr>
<tr>
<td>idTokenIssuer</td>
<td>string</td>
<td>Only available if providerType is OpenIdConnect. This value identifies the source of the authentication token in the form <a href="https://UR1">https://UR1</a>. Used only with OpenID Connect authentication providers. If provided, Salesforce validates the returned id_token value. The OpenID Connect specification requires an id_token value to be returned with the access_token value. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td>includeOrgIdInIdentifier</td>
<td>Boolean</td>
<td>For Auth. Providers created before API version 27. Used to distinguish between users with the same user ID (such as sandbox copies). With this setting enabled, Salesforce stores the org ID of the third-party identity in addition to the user ID. This setting can be enabled, but it cannot be disabled. It applies only to a Salesforce Auth. Provider. Available in API version 27.0 and later.</td>
</tr>
<tr>
<td>LinkKickoffUrl</td>
<td>string</td>
<td>Provides the URL for linking existing Salesforce users to a third-party account. This field is read-only. Available in API version 43 and later.</td>
</tr>
<tr>
<td>logoutUrl</td>
<td>string</td>
<td>Provides a specific destination for users after they log out if they authenticated using the single sign-on flow. The URL must be fully qualified with an http or https prefix, such as <a href="https://acme.my.salesforce.com">https://acme.my.salesforce.com</a>. Available in API version 33.0 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OauthKickoffUrl</td>
<td>string</td>
<td>Provides the URL for obtaining OAuth access tokens for a third party. This field is read-only. Available in API version 43 and later.</td>
</tr>
<tr>
<td>portal</td>
<td>string</td>
<td>This field is used only with portals, which are deprecated. Salesforce does not support creating new portals, however, existing portals are supported.</td>
</tr>
<tr>
<td>plugin</td>
<td>string</td>
<td>An existing Apex class that extends the Auth.AuthProviderPluginClass abstract class. Available in API version 36.0 and later.</td>
</tr>
</tbody>
</table>
| providerType       | AuthProviderType (enumeration of type string) | Required. The third-party single sign-on provider to use. Valid values are:  
  - Facebook  
  - Google  
  - Salesforce  
  - Janrain  
  - LinkedIn (Available in API version 32.0 and later.)  
  - Twitter (Available in API version 32.0 and later.)  
  - OpenIdConnect (Available in API version 29.0 and later.)  
  - MicrosoftACS: Microsoft Access Control Service typically provides authentication for a Microsoft Office 365 service like SharePoint® Online. (Available in API version 31.0 and later.)  
  - GitHub: Use the GitHub provider to log in users of your Lightning Platform app to GitHub using OAuth. When logged in to GitHub, your app can make calls to GitHub APIs. The GitHub provider isn’t available as an SSO provider, so users can’t log in to your Salesforce org using their GitHub login credentials. (Available in API version 35.0 and later.)  
  - Custom: A provider configured with a custom authentication provider plug-in. (Available in API version 36.0 and later.)  
  - Note: This type requires values for the following fields:  
    - authorizeUrl  
    - defaultScopes  
    - tokenUrl  
    - userInfoUrl  
<p>| registrationHandler | string           | An existing Apex class that implements the Auth.RegistrationHandler interface.                                                                                                                                                                                                                                                                                                                                             |
| sendAccessTokenInHeader | boolean       | Required only if providerType is OpenIdConnect. When true, the access token is sent to the userInfoUrl in a header instead of a query string. Used only with OpenID Connect authentication providers. Available in API version 30.0 and later.                                                                                                           |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sendClientCredentialsInHeader</td>
<td>boolean</td>
<td>Required only if providerType is OpenIdConnect. When true, the client credentials are sent in a header, instead of a query string, to the tokenUrl. The credentials are in the standard OpenID Connect Basic Credentials header form, which is Basic &lt;token&gt;, where &lt;token&gt; is the base64-encoded string &quot;clientkey:clientsecret.&quot; Used only with OpenID Connect authentication providers. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td>SsoKickoffUrl</td>
<td>string</td>
<td>Provides the URL for performing single sign-on into Salesforce from a third party by using its third-party credentials. This field is read-only. Available in API version 43 and later.</td>
</tr>
<tr>
<td>tokenUrl</td>
<td>string</td>
<td>Required, but only if providerType is OpenIdConnect. The OAuth token endpoint URL. Used only with OpenID Connect authentication providers. Available in API version 29.0 and later. In API version 33.0 and later, the behavior of this field changed to support the Salesforce-managed auth provider configuration, which allows Salesforce to manage the value for Facebook, Salesforce, LinkedIn, Twitter, or Google authentication. For more information, see the Usage section.</td>
</tr>
<tr>
<td>userInfoUrl</td>
<td>string</td>
<td>Required, but only if providerType is OpenIdConnect. The OpenID Connect endpoint URL. Used only with OpenID Connect authentication providers. Available in API version 29.0 and later. In API version 33.0 and later, the behavior of this field changed to support the Salesforce-managed auth provider configuration, which allows Salesforce to manage the value for Facebook, Salesforce, LinkedIn, Twitter, or Google authentication. For more information, see the Usage section.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

```xml
<?xml version="1.0" encoding="UTF-8"?>
<AuthProvider xmlns="http://soap.sforce.com/2006/04/metadata">
    <consumerKey>yourappkey</consumerKey>
    <consumerSecret>PwdVxXjzu3NCZ3MD4He+wA==</consumerSecret>
    <executionUser>admin@your.org</executionUser>
    <friendlyName>FacebookAuthProvider</friendlyName>
    <providerType>Facebook</providerType>
    <registrationHandler>RegistrationHandler</registrationHandler>
</AuthProvider>
```

The following is an example package manifest that references the previous AuthProvider definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>FacebookAuthProvider</members>
        <name>AuthProvider</name>
    </types>
</Package>
```
Usage

For API version 33.0 and later when implementing the Salesforce-managed authentication provider configuration, you can have Salesforce create and manage the following fields for you:

- authorizeUrl
- consumerKey
- consumerSecret
- defaultScopes
- tokenURL
- userInfoUrl

To configure a Salesforce-managed authentication provider, set up a Facebook, Salesforce, LinkedIn, Twitter, or Google auth provider and leave all of the listed fields blank. Salesforce automatically creates values for you. However, if you provide a value for any of these fields, then consumerKey and consumerSecret must also be specified.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

AutoResponseRules

Represents an auto-response rule that sets conditions for sending automatic email responses to lead or case submissions based on the attributes of the submitted record. You can access rules metadata for all applicable objects, for a specific object, or for a specific rule on a specific object.

The `package.xml` syntax for accessing all auto-response rules for all objects is:

```xml
<types>
  <members>*</members>
  <name>AutoResponseRules</name>
</types>
```

All rules for a specific object uses a similar syntax without the wildcard. For example, all auto-response rules for the Case object would use this syntax:

```xml
<types>
  <members>Case</members>
  <name>AutoResponseRules</name>
</types>
```
You can also access specific auto-response rules for an object. The following example only accesses the “samplerule” and “newrule” auto-response rules on the Case object. Notice that for this example the type name syntax is `AutoResponseRule` and not `AutoResponseRules`.

```xml
<types>
  <members>Case.samplerule</members>
  <members>Case.newrule</members>
  <name>AutoResponseRule</name>
</types>
```

**File Suffix and Directory Location**

AutoResponseRules for an object have the suffix `.autoResponseRules` and are stored in the `autoResponseRules` folder. For example, all Case auto-response rules are stored in the `Case.autoResponseRules` file.

**Version**

AutoResponseRules components are available in API version 27.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoresponserule</td>
<td><code>AutoResponseRule[]</code></td>
<td>Represents the definitions of the named auto-response rules.</td>
</tr>
</tbody>
</table>

**AutoResponseRule**

Represents whether a rule is active or not and the order in which the entry is processed in the rule.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Indicates whether the autorespons rule is active (true) or not (false).</td>
</tr>
<tr>
<td>fullname</td>
<td>string</td>
<td>Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See <code>createMetadata()</code> to see an example of this field specified for a call. This value cannot be <code>null</code>.</td>
</tr>
<tr>
<td>ruleEntry</td>
<td><code>RuleEntry[]</code></td>
<td>Represents the type and description for the auto-response rule.</td>
</tr>
</tbody>
</table>

**RuleEntry**

Represents the fields used by the rule.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Advanced filter conditions that were specified for the rule.</td>
</tr>
<tr>
<td>criteriaItems</td>
<td>FilterItem[]</td>
<td>The items in the list that define the assignment criteria.</td>
</tr>
<tr>
<td>formula</td>
<td>string</td>
<td>The validation formula.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Specify either <code>formula</code> or <code>criteriaItems</code>, but not both fields.</td>
</tr>
<tr>
<td>replyToEmail</td>
<td>string</td>
<td>The email address that appears in the reply-to header.</td>
</tr>
<tr>
<td>senderEmail</td>
<td>string</td>
<td>The email address of the person or queue sending the email notification.</td>
</tr>
<tr>
<td>senderName</td>
<td>string</td>
<td>The name of the person or queue sending the email notification.</td>
</tr>
<tr>
<td>template</td>
<td>string</td>
<td>Specifies the template to use for the email that is automatically sent to the designated recipient. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example `AutoResponseRules` component:

```xml
<AutoResponseRules xmlns="http://soap.sforce.com/2006/04/metadata">
  <autoResponseRule>
    <fullName>ajbdeploytest2</fullName>
    <active>false</active>
    <ruleEntry>
      <criteriaItems>
        <field>Case.Description</field>
        <operation>contains</operation>
        <value>testing</value>
      </criteriaItems>
      <senderEmail>test@test.org</senderEmail>
      <senderName>tester name j</senderName>
      <replyToEmail>test@test.org</replyToEmail>
      <template>emailtemplate</template>
    </ruleEntry>
  </autoResponseRule>
</AutoResponseRules>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
Bot

Represents a definition of an Einstein Bot configuration that can have one or more versions. Only one version can be active. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

Bot components have the suffix .bot and are stored in the bot folder.

Version

Bot components are available in API version 43.0 and later.

Special Access Rules

Bot is available only if Chat and Einstein Bots are enabled in your org.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>botMlDomain</td>
<td>LocalMlDomain</td>
<td>Represents the Einstein intent set that groups intents, entities, and variables associated with a bot. All Einstein Bot versions under the same bot now share an intent set. Available in API version 44.0 and later.</td>
</tr>
<tr>
<td>botUser</td>
<td>string</td>
<td>Represents a user profile associated with the bot. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>botVersions</td>
<td>BotVersion</td>
<td>Represents the configuration details for a specific Einstein Bots version, including dialogs, intents, entities, and variables.</td>
</tr>
<tr>
<td>contextVariables</td>
<td>ConversationContextVariable</td>
<td>Represents the context variables that enable your bot to gather customer information regardless of channel. Available in API version 45.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the bot.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Label that identifies the bot throughout the Salesforce user interface.</td>
</tr>
</tbody>
</table>

ConversationContextVariable

A context variable local to the current bot version. Available in API version 45.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextVariableMappings</td>
<td>ContextVariableMapping</td>
<td>Represents the mapping between a context variable, channel type, and sObject field.</td>
</tr>
</tbody>
</table>
### ConversationContextVariableMapping

Represents the mapping between a context variable, channel type, and sObject field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>string</td>
<td>Required. The API name of an SObject field to be used as part of the mapping.</td>
</tr>
<tr>
<td>messageType</td>
<td>MessageType</td>
<td>Required. Represents the channel, either SMS messaging or Chat. Required. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td></td>
</tr>
</tbody>
</table>
### Metadata Types

#### Field Name | Field Type | Description
--- | --- | ---
| | | • WhatsApp
| | | • Phone
| | | Required. SObject type for the field property defined as part of the mapping. Valid values are:
| | | • LiveChatTranscript
| | | • MessagingEndUser
| | | • MessagingSession

### LocalMlDomain

An Einstein Intent Set local to the current bot version.

#### Field Name | Field Type | Description
--- | --- | ---
| label | string | Label that represents an Einstein Intent Set local to the current bot version throughout the Salesforce user interface.
| mlIntents | MlIntent[] | List of intents associated with this local intent set.
| mlSlotClasses | MlSlotClass[] | List of entities associated with this local intent set.
| name | string | Required. This unique name prevents conflicts with other local Einstein Intent Sets. This name can contain only underscores and alphanumeric characters and must be unique in your org. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.

### Declarative Metadata Sample Definition

The following is an example of a Bot. This example has been trimmed to make it easier to read.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Bot xmlns="http://soap.sforce.com/2006/04/metadata">
    <botMlDomain>
        <label>Astros Pizza</label>
        <mlIntents>
            <developerName>New Order</developerName>
            <label>New Order</label>
            <mlIntentUtterances>
                <utterance>Today is pie day so I want pie</utterance>
            </mlIntentUtterances>
        </mlIntents>
        <mlSlotClasses>
            <developerName>Size</developerName>
            <extractionType>Value</extractionType>
            <label>Size</label>
            <mlSlotClassValues>
                <synonymGroup>
                    188
                </synonymGroup>
            </mlSlotClassValues>
        </mlSlotClasses>
    </botMlDomain>
</Bot>
```
<languages>en_US</languages>
/terms>Big</terms>
/terms>Extra Large</terms>
/terms>X-Large</terms>
/terms>Grande</terms>
/terms>Huge</terms>
</synonymGroup>
/value>Large</value>
</mlSlotClassValues>
</mlSlotClasses>
</name>Astros_Pizza_ld1</name>
</botMIDomain>
<botVersions>
<fullName>v1</fullName>
<botDialogGroups>
<developerName>Order_Management</developerName>
<label>Order Management</label>
</botDialogGroups>
<botDialogs>
<botDialogGroup>Order_Management</botDialogGroup>
<botSteps>
<botMessages>
<message>Pizza Time! _PACKET</message>
</botMessages>
</botSteps>
<botSteps>
<botStepConditions>
<leftOperandName>Verified_User</leftOperandName>
<leftOperandType>ConversationVariable</leftOperandType>
<operatorType>Equals</operatorType>
<rightOperandValue>false</rightOperandValue>
</botStepConditions>
</botSteps>
<botNavigation>
<botNavigationLinks>
<targetBotDialog>Customer_Verification</targetBotDialog>
</botNavigationLinks>
</botNavigation>
</botSteps>
<botSteps>
<botStepConditions>
<leftOperandName>Location</leftOperandName>
<leftOperandType>ConversationVariable</leftOperandType>
<operatorType>IsNotSet</operatorType>
</botStepConditions>
</botSteps>
<botNavigation>
<botNavigationLinks>
<targetBotDialog>Select_Location</targetBotDialog>
</botNavigationLinks>
</botNavigation>
</botSteps>

<botSteps><botInvocation>
  <invocationActionName>CreateOrderService</invocationActionName>
  <invocationActionType>apex</invocationActionType>
  <invocationMappings>
    <parameterName>customer</parameterName>
    <type>Input</type>
    <variableName>Contact</variableName>
    <variableType>ConversationVariable</variableType>
  </invocationMappings>
  <invocationMappings>
    <parameterName>location</parameterName>
    <type>Input</type>
    <variableName>Location</variableName>
    <variableType>ConversationVariable</variableType>
  </invocationMappings>
  <invocationMappings>
    <parameterName>output</parameterName>
    <type>Output</type>
    <variableName>Pizza_Order</variableName>
    <variableType>ConversationVariable</variableType>
  </invocationMappings>
</botInvocation></botVariableOperation>
<botMessages>
  <message>Perfect, let's work on your order from our {!Location.Name} location</message>
</botMessages></botSteps><botNavigation><botNavigationLinks>
  <targetBotDialog>Add_Items_to_Order</targetBotDialog>
</botNavigationLinks></botNavigation>
<showInFooterMenu>false</showInFooterMenu>
</botDialogs>
<conversationVariables>
<dataType>Object</dataType>
<developerName>Contact</developerName>
<label>Contact</label>
</conversationVariables>
<conversationVariables>
<dataType>Text</dataType>
<developerName>Delivery_Address</developerName>
<label>Delivery Address</label>
</conversationVariables>
<conversationVariables>
<dataType>Object</dataType>
<developerName>Pizza_Order</developerName>
<label>Pizza Order</label>
</conversationVariables>
<entryDialog>Welcome</entryDialog>
<mainMenuDialog>Main_Menu</mainMenuDialog>
</botVersions>
<contextVariables>
<contextVariableMappings>
<SObjectType>LiveChatTranscript</SObjectType>
(fieldName)LiveChatTranscript.ChatKey
<messageType>WebChat</messageType>
</contextVariableMappings>
<dataType>Text</dataType>
<developerName>ChatKey</developerName>
<label>Chat Key</label>
</contextVariables>
<contextVariables>
<contextVariableMappings>
<SObjectType>LiveChatTranscript</SObjectType>
(fieldName)LiveChatTranscript.ContactId
<messageType>WebChat</messageType>
</contextVariableMappings>
<dataType>Id</dataType>
<developerName>ContactId</developerName>
<label>Contact Id</label>
</contextVariables>
<contextVariableMappings>
<SObjectType>LiveChatTranscript</SObjectType>
(fieldName)LiveChatTranscript.LiveChatVisitorId
<messageType>WebChat</messageType>
</contextVariableMappings>
<dataType>Id</dataType>
<developerName>EndUserId</developerName>
<label>End User Id</label>
</contextVariables>
<contextVariableMappings>
<SObjectType>LiveChatTranscript</SObjectType>
(fieldName)LiveChatTranscript.Id
</contextVariableMappings>
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Pizza_Bot</members>
    <name>Bot</name>
  </types>
  <version>45.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**BotVersion**

Represents the configuration details for a specific Einstein Bot version, including dialogs, intents, entities, and variables. This type extends the Metadata metadata type and inherits its fullName field.

**File Suffix and Directory Location**

BotVersion components have the suffix `.bot` and are stored in the `bot` folder. BotVersion is a top-level child of Bot and shares its suffix and file directory.

**Version**

BotVersion components are available in API version 43.0 and later.

**Special Access Rules**

BotVersion is available only if Chat and Einstein Bots are enabled in your org.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>botDialogGroups</td>
<td>BotDialogGroup[]</td>
<td>The list of dialog groups in this bot version.</td>
</tr>
</tbody>
</table>
### BotDialogGroup

The list of dialog groups in this bot version.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the bot dialog group.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. This unique name prevents conflicts with other dialog groups associated with the same bot version. This name can contain only underscores and alphanumeric characters. The name must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. A label that identifies the dialog group throughout the Salesforce user interface.</td>
</tr>
</tbody>
</table>

### BotDialog

The list of dialogs in this bot version.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>botDialogGroup</td>
<td>string</td>
<td>The bot dialog group that contains this bot dialog.</td>
</tr>
<tr>
<td>botSteps</td>
<td>BotStep[]</td>
<td>A list of steps that are executed as part of the dialog.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the bot dialog.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. This unique name prevents conflicts with other dialogs associated with the same bot version. This name can contain only underscores and alphanumeric characters. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.</td>
</tr>
</tbody>
</table>
Field Name | Field Type | Description
---|---|---
label | string | Required. A label that identifies the dialog throughout the Salesforce user interface.
mlIntent | string | Required. The name of the intent associated with a dialog.
mlIntentTrainingEnabled | boolean | Indicates whether Einstein is turned on to train an intent model for the dialog intent (true) or turned off for the exact match option (false). The default value is false. Available in API version 46.0 and later.
showInFooterMenu | boolean | Indicates whether to show this dialog in the Bot Options menu. The default value is false.

**BotStep**

A step that is executed as part of the dialog.

Field Name | Field Type | Description
---|---|---
booleanFilter | string | This field is reserved for future use.
botInvocation | BotInvocation | Bot Invocation used by a BotStep of type Invocation.
botMessages | BotMessage[] | List of bot messages used by a BotStep of type Message.
botStepConditions | BotStepCondition[] | List of BotStep conditions associated with a BotStep of type Group.
botSteps | BotStep[] | List of BotSteps associated to a Bot Step of type Group.
botVariableOperation | BotVariableOperation[] | Bot Variable Operation used by a BotStep of type VariableOperation.
conversationSystemMessage | ConversationSystemMessage[] | System messages that represent an action for a Bot Step, such as transferring to an agent or ending a chat. Available in API version 46.0 and later.
type | BotStepType (enumeration of type string) | Required. Valid values are:
- Navigation
- Invocation
- VariableOperation
- Message
- Wait
- Group

**BotInvocation**

Bot Invocation used by a BotStep of type Invocation.
## Metadata Types

### BotInvocation

List of Bot Invocation Mappings for a Bot Invocation.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>invocationActionName</td>
<td>string</td>
<td>Required. The name of the invocable action used by a Bot Invocation.</td>
</tr>
<tr>
<td>invocationActionType</td>
<td>ConversationInvocableType</td>
<td>Available dialog action types are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• apex</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• flow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• standardInvocableAction</td>
</tr>
<tr>
<td>invocationMappings</td>
<td>BotInvocationMapping[]</td>
<td>List of Bot Invocation Mappings for a Bot Invocation.</td>
</tr>
</tbody>
</table>

### BotInvocationMapping

List of Bot Invocation Mappings for a Bot Invocation.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameterName</td>
<td>string</td>
<td>Required. Name of an Input/Output parameter of the parent Bot Invocation target.</td>
</tr>
<tr>
<td>type</td>
<td>BotInvocationMappingType</td>
<td>Required. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• Input</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Output</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Literal Value to be assigned to the specified parameter.</td>
</tr>
<tr>
<td>variableName</td>
<td>string</td>
<td>Name of the Bot Variable that is used as part of an Invocation mapping.</td>
</tr>
<tr>
<td>variableType</td>
<td>ConversationVariableType</td>
<td>Type of the variable used in this invocation mapping. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• ConversationVariable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ContextVariable</td>
</tr>
</tbody>
</table>

### BotMessage

A bot messages used by a BotStep of type Message.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>string</td>
<td>Required. Message to display as part of an outgoing message from the bot to the customer.</td>
</tr>
</tbody>
</table>

### BotNavigation

Bot navigation used by a BotStep of type Navigation.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>BotNavigationType</td>
<td>Required. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Redirect</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TransferToAgent</td>
</tr>
</tbody>
</table>

### BotNavigationLink

List of Bot Navigation links associated with a Bot Navigation of type Call or Redirect.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Label displayed when more than one Bot Navigation Link is available under a Bot Navigation of type Redirect. The target dialog label is used when no label is provided.</td>
</tr>
<tr>
<td>targetBotDialog</td>
<td>string</td>
<td>Required. Name of the target dialog to be called as part of this Bot Navigation Link.</td>
</tr>
</tbody>
</table>

### BotStepCondition

List of BotStep conditions associated with a BotStep of type Group.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operatorType</td>
<td>BotStepConditionOperatorType</td>
<td>Required. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Equals</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NotEquals</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IsSet</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IsNotSet</td>
</tr>
<tr>
<td>rightOperandValue</td>
<td>string</td>
<td>Value that is used as the right side of the condition operation. This value is ignored when using IsSet and IsNotSet operators.</td>
</tr>
<tr>
<td>leftOperandName</td>
<td>string</td>
<td>Required. Name of the variable used as the left side of the condition operation.</td>
</tr>
<tr>
<td>leftOperandType</td>
<td>ConversationVariableType</td>
<td>Required. Type of the variable used as the left side of the condition operation. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ConversationVariable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ContextVariable</td>
</tr>
</tbody>
</table>
**BotVariableOperation**

Bot variable operation used by a BotStep of type `VariableOperation`.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>botInvocation</code></td>
<td><code>BotInvocation</code></td>
<td>Bot Invocation used to provide Dynamic choices by a Bot Variable Operation of type <code>Collect</code> and <code>quickReplyType</code> of <code>Dynamic</code>.</td>
</tr>
<tr>
<td><code>botMessages</code></td>
<td><code>BotMessage[]</code></td>
<td>List of Bot Messages used as prompt messages by a Bot Variable Operation of type <code>Collect</code>.</td>
</tr>
<tr>
<td><code>botQuickReplyOptions</code></td>
<td><code>BotQuickReplyOption[]</code></td>
<td>List of static choice options used by a Bot Variable Operation of type <code>Collect</code> and <code>quickReplyType</code> of <code>Static</code>.</td>
</tr>
<tr>
<td><code>botVariableOperands</code></td>
<td><code>BotVariableOperand[]</code></td>
<td>List of Bot Variable Operands associated with a Bot Variable of type <code>Set</code> or <code>Unset</code>.</td>
</tr>
<tr>
<td><code>invalidInputBotNavigation</code></td>
<td><code>BotNavigation</code></td>
<td>Bot Navigation used by a Bot Variable Operation of type <code>Collect</code>. This navigation is executed when the associated Bot Invocation doesn’t return any options.</td>
</tr>
<tr>
<td><code>quickReplyOptionTemplate</code></td>
<td><code>string</code></td>
<td>Formula template used to resolve a label for Dynamic choice options of type <code>Object</code>.</td>
</tr>
<tr>
<td><code>quickReplyType</code></td>
<td><code>BotQuickReplyType</code> (enumeration of type <code>string</code>)</td>
<td>Valid values are: <code>Static</code> and <code>Dynamic</code></td>
</tr>
<tr>
<td><code>quickReplyWidgetType</code></td>
<td><code>BotWidgetType</code> (enumeration of type <code>string</code>)</td>
<td>Valid values are: <code>Menu</code> and <code>Buttons</code></td>
</tr>
<tr>
<td><code>type</code></td>
<td><code>BotVariableOperationType</code> (enumeration of type <code>string</code>)</td>
<td>Required. Valid values are: <code>Set</code>, <code>Unset</code>, and <code>Collect</code>.</td>
</tr>
</tbody>
</table>

**BotQuickReplyOption**

List of static choice options used by a bot variable operation of type `Collect` and `quickReplyType` of `Static`.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>literalValue</code></td>
<td><code>string</code></td>
<td>Required. Value to be displayed as a menu or button choice to your customer.</td>
</tr>
</tbody>
</table>

**BotVariableOperand**

List of bot variable operands associated with a bot variable of type `Set` or `Unset`. 
Disables auto-fill behavior for a bot variable under a bot variable operation of type `Collect`.

**sourceName**
Name of the source CustomField or MlSlotClass used in the variable operation.

**sourceType**
Valid values are:
- `StandardConversationVariable`
- `ConversationVariable`
- `ContextVariable`
  
  Available in API 45.0 and later.
- `MlSlotClass`
- `StandardMlSlotClass`
- `Value`

**sourceValue**
Literal value used as the source for this variable operation.

**targetName**
Required. Name of the target variable used in the variable operation.

**targetType**
Required. Type of the target used in the variable operation. Valid values are:
- `ConversationVariable`
- `ContextVariable`

---

**ConversationSystemMessage**
System messages that represent an action for a Bot Step, such as transferring to an agent or ending a chat. Available in API version 46.0 and later.

**Field Name**
**Field Type**
**Description**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>systemMessageMappings</td>
<td>ConversationSystemMessageType</td>
<td>Defines the type of system message to be sent.</td>
</tr>
<tr>
<td>type</td>
<td>ConversationSystemMessageType</td>
<td>Required. Valid value is Transfer.</td>
</tr>
</tbody>
</table>

**ConversationSystemMessageMapping**
List of mappings that indicate additional information provided for the system message. Available in API version 46.0 and later.

**Field Name**
**Field Type**
**Description**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mappingType</td>
<td>ConversationMappingType</td>
<td>Required. Defines the type of mapping used in the record. Valid values are Input and Output.</td>
</tr>
</tbody>
</table>
**Description Field Type Field Name**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameterType</td>
<td>string (enumeration of type string)</td>
<td>Required. Defines the type of parameter the value is mapped to. Valid value is Transfer.</td>
</tr>
<tr>
<td>variableName</td>
<td>string</td>
<td>Required. Name of the variable that contains the value passed to the system message.</td>
</tr>
</tbody>
</table>

### ConversationVariable

A container that stores a specific piece of data collected from the customer. You can use variables within dialog actions as both inputs and outputs. Available in API version 44.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. This name can contain only underscores and alphanumeric characters and must be unique in your org. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.</td>
</tr>
</tbody>
</table>
| dataType     | ConversationVariableDataType (enumeration of type string) | Required. Valid values are:  
  - Boolean  
  - Currency  
  - Date  
  - DateTime  
  - Id (available in API 45.0 and later)  
  - Object  
  - Number  
  - Text  |
| label        | string      | Required. Label that identifies a variable throughout the Salesforce user interface.                                                           |
| SObjectType  | string      | Specifies the SObjectType of the ID stored in a bot variable. Valid values are:  
  - BotDefinition  
  - Queue  |

### Declarative Metadata Sample Definition

The following is an example of a BotVersion.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Bot xmlns="http://soap.sforce.com/2006/04/metadata">
  <botMlDomain>
    <label>Astros Pizza</label>
    <mlIntents>
      <developerName>New_Order</developerName>
      <label>New Order</label>
    </mlIntents>
  </botMlDomain>
</Bot>
```
<mlIntentUtterances>
  <utterance>Today is pie day so I want pie</utterance>
</mlIntentUtterances>

<mlSlotClasses>
  <developerName>Size</developerName>
  <extractionType>Value</extractionType>
  <label>Size</label>
  <mlSlotClassValues>
    <synonymGroup>
      <languages>en_US</languages>
      <terms>Big</terms>
      <terms>Extra Large</terms>
      <terms>X-Large</terms>
      <terms>Grande</terms>
      <terms>Huge</terms>
    </synonymGroup>
    <value>Large</value>
  </mlSlotClassValues>
</mlSlotClasses>

<name>Astros_Pizza_ld1</name>
</botMlDomain>

<botVersions>
  <fullName>v1</fullName>
  <botDialogGroups>
    <developerName>Order_Management</developerName>
    <label>Order Management</label>
  </botDialogGroups>
  <botDialogs>
    <botDialogGroup>Order_Management</botDialogGroup>
    <botSteps>
      <botMessages>
        <message>🍕🍕Pizza Time! 🍕🍕</message>
      </botMessages>
      <type>Message</type>
    </botSteps>
    <botSteps>
      <botStepConditions>
        <leftOperandName>Verified_User</leftOperandName>
        <leftOperandType>ConversationVariable</leftOperandType>
        <operatorType>Equals</operatorType>
        <rightOperandValue>false</rightOperandValue>
      </botStepConditions>
    </botSteps>
    <botNavigation>
      <botNavigationLinks>
        <targetBotDialog>Customer_Verification</targetBotDialog>
      </botNavigationLinks>
      <type>Call</type>
    </botNavigation>
    <type>Navigation</type>
    <botSteps>
      <type>Group</type>
    </botSteps>
  </botDialogs>
</botVersions>
<botSteps>
  <botStepConditions>
    <leftOperandName>Location</leftOperandName>
    <leftOperandType>ConversationVariable</leftOperandType>
    <operatorType>IsNotSet</operatorType>
  </botStepConditions>
  <botSteps>
    <botNavigation>
      <botNavigationLinks>
        <targetBotDialog>Select_Location</targetBotDialog>
      </botNavigationLinks>
      <type>Call</type>
    </botNavigation>
    <type>Navigation</type>
  </botSteps>
</botSteps>
</botSteps>
<botVariableOperation>
  <botInvocation>
    <invocationActionName>CreateOrderService</invocationActionName>
    <invocationActionType>apex</invocationActionType>
    <invocationMappings>
      <parameterName>customer</parameterName>
      <type>Input</type>
      <variableName>Contact</variableName>
      <variableType>ConversationVariable</variableType>
    </invocationMappings>
    <invocationMappings>
      <parameterName>location</parameterName>
      <type>Input</type>
      <variableName>Location</variableName>
      <variableType>ConversationVariable</variableType>
    </invocationMappings>
    <invocationMappings>
      <parameterName>output</parameterName>
      <type>Output</type>
      <variableName>Pizza_Order</variableName>
      <variableType>ConversationVariable</variableType>
    </invocationMappings>
  </botInvocation>
  <type>Set</type>
</botVariableOperation>
<botSteps>
  <botMessages>
    <message>Perfect, let's work on your order from our {(!Location.Name) location</message>
  </botMessages>
</botSteps>
</botSteps>
<botNavigation>
The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Astros Pizza_Bot.v1</members>
    <name>BotVersion</name>
  </types>
  <version>45.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type doesn’t support the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
BrandingSet

Represents the definition of a set of branding properties for a community, as defined in the Branding Panel in Community Builder. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

BrandingSet components have the suffix brandingSet and are stored in the brandingSets folder.

Version

BrandingSet components are available in API version 40.0 and later.

Special Access Rules

The BrandingSet object is available when at least one of the following is enabled in your org: Salesforce Communities, Surveys, or Lightning Experience.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>brandingSetProperty</td>
<td>BrandingSetProperty[]</td>
<td>An array containing the name and value of each branding property, such as TextColor:#333.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the set of branding properties.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The user interface name of the set of branding properties.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>The assigned branding set definition for this BrandingSet.</td>
</tr>
</tbody>
</table>

BrandingSetProperty

Represents the definition of a branding property in the Branding panel in Community Builder.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>propertyName</td>
<td>string</td>
<td>Required. The name of the branding property, such as TextColor.</td>
</tr>
<tr>
<td>propertyValue</td>
<td>string</td>
<td>The value of the branding property, such as #333.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a BrandingSet component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<BrandingSet xmlns="http://soap.sforce.com/2006/04/metadata">
  <brandingSetProperty>
    ...
  </brandingSetProperty>
</BrandingSet>
```
<brandingSetProperty>
  <propertyName>TextTransformStyle</propertyName>
  <propertyValue>uppercase</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>DetailTextColor</propertyName>
  <propertyValue>#696969</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>BorderColor</propertyName>
  <propertyValue>#D4D4D4</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>HeaderImage</propertyName>
  <propertyValue></propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>HeaderFonts</propertyName>
  <propertyValue>Montserrat</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>CardBackgroundColor</propertyName>
  <propertyValue>rgba(255, 255, 255, 0)</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>LoginBackgroundColor</propertyName>
  <propertyValue>#F4F4F4</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>ActionColor</propertyName>
  <propertyValue>#2574A9</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>_ActionColorTrans</propertyName>
  <propertyValue>rgba(25, 124, 190, 0.9)</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>CompanyLogo</propertyName>
  <propertyValue></propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>LoginBackgroundImage</propertyName>
  <propertyValue>../../../../sfsites/picasso/core/external/salesforceIdentity/images/background.jpg?v=1</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>_LinkColorDarker</propertyName>
  <propertyValue>#135F90</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>_ActionColorDarker</propertyName>
  <propertyValue>#135F90</propertyValue>
</brandingSetProperty>
<brandingSetProperty>
  <propertyName>_HoverColor</propertyName>
  <propertyValue></propertyValue>
</brandingSetProperty>
The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>MyBrandingSet</members>
    <name>BrandingSet</name>
  </types>
  <version>40.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**CallCenter**

Represents the Call Center definition used to integrate Salesforce with a third-party computer-telephony integration (CTI) system.

**File Suffix and Directory Location**

CallCenter components have the suffix `callCenter` and are stored in the `callCenters` folder.
Version
CallCenter components are available in API version 27.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterUrl</td>
<td>string</td>
<td>Optional field. A URL that points to an adapter.</td>
</tr>
<tr>
<td>displayName</td>
<td>string</td>
<td>The display name of this call center.</td>
</tr>
<tr>
<td>displayNameLabel</td>
<td>string</td>
<td>The label of the displayName field in Call Center setup page.</td>
</tr>
<tr>
<td>internalNameLabel</td>
<td>string</td>
<td>The label of the internalName field in Call Center setup page.</td>
</tr>
<tr>
<td>version</td>
<td>string</td>
<td>The version of this call center.</td>
</tr>
<tr>
<td>sections</td>
<td>CallCenterSection[]</td>
<td>Custom setup items defined for this call center.</td>
</tr>
</tbody>
</table>

CallCenterSection

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>CallCenterItem[]</td>
<td>Contains the label, name, and value that describe the sections.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label of the section.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the section.</td>
</tr>
</tbody>
</table>

CallCenterItem

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>The label of the custom setup item.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the custom setup item.</td>
</tr>
<tr>
<td>value</td>
<td>int or URL</td>
<td>The value of the custom setup item.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a CallCenter component:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CallCenter xmlns="http://soap.sforce.com/2006/04/metadata">
  <adapterUrl>http://localhost:11000</adapterUrl>
  <displayName>Demo Call Center Adapter</displayName>
  <displayNameLabel>Display Name</displayNameLabel>
</CallCenter>
```
<internalNameLabel>Internal Name</internalNameLabel>
<sections>
  <items>
    <label>Description</label>
    <name>reqDescription</name>
    <value>Demo Call Center Adapter</value>
  </items>
  <items>
    <label>CTI Connector ProgId</label>
    <name>reqProgId</name>
    <value>DemoAdapter.DemoAdapter.1</value>
  </items>
  <items>
    <label>Version</label>
    <name>reqVersion</name>
    <value>3.0</value>
  </items>
  <items>
    <label>CTI Adapter URL</label>
    <name>reqAdapterUrl</name>
    <value>http://localhost:11000</value>
  </items>
  <items>
    <label>General Information</label>
    <name>reqGeneralInfo</name>
  </items>
</sections>
<sections>
  <items>
    <label>Outside Prefix</label>
    <name>reqOutsidePrefix</name>
    <value>1</value>
  </items>
  <items>
    <label>Long Distance Prefix</label>
    <name>reqLongDistPrefix</name>
    <value>1</value>
  </items>
  <items>
    <label>International Prefix</label>
    <name>reqInternationalPrefix</name>
    <value>01</value>
  </items>
  <items>
    <label>Dialing Options</label>
    <name>reqDialingOptions</name>
  </items>
</sections>
</CallCenter>

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](#).
CampaignInfluenceModel

Represents a campaign influence model used by Customizable Campaign Influence.

You can’t configure Customizable Campaign Influence via the Metadata API, but you can add a campaign influence model.

⚠️ **Note:** This information applies only to Customizable Campaign Influence and not to Campaign Influence 1.0.

**File Suffix and Directory Location**

CampaignInfluenceModel values are stored in the `campaignInfluenceModels` directory of the corresponding package directory. The file name matches the model name, and the extension is `.campaignInfluenceModel`.

**Version**

CampaignInfluenceModel components are available in API version 38.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Indicates whether the model is active. Active models can generate campaign influence records. Deactivating a model deletes its campaign influence records. Custom models are always active and this field is ignored. This field is available beginning with API version 40.0.</td>
</tr>
<tr>
<td>isDefaultModel</td>
<td>boolean</td>
<td>Indicates if the model is the default model or not. Only campaign influence records associated with the default model appear on campaigns and opportunities. You can only have one default model at a time. A model must be active to become the default model. Activating or deactivating custom models does not automatically generate or delete campaign influence records.</td>
</tr>
<tr>
<td>isModelLocked</td>
<td>boolean</td>
<td>Indicates if the model is locked or not. Campaign Influence records for locked models can be manipulated only via the API.</td>
</tr>
<tr>
<td>modelDescription</td>
<td>string</td>
<td>A description of the influence model.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>A unique name for the model.</td>
</tr>
<tr>
<td>recordPreference</td>
<td>picklist</td>
<td>The value of this field determines when to create campaign influence records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AllRecords: Creates records regardless of the revenue attribution percentage.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RecordsWithAttribution: Creates records only when the revenue attribution is greater than 0%.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available beginning with API version 41.0.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example of a CampaignInfluenceModel component that represents the default Salesforce campaign influence attribution model. The default `isDefaultModel` value of `true` can be changed if another model is created and set as the default model. The `isModelLocked` value of `true` means that Campaign Influence records for this model can be seen in the UI, but not created, updated, or deleted.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CampaignInfluenceModel xmlns="http://soap.sforce.com/2006/04/metadata">
  <isDefaultModel>true</isDefaultModel>
  <isModelLocked>true</isModelLocked>
  <active>true</active>
  <recordPreference>AllRecords</recordPreference>
  <modelDescription>Primary Campaign gets 100% of the revenue share</modelDescription>
  <name>Salesforce Model</name>
</CampaignInfluenceModel>
```

The following is an example of a CampaignInfluenceModel component that creates an influence model called Last Touch, which will not be the default model.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CampaignInfluenceModel xmlns="http://soap.sforce.com/2006/04/metadata">
  <isDefaultModel>false</isDefaultModel>
  <isModelLocked>true</isModelLocked>
  <active>true</active>
  <recordPreference>RecordsWithAttribution</recordPreference>
  <modelDescription>This model gives 100% influence attribution to the last campaign that touched the contact.</modelDescription>
  <name>Last Touch</name>
</CampaignInfluenceModel>
```

CaseSubjectParticle

Represents the Social Business Rules custom format for the Case Subject field on cases created from inbound social posts.

File Suffix and Directory Location

CaseSubjectParticle components have the suffix `.CaseSubjectParticle` and are stored in the `CaseSubjectParticles` folder.

Version

CaseSubjectParticle is available in API version 41.0 and later.
### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>index</td>
<td>int</td>
<td>Required. The order in which the custom <strong>Case Subject</strong> is generated, meaning if the social network is 0 and the social message is 1, then the subject generates as `Twitter</td>
</tr>
<tr>
<td>textField</td>
<td>string</td>
<td>Specifies inbound social content added to <strong>Case Subject</strong> in case records.</td>
</tr>
</tbody>
</table>
| type       | CaseSubjectParticleType (enumeration of type string) | Required. Specifies the custom **Case Subject** format from which inbound social content appears in case records. Valid values are:  
  - ProvidedString  
  - Source  
  - MessageType  
  - SocialHandle  
  - SocialNetwork  
  - Sentiment  
  - RealName  
  - Content  
  - PipeSeparator  
  - ColonSeparator  
  - HyphenSeparator |

### Declarative Metadata Sample Definition

This is a sample of a `.CaseSubjectParticle` file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns=http://soap.sforce.com/2006/04/metadata">
  <types>
    <name>CaseSubjectParticle</name>
  </types>
  <version>41.0</version>
</Package>
```

### Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
Certificate

Represents a certificate used for digital signatures which verify that requests are coming from your org. Certificates are used for either authenticated single sign-on with an external website, or when using your org as an identity provider. This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.

File Suffix and Directory Location

Certificate components have the suffix .crt and are stored in the certs folder.

Version

Certificate components are available in API version 36.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>caSigned</td>
<td>boolean</td>
<td>Required. Indicates whether this certificate is signed by the issuer (true) or not (false).</td>
</tr>
<tr>
<td>encryptedWithPlatformEncryption</td>
<td>boolean</td>
<td>Indicates whether this certificate is encrypted with Platform Encryption.</td>
</tr>
<tr>
<td>expirationDate</td>
<td>dateTime</td>
<td>The date that this certificate expires and is no longer usable. For self-signed certificates, if keySize is 2048 bits, the expiration date is automatically 1 year after you create the certificate. If keySize is 4096 bits, the expiration date is automatically 2 years after you create the certificate. For CA-signed certificates, expirationDate is automatically updated to the signed certificate's expiration date when a signed certificate chain is uploaded. The date format is YYYY-MM-DD.</td>
</tr>
<tr>
<td>keySize</td>
<td>int</td>
<td>Certificate keys can be either 2048 bits or 4096 bits. A certificate with 4096-bit keys lasts 2 years, and a certificate with 2048-bit keys lasts 1 year. Certificates with 2048-bit keys are faster than certificates with 4096-bit keys. If keySize isn't specified when you create a certificate, the key size defaults to 2048 bits.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. A user-friendly name for the certificate that appears in the Salesforce user interface, such as in Certificate and Key Management. Limit: 64 characters.</td>
</tr>
<tr>
<td>privateKeyExportable</td>
<td>boolean</td>
<td>Indicates whether this certificate's private key is exportable. If privateKeyExportable isn't specified when you create a certificate, its default value is true.</td>
</tr>
</tbody>
</table>

Usage

The Metadata API can be used to create a self-signed or a CA-signed certificate. The .crt file's contents are the certificate chain, which can be updated when you renew or update the intermediate certificate chain of a CA-signed certificate. After creating a CA-signed
To copy an existing certificate's X.509 parameter data to a new certificate, upload the existing certificate. You can also use this procedure to renew a certificate. A new private+public key pair is created with a new certificate. Salesforce doesn’t allow the import or export of the private key via the API.

Using the Metadata API, you can download a CSR. After it's CA-signed, you can upload it back to Salesforce.

- **Downloading a CSR.** The CSR is downloadable after a CA-signed cert is created. If a signed certificate hasn’t been uploaded to that certificate, the content of the downloaded .crt file is the CSR.
- **Uploading a CA-Signed Certificate.** To upload the signed certificate chain back to Salesforce, save the signed certificate chain as the content of the .crt file and update it via the Metadata API.

Note: After the signed certificate chain is uploaded via the Metadata API, the CSR of that certificate can’t be downloaded via the API anymore. This is because the content of the .crt file is the signed certificate chain. However, the CSR can still be downloaded via the UI.

**Declarative Metadata Sample Definition**

The following is an example of a Certificate component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Certificate xmlns="http://soap.sforce.com/2006/04/metadata">
  <caSigned>true</caSigned>
  <encryptedWithPlatformEncryption>true</encryptedWithPlatformEncryption>
  <expirationDate>2017-03-19</expirationDate>
  <keySize>4096</keySize>
  <masterLabel>My Certificate Name</masterLabel>
  <privateKeyExportable>true</privateKeyExportable>
</Certificate>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**ChatterExtension**

 Represents the metadata used to describe a Rich Publisher App that’s integrated with the Chatter publisher.

**Retrieving ChatterExtension**

Using Workbench or another API tool, you can get extension information from package.xml using this code.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>xw1</members>
    <name>ChatterExtension</name>
  </types>
</Package>
```
Use the <members> tag to name a specific extension (in this example, xw1), or use the wildcard (*) symbol to retrieve all your extensions.

Here’s an example of retrieved information.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ChatterExtension xmlns="http://soap.sforce.com/2006/04/metadata">
  <compositionComponent>xwComp</compositionComponent>
  <description>des</description>
  <extensionName>xw1</extensionName>
  <headerText>h1</headerText>
  <hoverText>h2</hoverText>
  <icon>tiger</icon>
  <masterLabel>master</masterLabel>
  <renderComponent>xwRend</renderComponent>
  <type>Lightning</type>
</ChatterExtension>
```

**Version**

ChatterExtension is a new feature in API version 41.0.

**Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>compositionComponent</td>
<td>string</td>
<td>Required. The composition component of the Rich Publisher App that you provide. It’s comprised of the lightning:availableForChatterExtensionComposer interface.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Required. The description of your custom Rich Publisher App.</td>
</tr>
<tr>
<td>extensionName</td>
<td>string</td>
<td>Required. The name of your extension. That is, your Rich Publisher App.</td>
</tr>
<tr>
<td>headerText</td>
<td>string</td>
<td>The text to show in the header of your app composer. This is required for Lightning type extensions. This text can be localized.</td>
</tr>
<tr>
<td>hoverText</td>
<td>string</td>
<td>The text to show when a user mouses over your extension’s icon. This is required for Lightning type extensions. This text can be localized.</td>
</tr>
<tr>
<td>icon</td>
<td>string</td>
<td>Required. The icon to show in the Chatter publisher. Use an existing file asset id from your org.</td>
</tr>
<tr>
<td>isProtected</td>
<td>boolean</td>
<td>An auto-generated value. It currently has no impact.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Master label for the ChatterExtension object.</td>
</tr>
</tbody>
</table>
**Field** | **Field Type** | **Description**
--- | --- | ---
renderComponent | string | Required. The rendering component of the Rich Publisher App that you provide. It’s comprised of the `lightning:availableForChatterExtensionRenderer` interface.
type | ChatterExtensionType (enumeration of type string) | Required. Describes the type of the extension. Currently, the only value supported is `Lightning`. Included to allow for other possible types in the future.

**SEE ALSO:**
[Integrate Your Custom Apps into the Chatter Publisher](#)

## CleanDataService

Represents a data service that adds and updates data in standard objects. This type extends the Metadata metadata type and inherits its `fullName` field.

### File Suffix and Directory Location

CleanDataService components have the `.cleanDataService` suffix and are stored in the `cleanDataServices` directory. The name of the component file is based on the name of the object associated with the data service. For example, the component file name `cleanDataServices/DataCloudCompanyMatch.cleanDataService` describes a data service component called `DataCloudCompanyMatch` that's associated with the company name in account objects.

### Version

CleanDataService components are available in API version 46.0 and later.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cleanRules</td>
<td><code>CleanRule[]</code></td>
<td>Required. A list of data integration rules</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Required. A description of the data service</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Master label for this data service. Although this value is displayed, it’s an internal label for the data service and isn’t translated.</td>
</tr>
<tr>
<td>matchEngine</td>
<td>string</td>
<td>Required. A key that maps to the internal data service identifier.</td>
</tr>
</tbody>
</table>

## CleanRule

Represents information that controls how the data service adds and updates data in an org.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bulkEnabled</td>
<td>boolean</td>
<td>Required. If this field is set to true, Salesforce applies the data integration rule to existing records whenever the rule is updated or saved.</td>
</tr>
<tr>
<td>bypassTriggers</td>
<td>boolean</td>
<td>Required. If this field is set to true, Salesforce bypasses triggers when it applies the rule; otherwise, it applies triggers after it applies the rule.</td>
</tr>
<tr>
<td>bypassWorkflow</td>
<td>boolean</td>
<td>Required. If this field is set to true, Salesforce bypasses workflow rules when it applies the data integration rule; otherwise, it applies workflow rules after it applies the rule.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Required. User-friendly text that describes the data integration rule.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. This name can contain only underscores and alphanumeric characters, and must be unique in your org. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This unique name prevents conflicts with rules from other packages that have the same MasterLabel.</td>
</tr>
<tr>
<td>fieldMappings</td>
<td>FieldMapping[]</td>
<td>Required. A list of FieldMapping entries for the rule.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Master label for this object. This display value is the internal label that is not translated.</td>
</tr>
<tr>
<td>matchRule</td>
<td>string</td>
<td>Required. An internal label for a matching rule in the data service that’s associated with the CleanRule.</td>
</tr>
<tr>
<td>sourceSobjectType</td>
<td>string</td>
<td>Required. A virtual object in the data service that is associated with the CleanRule. Specifying a non-existent object causes an error.</td>
</tr>
<tr>
<td>status</td>
<td>string</td>
<td>Required. Status of the data integration rule. Valid values are Active and Inactive.</td>
</tr>
<tr>
<td>targetSobjectType</td>
<td>string</td>
<td>Required. A standard object that’s the target of additions and updates specified by this CleanRule. Specifying an object that the data service does not support causes an error.</td>
</tr>
</tbody>
</table>

**FieldMapping**

Represents a mapping between fields in the data service and fields in an object in the org.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. This name can contain only underscores and alphanumeric characters, and must be unique in your org. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This unique name prevents conflicts with field mappings from other packages that have the same MasterLabel.</td>
</tr>
<tr>
<td>fieldMappingRows</td>
<td>FieldMappingRow[]</td>
<td>Required. A list of FieldMappingRow entries. Each entry represents a field in a standard object that maps to a field in the data service.</td>
</tr>
</tbody>
</table>
### FieldMappingRow

Represents the status of a CleanRule.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>string</td>
<td>The display name for the field represented by the FieldMappingRow.</td>
</tr>
<tr>
<td>fieldMappingFields</td>
<td>FieldMappingField[]</td>
<td>Required. A list of FieldMappingField entries. Each entry is a field in a standard object that maps to a field in the data service.</td>
</tr>
<tr>
<td>mappingOperation</td>
<td>string</td>
<td>The comparison operation the data service applies when it compares the value of this FieldMappingRow to the mapped field in the object specified in SObjectType. The value of this field is AutoFill, which indicates that the data service only adds data if the object field is blank.</td>
</tr>
<tr>
<td>SObjectType</td>
<td>string</td>
<td>The standard object for the field mapped to the FieldMappingRow. Specifying an object that the data service does not support causes an error.</td>
</tr>
</tbody>
</table>

### FieldMappingField

Represents a field in a standard object. A FieldMappingField maps to a FieldMappingRow entry in a data service.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataServiceField</td>
<td>string</td>
<td>Required. A field in the data service that is mapped to this field.</td>
</tr>
<tr>
<td>dataServiceObjectName</td>
<td>string</td>
<td>Required. An object in the data service that contains the FieldMappingRow associated with this FieldMappingField. Specifying a non-existent object causes an error.</td>
</tr>
<tr>
<td>priority</td>
<td>int</td>
<td>Required. Represents the priority that the data service uses when it updates the field, relative to other update rules for the same field. Valid values are 1-100.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of a CleanDataService component for the lead standard object.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CleanDataService xmlns="http://soap.sforce.com/2006/04/metadata">
  <cleanRules>
    <bulkEnabled>false</bulkEnabled>
    <bypassTriggers>false</bypassTriggers>
  </cleanRules>
</CleanDataService>
```
<bypassWorkflow>false</bypassWorkflow>
<description>Adds data info to leads</description>
<developerName>DataService_Leads_Enrichment</developerName>

<fieldMappings>
  <SObjectType>DataServiceCompanyObject</SObjectType>
  <developerName>DataService_Leads_Enrichment_InputMapping</developerName>
  <fieldMappingRows>
    <SObjectType>DataServiceCompanyObject</SObjectType>
    <fieldMappingFields>
      <dataServiceField>Email</dataServiceField>
      <dataServiceObjectName>Lead</dataServiceObjectName>
      <priority>1</priority>
    </fieldMappingFields>
    <fieldName>Email</fieldName>
    <mappingOperation>Autofill</mappingOperation>
  </fieldMappingRows>
  <fieldMappingRows>
    <SObjectType>DataServiceCompanyObject</SObjectType>
    <fieldMappingFields>
      <dataServiceField>Company</dataServiceField>
      <dataServiceObjectName>Lead</dataServiceObjectName>
      <priority>1</priority>
    </fieldMappingFields>
    <fieldName>Name</fieldName>
    <mappingOperation>Autofill</mappingOperation>
  </fieldMappingRows>
  <masterLabel>DataServiceInputMapping</masterLabel>
</fieldMappings>

<fieldMappings>
  <SObjectType>Lead</SObjectType>
  <developerName>DataService_Leads_Enrichment_OutputMapping</developerName>
  <fieldMappingRows>
    <SObjectType>Lead</SObjectType>
    <fieldMappingFields>
      <dataServiceField>EmployeesTotal</dataServiceField>
      <dataServiceObjectName>DataServiceCompanyObject</dataServiceObjectName>
      <priority>1</priority>
    </fieldMappingFields>
    <fieldName>NumberOfEmployees</fieldName>
    <mappingOperation>Autofill</mappingOperation>
  </fieldMappingRows>
  <fieldMappingRows>
    <SObjectType>Lead</SObjectType>
    <fieldMappingFields>
      <dataServiceField>Revenue</dataServiceField>
      <dataServiceObjectName>DataServiceCompanyObject</dataServiceObjectName>
      <priority>1</priority>
    </fieldMappingFields>
    <fieldName>AnnualRevenue</fieldName>
    <mappingOperation>Autofill</mappingOperation>
  </fieldMappingRows>
</fieldMappings>
Usage

Use CleanDataService to retrieve all the metadata that describes a data enrichment service. To configure the service in a new org, deploy the metadata you retrieved. Avoid using CRUD-Based Calls with CleanDataService.

To make small modifications to the CleanDataService component, use the Tooling API.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

CMSConnectSource

Represents the connection information for external content management systems that feed content to Communities. This type extends the Metadata metadata type and inherits its fullName field.

Note: For use with Change Sets, CMSConnectSource is a dependent of Network and Community.
File Suffix and Directory Location

CMSConnectSource components have the suffix .cmsConnectSource and are stored in the cmsConnectSource folder. In that folder, separate files exist for each network (for example, networkname.sourcedevelopername.cmsConnectSource). Each file represents a CMS connection.

Version

CMSConnectSource components are available in API version 43.0 and later.

Special Access Rules

The CMS Connect org permission must be enabled.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmsConnectAsset</td>
<td>CMSConnectAsset on page 220</td>
<td>Represents CSS or JavaScript defined for the connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0–10 for CSS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0–10 for JavaScript</td>
</tr>
<tr>
<td>cmsConnectLanguage</td>
<td>CMSConnectLanguage on page 221</td>
<td>0 to more. Represents language mappings defined for the connection.</td>
</tr>
<tr>
<td>cmsConnectPersonalization</td>
<td>CMSConnectPersonalization[] on page 221</td>
<td>0 or 1. This represents personalization defined for the connection. Only for use when type is AEM.</td>
</tr>
<tr>
<td>cmsConnectResourceType</td>
<td>CMSConnectResourceType on page 221</td>
<td>0–5. This represents JSON definitions defined for the connection.</td>
</tr>
<tr>
<td>connectionType</td>
<td>CMSSourceConnectionType of type string</td>
<td>Required. Type of authentication being used with outside system. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Public</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Authenticated</td>
</tr>
<tr>
<td>cssScope</td>
<td>string</td>
<td>The class name used to prefix and scope the CSS rules.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. API name of the CMSConnectSource entity.</td>
</tr>
<tr>
<td>languageEnabled</td>
<td>string</td>
<td>Required. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Y to enable language mapping for connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• N if no language mapping is needed.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Connection name</td>
</tr>
<tr>
<td>namedCredential</td>
<td>string</td>
<td>Required when the connectionType is Authenticated. API name of namedCredential. Before deploying namedCredential, it must exist on the destination org.</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
personalizationEnabled | string | Required. Valid values are:
  - `Y` to enable personalization mapping for connection.
  - Otherwise `N`.
rootPath | string | Required. Root path.
sortOrder | int | Required. Defines the load order of the connection when multiple connections defined on page. The load order begins with 1.
status | CMSConnectionStatus (enumeration of type string) | Required. Status of connection. Valid values are:
  - ACTIVE
  - INACTIVE
type | CMSConnectionSourceType (enumeration of type string) | Required. The identification of the source connection system. Valid values are:
  - AEM
  - Drupal
  - WordPress
  - SDL
  - Sitecore
  - Other
websiteUrl | string | Required if connectionType is Public

**Note:** Because there may be existing connections when a package comes in, there is some INSERT or UPDATE logic that should be taken into account:

- If `developerName` is found in the destination, then update the existing collection with all details form source.
- If `namedCredential` is handled through `developerName`. If `namedCredential` with `developerName` is not found, then an error is generated.
- If `sortOrder` from the source is not in the destination, then insert/update with the source `sortOrder`.
- If `sortOrder` from the source is already in the destination, then increase the source `sortOrder` by 1 for connections such that the destination `sortOrder` > `sortOrder` from the source.

### CMSConnectAsset

CMSConnectAsset defines the location, types, and order of assets necessary to support the incoming content, such as JavaScript and CSS files.

**Note:** Because there may be existing connections when a package comes in, there is some INSERT or UPDATE logic that should be taken into account:

- If `assetPath` exists in the destination, then update the existing record, else the new `assetPath` will be inserted.
- Always keep the `sortOrder` from the source and adjust the destination accordingly.
**CMSConnectSource**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assetPath</td>
<td>string</td>
<td>Relative path of the asset.</td>
</tr>
</tbody>
</table>
| assetType    | string     | When used in Apex, this can be sent as an enum, otherwise, this has a field type of string. Allowed values as string
  • CSS
  • Javascript Allowed values as enum
  • CSS
  • Javascript |
| sortOrder    | int        | Loading sequence on the page.                                               |

**CMSConnectLanguage**

CMSConnectLanguage components determine the presented language of the content.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmsLanguage</td>
<td>string</td>
<td>When a language placeholder is in the URL path, this value is used to replace it.</td>
</tr>
<tr>
<td>language</td>
<td>string</td>
<td>Salesforce supported language.</td>
</tr>
</tbody>
</table>

**CMSConnectPersonalization**

CMSConnectPersonalization is only used in conjunction with Adobe Experience Manager (AEM).

Note: Because there may be existing connections when a package comes in, there is some INSERT or UPDATE logic that should be taken into account. If personalization is not enabled in the source system, but it is enabled in the destination, the destination is disabled. The record for the connection is deleted from the table.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectorPage</td>
<td>string</td>
<td>The path to the JSP file that you created and installed in AEM.</td>
</tr>
<tr>
<td>connectorPageAsset</td>
<td>string</td>
<td>The path to your Javascript file. Providing this path allows you to run scripts dynamically.</td>
</tr>
</tbody>
</table>

**CMSConnectResourceType**

CMSConnectResourceType is for use only to define JSON connections.
Note: Because there may be existing connections when a package comes in, there is some INSERT or UPDATE logic that should be taken into account. If the developer name was found in the destination, then update the existing record with all details from the source.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmsConnectResourceDefinition</td>
<td>cmsConnectResourceDefinition</td>
<td>0–10 allowed per CMSConnectResourceType.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>API name of CMSConnectResourceType.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Content type name.</td>
</tr>
<tr>
<td>resourceType</td>
<td>string</td>
<td>The only allowed value is JSON.</td>
</tr>
</tbody>
</table>

**CMSConnectResourceDefinition**

cmsConnectResourceDefinition is used to define JSON connections.

Note: Because there may be existing connections when a package comes in, there is some INSERT or UPDATE logic that should be taken into account:

- If developerName is found in the destination, then the existing record is updated with all details from the new source, else the new value is inserted.
- If the current source is DETAIL and the destination has DETAIL with a different name, then the destination is updated to LIST and the source inserted as DETAIL.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. API name of CMSConnectResourceDefinition.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. developerName of Content Item or Content List.</td>
</tr>
<tr>
<td>options</td>
<td>int</td>
<td>Required. Identifies whether the content from the external source is a single item or a list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 for Content List</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 for Content Item</td>
</tr>
<tr>
<td>payloadType</td>
<td>string</td>
<td>Required. The only valid value is JSON.</td>
</tr>
<tr>
<td>resourceIdPath</td>
<td>string</td>
<td>Relative path to ID. Required for Content Item.</td>
</tr>
<tr>
<td>resourceNamePath</td>
<td>string</td>
<td>Relative path to resource name. Required for Content Item.</td>
</tr>
<tr>
<td>resourcePath</td>
<td>string</td>
<td>Required. JSON resource path.</td>
</tr>
<tr>
<td>rootNodePath</td>
<td>string</td>
<td>Only for Content List and collection. Defines the initial starting path for a collection or list.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example of a CMSConnectSource definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CMSConnectSource xmlns="http://soap.sforce.com/2006/04/metadata">
  <cmsConnectAsset>
    <assetPath>etc/designs/capricorn/app-prefixed.min.css</assetPath>
    <assetType>CSS</assetType>
    <sortOrder>1</sortOrder>
  </cmsConnectAsset>
  <cmsConnectAsset>
    <assetPath>etc/designs/capricorn/w3data.js</assetPath>
    <assetType>Javascript</assetType>
    <sortOrder>1</sortOrder>
  </cmsConnectAsset>
  <cmsConnectLanguage>
    <cmsLanguage>en</cmsLanguage>
    <language>en_US</language>
  </cmsConnectLanguage>
  <cmsConnectLanguage>
    <cmsLanguage>fr</cmsLanguage>
    <language>fr</language>
  </cmsConnectLanguage>
  <cmsConnectPersonalization>
    <connectorPage>content.salesforceConnector.js</connectorPage>
    <connectorPageAsset>content/js/capricorn/assets.js</connectorPageAsset>
  </cmsConnectPersonalization>
  <cmsConnectResourceType>
    <cmsConnectResourceDefinition>
      <developerName>Details</developerName>
      <masterLabel>Details</masterLabel>
      <options>0</options>
      <payloadType>JSON</payloadType>
      <resourceIdPath>ID</resourceIdPath>
      <resourceNamePath>title</resourceNamePath>
      <resourcePath>rest/v1.1/sites/cmstry.wordpress.com/posts/{component}</resourcePath>
    </cmsConnectResourceDefinition>
    <cmsConnectResourceDefinition>
      <developerName>List</developerName>
      <masterLabel>List</masterLabel>
      <options>1</options>
      <payloadType>JSON</payloadType>
      <resourceIdPath>ID</resourceIdPath>
      <resourceNamePath>title</resourceNamePath>
      <resourcePath>rest/v1.1/sites/cmstry.wordpress.com/posts?number={itemsPerPage}&amp;page={pageNumber}</resourcePath>
    </cmsConnectResourceDefinition>
    <developerName>Posts</developerName>
    <masterLabel>Posts</masterLabel>
    <resourceType>JSON</resourceType>
  </cmsConnectResourceType>
  <connectionType>Public</connectionType>
  <cssScope>capricorn</cssScope>
  <developerName>Capricorn</developerName>
</CMSConnectSource>
```
The following is an example package.xml.

```xml
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>NetworkA.*</members>
    <name>CMSConnectSource</name>
  </types>
  <version>43.0</version>
</Package>
```

To retrieve a specific connection:

```xml
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>NetworkA.DeveloperName</members>
    <name>CMSConnectSource</name>
  </types>
  <version>43.0</version>
</Package>
```

**Usage**

The INSERT or UPDATE logic for the incoming information is always UPSERT. If data is not in the entity, then it is inserted, otherwise the current data is updated.

Before doing upsert, the content from the package is validated against the maximum limits for the following:

- CSS assets \(\leq 10\)
- JavaScript assets \(\leq 10\)
- Resource types \(\leq 5\)
- Resource definitions for each type \(\leq 10\)

For example

1. The validation on a new connection only totals the elements in the incoming package.
2. Validation of existing connections totals the existing assets and new elements to assess validity. For example, if a connection on the destination org already has six CSS definitions, and the incoming package has defined seven CSS definitions (four new + three existing). The new total will be the six current from the database, ignoring the three repeated in the package, and adds four new definitions from the incoming package, totaling 10 definitions, which is at or below the 10 asset threshold and it passes validation.

Refer below for more details for each entity how is handled while saving the details from package to destination org:
CMSConnectSource

- If `developerName` is found in the destination, then update the existing collection with all details from source.
- `namedCredential` is handled through `developerName`. If `namedCredential` with `developerName` is not found, then an error is generated.
- If `sortOrder` from the source is not in the destination, then insert/update with the source `sortOrder`.
- If `sortOrder` from the source is already in the destination, then increase the source `sortOrder` by 1 for connections such that the destination `sortOrder` > `sortOrder` from the source.

CMSConnectAsset

- If `assetPath` exists in the destination, then update the existing record, else the new `assetPath` will be inserted.
- Always keep the `sortOrder` from the source and adjust the destination accordingly.

CMSConnectPersonalization

If personalization is not enabled in the source system, but it is enabled in the destination, the destination is disabled. The record for the connection is deleted from the table.

CMSConnectResourceType

If the developer name was found in the destination, then update the existing record with all details from the source.

CMSConnectResourceDefinition

- If `developerName` is found in the destination, then the existing record is updated with all details from the new source, else the new value is inserted.
- If the current source is DETAIL and the destination has DETAIL with a different name, then the destination is updated to LIST and the source inserted as DETAIL.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- Select Components for an Outbound Change Set
- View and Add Dependent Components to a Change Set
- Developer Guide: Deploying and Retrieving Metadata
- Salesforce Help: Use Personalized Content in CMS Connect
- Developer Guide: Translations
Community (Zone)

Represents a zone that contains Ideas or Chatter Answers objects. Zones are shared by the Ideas, Answers, and Chatter Answers features, allowing you to view and create zones from those locations. This type extends the Metadata metadata type and inherits its fullName field.

Note: Starting with the Summer '13 release, Chatter Answers and Ideas “communities” have been renamed to “zones.” In API version 28, the API object label has changed to Zone, but the API type is still Community.

File Suffix and Directory Location

Zones have the suffix community and are stored in the communities folder.

Version

Community (Zone) components are available in API version 27.0 and later.

Fields

Note: When enableChatterAnswers is set to false, values specified for the following fields are ignored and not saved: communityFeedPage, emailFooterDocument, emailHeaderDocument, enablePrivateQuestions, emailNotificationUrl, and site.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Indicates whether the zone is active (true) or not (false).</td>
</tr>
<tr>
<td>chatterAnswersFacebookSsoUrl</td>
<td>string</td>
<td>(Read only) The Facebook sign-on URL, which is based on the Facebook authentication provider selected in your Chatter Answers settings. This field is available only if Chatter Answers and Facebook Single Sign-On for Chatter Answers are enabled.</td>
</tr>
<tr>
<td>communityFeedPage</td>
<td>string</td>
<td>The Visualforce page that hosts the zone's feeds. This field is available when Chatter Answers is enabled in the organization.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the zone.</td>
</tr>
<tr>
<td>emailFooterDocument</td>
<td>string</td>
<td>The text or HTML file that incorporates your organization’s branding into the footer of email notifications. This field is available when Chatter Answers is enabled in the organization.</td>
</tr>
<tr>
<td>emailHeaderDocument</td>
<td>string</td>
<td>The text or HTML file that incorporates your organization’s branding into the header of email notifications. This field is available when Chatter Answers is enabled in the organization.</td>
</tr>
<tr>
<td>emailNotificationUrl</td>
<td>string</td>
<td>The URL that’s included in email notifications. This field is available when Chatter Answers is enabled in the organization. This field replaces portalEmailNotificationUrl in API version 28.0 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>enableChatterAnswers</td>
<td>boolean</td>
<td>Indicates whether the zone has Chatter Answers enabled (true) or not (false). This field is available when Chatter Answers is enabled in the organization.</td>
</tr>
<tr>
<td>enablePrivateQuestions</td>
<td>boolean</td>
<td>Indicates whether Chatter Answers questions can be escalated to cases (true) or not (false). This field is available when Chatter Answers is enabled in the organization.</td>
</tr>
<tr>
<td>expertsGroup</td>
<td>string</td>
<td>The name of the public group that act as experts in the zone. This field is available when either Ideas or Answers are enabled in the organization.</td>
</tr>
<tr>
<td>portal</td>
<td>string</td>
<td>The name of the portal in which to display the zone.</td>
</tr>
<tr>
<td>portalEmailNotificationUrl</td>
<td>string</td>
<td>The portal URL that's included in email notifications. This field is available when Chatter Answers is enabled in the organization. This field has been replaced by <code>emailNotificationUrl</code> in API version 28.0 and later.</td>
</tr>
<tr>
<td>reputationLevels</td>
<td>ReputationLevels</td>
<td>The fields that define the points and name of each reputation level you define. You can create up to 25 reputation levels per zone.</td>
</tr>
<tr>
<td>showInPortal</td>
<td>boolean</td>
<td>Indicates whether the zone is available to all portals (true) or not available to any portals (false).</td>
</tr>
<tr>
<td>site</td>
<td>string</td>
<td>The name of the site for the zone. This field is available when Chatter Answers is enabled in the organization.</td>
</tr>
</tbody>
</table>

### ReputationLevels

Represents the points and reputation label that displays on hover over a user’s photo in the feed.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chatterAnswersReputationLevels</td>
<td>ChatterAnswersReputationLevel</td>
<td>Contains the name and value pair that describes the reputation level for Chatter Answers. Available in API version 28.0 and later.</td>
</tr>
<tr>
<td>ideaReputationLevels</td>
<td>IdeaReputationLevel</td>
<td>Contains the name and value pair that describes the reputation for Ideas. Available in API version 28.0 and later.</td>
</tr>
</tbody>
</table>

### ChatterAnswersReputationLevel

Represents the reputation name and the number of points for that level for Chatter Answers.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the reputation level, for example, &quot;Expert.&quot;</td>
</tr>
<tr>
<td>value</td>
<td>int</td>
<td>The minimum number of points for the reputation level.</td>
</tr>
</tbody>
</table>
IdeaReputationLevel

Represents the reputation name and the number of points for that level for Ideas. Available in API version 28.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the reputation level, for example, “Expert.”</td>
</tr>
<tr>
<td>value</td>
<td>int</td>
<td>The minimum number of points for the reputation level.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is the definition of a community (zone) component:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Community xmlns="http://soap.sforce.com/2006/04/metadata">
  <active>true</active>
  <communityFeedPage>communityWithHeaderAndFooter_main</communityFeedPage>
  <description>Metadata Test</description>
  <emailFooterDocument>sampleFolder/emailFooter.html</emailFooterDocument>
  <emailHeaderDocument>sampleFolder/emailHeader.html</emailHeaderDocument>
  <enableChatterAnswers>true</enableChatterAnswers>
  <enablePrivateQuestions>true</enablePrivateQuestions>
  <expertsGroup>CommunityExperts</expertsGroup>
  <portal>Customer Portal</portal>
  <emailNotificationUrl>http://yourURL</emailNotificationUrl>
  <reputationLevels>
    <chatterAnswersReputationLevels>
      <name>Newbie</name>
      <value>0</value>
    </chatterAnswersReputationLevels>
    <chatterAnswersReputationLevels>
      <name>Smartie</name>
      <value>500</value>
    </chatterAnswersReputationLevels>
    <chatterAnswersReputationLevels>
      <name>Pro</name>
      <value>2000</value>
    </chatterAnswersReputationLevels>
    <chatterAnswersReputationLevels>
      <name>All Star</name>
      <value>5000</value>
    </chatterAnswersReputationLevels>
    <ideaReputationLevels>
      <name>Observer</name>
      <value>0</value>
    </ideaReputationLevels>
    <ideaReputationLevels>
      <name>Contributor</name>
      <value>100</value>
    </ideaReputationLevels>
    <ideaReputationLevels>
      <name>Influencer</name>
    </ideaReputationLevels>
  </reputationLevels>
</Community>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

CommunityTemplateDefinition

Represents the definition of a community template. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

CommunityTemplateDefinition components have the suffix .communityTemplateDefinition and are stored in the communityTemplateDefinitions folder.

Version

CommunityTemplateDefinition components are available in API version 38.0 and later.

Special Access Rules

This type is available only if Salesforce Communities is enabled in your org.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseTemplate</td>
<td>CommunityBase Template (enumeration of type string)</td>
<td>Denotes that this CommunityTemplateDefinition was created in API version 41.0 or later. The only valid value is c. This field is available in API 41.0 and later.</td>
</tr>
<tr>
<td>bundlesInfo</td>
<td>CommunityTemplateBundleInfo []</td>
<td>The list of preview images and feature highlights of this CommunityTemplateDefinition.</td>
</tr>
</tbody>
</table>
## CommunityTemplateDefinition Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>category</td>
<td>CommunityTemplateCategory (enumeration of type string)</td>
<td>Required. The optimized use case of this CommunityTemplateDefinition. Valid values are: IT, Marketing, Sales, Service</td>
</tr>
<tr>
<td>defaultBrandingSet</td>
<td>string</td>
<td>The set of branding properties associated with this CommunityTemplateDefinition, as defined in the Branding panel in Experience Builder. Available in API version 40.0 and later. In API version 44.0 and later, this field is read-only and can be implemented in CommunityThemeDefinition on page 235.</td>
</tr>
<tr>
<td>defaultThemeDefinition</td>
<td>string</td>
<td>Required. The assigned theme definition for this CommunityTemplateDefinition.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The optional description text of this CommunityTemplateDefinition.</td>
</tr>
<tr>
<td>enableExtendedCleanUpOnDelete</td>
<td>boolean</td>
<td>False by default. Determines if deleting this CommunityTemplateDefinition attempts to delete other directly or indirectly referenced objects automatically, for example, CommunityThemeDefinition on page 235, Flexipage on page 477, or StaticResource on page 876. Values are true or false.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The label for this CommunityTemplateDefinition, which displays in Setup.</td>
</tr>
<tr>
<td>navigationLinkSet</td>
<td>NavigationLinkSet</td>
<td>The navigation menu associated with this CommunityTemplateDefinition. A navigation menu consists of items that users can click to go to other parts of the community. Available in API version 40.0 and later.</td>
</tr>
<tr>
<td>pageSetting</td>
<td>CommunityTemplatePageSetting []</td>
<td>The list of FlexiPages of this CommunityTemplateDefinition.</td>
</tr>
<tr>
<td>publisher</td>
<td>string</td>
<td>Defines the name of the publisher as seen in the Change Theme wizard. If no name is provided, the name of the org from which the package was originally exported is used. This field is available in API version 45.0 and later.</td>
</tr>
</tbody>
</table>

### CommunityTemplateBundleInfo

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The optional description text of its CommunityTemplateBundleInfo.</td>
</tr>
<tr>
<td>image</td>
<td>string</td>
<td>Required only when the type is PreviewImage, otherwise this field is optional. A preview image for this CommunityTemplateDefinition.</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
---|---|---
order | int | Required. An integer specifying the position of this CommunityTemplateBundleInfo relative to others of the same type within its CommunityTemplateDefinition. 1 is the first position, 3 is the maximum position for PreviewImage type, and 4 is the maximum position for the Highlight type.
title | string | Required. The title of this CommunityTemplateBundleInfo to use in code.
type | CommunityTemplateBundleInfoType (enumeration of type string) | Required. Stores descriptive information about the template that is included in the export. The template powers the interface of the Community Creation Wizard. Valid values are:
- **Highlight**—This CommunityTemplateBundleInfo is used as a highlighted feature. Up to 4 are supported.
- **PreviewImage**—This CommunityTemplateBundleInfo is used as a preview image. Up to 3 are supported.

### CommunityTemplatePageSetting

| Field Name | Field Type | Description |
---|---|---|
page | string | Required. The list of FlexiPages of this CommunityTemplateDefinition. |
themeLayout | string | Required. The name of the FlexiPage for the theme layout. This field is available in API version 39.0 and later.

### Declarative Metadata Sample Definition

The following is an example of a CommunityTemplateDefinition component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CommunityTemplateDefinition xmlns="http://soap.sforce.com/2006/04/metadata">
  <baseTemplate>c</baseTemplate>
  <bundlesInfo>
    <description>Feature Description</description>
    <order>1</order>
    <title>Feature Heading</title>
    <type>Highlight</type>
  </bundlesInfo>
  <bundlesInfo>
    <image>siteAsset_2dbe594eb6794173af78da264cd6a4a7</image>
    <order>1</order>
    <title>Preview Image</title>
    <type>PreviewImage</type>
  </bundlesInfo>
  <category>Sales</category>
  <defaultThemeDefinition>communityTemplate</defaultThemeDefinition>
  <description>This is a lightning community template</description>
</CommunityTemplateDefinition>
```
</pageSetting>
<pageSetting>
  <page>communityTemplate_User_Profile</page>
  <themeLayout>communityTemplate_themeLayout_Default</themeLayout>
</pageSetting>
<pageSetting>
  <page>communityTemplate_Case_Detail</page>
  <themeLayout>communityTemplate_themeLayout_Default</themeLayout>
</pageSetting>
<pageSetting>
  <page>communityTemplate_Stream_Related_List</page>
  <themeLayout>communityTemplate_themeLayout_Default</themeLayout>
</pageSetting>
<pageSetting>
  <page>communityTemplate_Dashboard_Detail</page>
  <themeLayout>communityTemplate_themeLayout_Default</themeLayout>
</pageSetting>
<pageSetting>
  <page>communityTemplate_Group_List</page>
  <themeLayout>communityTemplate_themeLayout_Default</themeLayout>
</pageSetting>
<pageSetting>
  <page>communityTemplate_Canvasapp_Page</page>
  <themeLayout>communityTemplate_themeLayout_Default</themeLayout>
</pageSetting>
<pageSetting>
  <page>communityTemplate_Case_List</page>
  <themeLayout>communityTemplate_themeLayout_Default</themeLayout>
</pageSetting>
<pageSetting>
  <page>communityTemplate_User_List</page>
  <themeLayout>communityTemplate_themeLayout_Default</themeLayout>
</pageSetting>
<pageSetting>

233
The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>MyTemplate</members>
    <name>CommunityTemplateDefinition</name>
  </types>
  <version>46.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**CommunityThemeDefinition**

Represents the definition of a community theme. This type extends the `Metadata` metadata type and inherits its `fullName` field.

**File Suffix and Directory Location**

`CommunityThemeDefinition` components have the suffix `.communityThemeDefinition` and are stored in the `communityThemeDefinitions` folder.

**Version**

`CommunityThemeDefinition` components are available in API version 38.0 and later.
Special Access Rules
This type is available only if Salesforce Communities is enabled in your org.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bundlesInfo</td>
<td>CommunityThemeBundleInfo</td>
<td>If specified, at least one preview image and one highlight are required. Up to 3 preview images and 4 highlights are supported. Available in API version 44.0 and later.</td>
</tr>
<tr>
<td>customThemeLayoutType</td>
<td>CommunityThemeLayoutType</td>
<td>The list of custom theme layout types available to the theme layout.</td>
</tr>
<tr>
<td>defaultBrandingSet</td>
<td>string</td>
<td>The set of branding properties associated with this CommunityThemeDefinition, as defined in the Branding panel in Experience Builder. Available in API version 44.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The optional description text of this CommunityThemeDefinition.</td>
</tr>
<tr>
<td>enableExtendedCleanUpOnDelete</td>
<td>boolean</td>
<td>False by default. Determines if deleting this CommunityThemeDefinition attempts to delete other directly or indirectly referenced objects automatically, for example, FlexiPage. Values are true or false.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The label for this CommunityThemeDefinition, which displays in Setup.</td>
</tr>
<tr>
<td>publisher</td>
<td>string</td>
<td>Defines the name of the publisher as seen in the Community Creation wizard. If no name is provided, the name of the org from which the package was originally exported is used. This field is available in API version 45.0 and later.</td>
</tr>
<tr>
<td>themeRouteOverride</td>
<td>CommunityThemeRouteOverride</td>
<td>List of theme layout type overrides for flexipages (currently only for home). Available in API version 44.0 and later.</td>
</tr>
<tr>
<td>themeSetting</td>
<td>CommunityThemeSetting</td>
<td>Required. The list of settings for this CommunityThemeDefinition.</td>
</tr>
</tbody>
</table>

CommunityThemeBundleInfo

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The optional description text of its CommunityThemeBundleInfo.</td>
</tr>
<tr>
<td>image</td>
<td>string</td>
<td>Required only when the type is PreviewImage, otherwise this field is optional. A preview image for this CommunityThemeDefinition.</td>
</tr>
</tbody>
</table>
| order              | int        | Required. An integer specifying the position of this CommunityThemeBundleInfo relative to others of the same type within its CommunityThemeDefinition. 1 is the first position, 3 is the maximum position.
### CommunityThemeDefinitionMetadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>string</td>
<td>Required. The title of this CommunityThemeBundleInfo to use in code.</td>
</tr>
</tbody>
</table>
| type          | CommunityTemplateBundleInfoType (enumeration of type string) | Required. Stores descriptive information about the theme that is included in the export. Valid values are:  
  - Highlight—This CommunityThemeBundleInfo is used as a highlighted feature. Up to 4 are supported.  
  - PreviewImage—This CommunityThemeBundleInfo is used as a preview image. Up to 3 are supported. |

#### CommunityCustomThemeLayoutType

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the custom theme layout type.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The name of the custom theme layout type. The values Inner, Home, and Login are reserved.</td>
</tr>
</tbody>
</table>

#### CommunityThemeRouteOverride

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customThemeLayoutType</td>
<td>string</td>
<td>Required when themeLayoutType is not specified. Provides the custom theme layout type associated with the theme layout. This field and themeLayoutType are mutually exclusive; you can't specify both.</td>
</tr>
<tr>
<td>pageAttributes</td>
<td>string</td>
<td>Required. Specifies the attributes of the community page for which the default theme layout type is overridden. The only valid value is {&quot;PageName&quot;:&quot;Home&quot;}.</td>
</tr>
<tr>
<td>pageType</td>
<td>string</td>
<td>Required. Specifies the type of the community page for which the default theme layout type is overridden. The only valid value is comm__standardPage.</td>
</tr>
<tr>
<td>themeLayoutType</td>
<td>CommunityThemeLayoutType (enumeration of type string)</td>
<td>Required if customThemeLayoutType is not specified. Provides the default theme layout type associated with the theme layout. Valid values are Inner, Home, or Login. This field and customThemeLayoutType are mutually exclusive; you can’t specify both.</td>
</tr>
</tbody>
</table>
### CommunityThemeSetting

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customThemeLayoutType</td>
<td>string</td>
<td>Required when themeLayoutType is not specified. The custom theme layout type associated with the theme layout. This field and themeLayoutType are mutually exclusive; you can't specify both.</td>
</tr>
<tr>
<td>themeLayout</td>
<td>string</td>
<td>Required. The configuration and layout for this theme.</td>
</tr>
<tr>
<td>themeLayoutType</td>
<td>CommunityTheme LayoutType (enumeration of type string)</td>
<td>Required when customThemeLayoutType is not specified. The default theme layout type associated with the theme layout. Valid values are Inner, Home, or Login. This field and customThemeLayoutType are mutually exclusive; you can't specify both.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of a CommunityThemeDefinition component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CommunityThemeDefinition xmlns="http://soap.sforce.com/2006/04/metadata">
  <bundlesInfo>
    <description>Batman Feature1 description</description>
    <order>1</order>
    <title>Batman Feature1</title>
    <type>Highlight</type>
  </bundlesInfo>
  <bundlesInfo>
    <image>siteAsset_d90e2d5ce4cf4d8899e233c051091246</image>
    <order>1</order>
    <title>siteAsset_d90e2d5ce4cf4d8899e233c051091246</title>
    <type>PreviewImage</type>
  </bundlesInfo>
  <defaultBrandingSet>Batman</defaultBrandingSet>
  <description>Batman theme</description>
  <enableExtendedCleanUpOnDelete>true</enableExtendedCleanUpOnDelete>
  <masterLabel>Batman</masterLabel>
  <themeRouteOverride>
    <pageAttributes>{"PageName":"&quot;Home&quot;"}</pageAttributes>
    <pageType>comm__standardPage</pageType>
    <themeLayoutType>Home</themeLayoutType>
  </themeRouteOverride>
  <themeSetting>
    <themeLayout>Batman_themeLayout_Login</themeLayout>
    <themeLayoutType>Login</themeLayoutType>
  </themeSetting>
  <themeSetting>
    <themeLayout>Batman_themeLayout_Home</themeLayout>
    <themeLayoutType>Home</themeLayoutType>
  </themeSetting>
  <themeSetting>
    <themeLayout>Batman_themeLayout_Default</themeLayout>
    <themeLayoutType>Inner</themeLayoutType>
  </themeSetting>
</CommunityThemeDefinition>
The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>Batman</members>
        <name>BrandingSet</name>
    </types>
    <types>
        <members>Batman</members>
        <name>CommunityThemeDefinition</name>
    </types>
    <types>
        <members>Batman_themeLayout_Default</members>
        <members>Batman_themeLayout_Home</members>
        <members>Batman_themeLayout_Login</members>
        <name>FlexiPage</name>
    </types>
    <types>
        <members>siteAsset_d90e2d5ce4cf4d8899e233c051091246</members>
        <name>StaticResource</name>
    </types>
    <version>46.0</version>
</Package>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retreiving Metadata with the Zip File.

### ConnectedApp

Represents a connected app configuration. A connected app integrates an application with Salesforce using APIs. Connected apps use standard SAML and OAuth protocols to authenticate, provide single sign-on, and provide tokens for use with Salesforce APIs. In addition to standard OAuth capabilities, Salesforce admins set various security policies for connected apps and have explicit control over who can use the corresponding apps.

This type extends the `Metadata` metadata type and inherits its `fullName` field.

### File Suffix and Directory Location

ConnectedApp components have the suffix `.connectedApp` and are stored in the `connectedApps` folder.

### Version

ConnectedApp components are available in API version 29.0 and later.
## Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>ConnectedAppAttribute</td>
<td>A custom attribute of the connected app.</td>
</tr>
<tr>
<td>canvasConfig</td>
<td>ConnectedAppCanvasConfig</td>
<td>The configuration options of the connected app if it’s exposed as a canvas app.</td>
</tr>
<tr>
<td>contactEmail</td>
<td>string</td>
<td>Required. The email address Salesforce uses for contacting you or your support team.</td>
</tr>
<tr>
<td>contactPhone</td>
<td>string</td>
<td>The phone number for Salesforce to use to contact you.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>An optional description for your application.</td>
</tr>
<tr>
<td>iconUrl</td>
<td>string</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>infoUrl</td>
<td>string</td>
<td>An optional URL for a web page with more information about your application.</td>
</tr>
<tr>
<td>ipRanges</td>
<td>ConnectedAppIpRange</td>
<td>Specifies the ranges of IP addresses that can access the app without requiring the user to authenticate with the connected app.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The name of the app.</td>
</tr>
<tr>
<td>logoUrl</td>
<td>string</td>
<td>An optional application logo. The logo appears with the application’s entry in the list of apps and on the consent page the user sees when authenticating. The URL must use HTTPS, and the logo can’t be larger than 125 pixels high or 200 pixels wide. The default logo is a cloud. Users are directed to this URL after they’ve authenticated when the app is accessed from a mobile device. If you don’t give a URL, the user is sent to the application’s default start page after authentication completes. If the connected app that you’re creating is a canvas app, then you can leave this field blank. The Canvas App URL field contains the URL that gets called for the connected app.</td>
</tr>
<tr>
<td>mobileStartUrl</td>
<td>string</td>
<td>Users are directed to this URL after they’ve authenticated when the app is accessed from a mobile device. If you don’t give a URL, the user is sent to the application’s default start page after authentication completes. If the connected app that you’re creating is a canvas app, then you can leave this field blank. The Canvas App URL field contains the URL that gets called for the connected app.</td>
</tr>
<tr>
<td>oauthConfig</td>
<td>ConnectedAppOAuthConfig</td>
<td>Specifies how your application communicates with Salesforce.</td>
</tr>
<tr>
<td>permissionSetName</td>
<td>string</td>
<td>Specifies the permissions required to perform different functions with the connected app. Available in API version 46.0 and later. You can assign multiple permission sets to the connected app, but you must enter each permission set name on a separate line. You can’t enter the same permission set name more than one time for each connected app. You can also change a permission set by replacing the current permission set with a new permission set. Make sure that each permission set name assigned to the connected app is unique. You can delete individual permission sets or remove all permission sets from a connected app by entering an empty permissionSetName string on deployment of the connected app (&lt;permissionSetName&gt;&lt;/permissionSetName&gt;).</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>plugin</td>
<td>string</td>
<td>The name of a custom Apex class that extends Auth.ConnectedAppPlugin to customize the behavior of the app.</td>
</tr>
<tr>
<td>pluginExecutionUser</td>
<td>string</td>
<td>Specifies the user to run the plugin as. If the user isn’t authorized to use the connected app, use the authorize method. See the ConnectedAppPlugin class in the Apex Developer Guide. Available in API version 46.0 and later. Enter a user that is part of your org. Otherwise, the user is removed from this field when you deploy the connected app. If you don’t want to specify a user, you can leave this field empty.</td>
</tr>
<tr>
<td>profileName</td>
<td>string</td>
<td>Specifies the profile (base-level user permissions) required to perform different functions with the connected app. Available in API version 46.0 and later. You can assign multiple profiles to the connected app, but you must enter each profile name on a separate line. You can’t enter the same profile name more than one time for each connected app. You can also change profiles by replacing the current profiles with new profiles. Make sure that each profile name assigned to the connected app is unique. You can also delete individual profiles, or remove all profiles from a connected app by entering an empty profileName string on deployment of the connected app (&lt;profileName&gt;&lt;/profileName&gt;).</td>
</tr>
<tr>
<td>samlConfig</td>
<td>ConnectedAppSamlConfig</td>
<td>Controls how the app uses single sign-on.</td>
</tr>
<tr>
<td>startUrl</td>
<td>string</td>
<td>If the app is not accessed from a mobile device, users are directed to this URL after they’ve authenticated. If you don’t give a URL, the user is sent to the application’s default start page after authentication completes. If the app is accessed from a mobile device, see mobileStartUrl. If the connected app that you’re creating is a canvas app, then you can leave this field empty. The Canvas App URL field contains the URL that gets called for the connected app.</td>
</tr>
</tbody>
</table>

Note: To use this field, the isAdminApproved field on the ConnectedAppOAuthConfig subtype must be set to true.

Note: To use this field in an org, the ConAppPluginExecuteAsUser org perm must be enabled.

Note: To use this field, the isAdminApproved field on the ConnectedAppOAuthConfig subtype must be set to true.
**ConnectedAppAttribute**

Represents the field names that make up a custom attribute when using SAML with a ConnectedApp. Tailor these values to a specific service provider.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>formula</td>
<td>string</td>
<td>Required. The value of the attribute.</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>Required. The attribute’s identifier.</td>
</tr>
</tbody>
</table>

**ConnectedAppCanvasConfig**

The configuration options of the connected app if it’s exposed as a canvas app.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessMethod</td>
<td>AccessMethod (enumeration of type string)</td>
<td>Required. Indicates how the canvas app initiates the OAuth authentication flow. The valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Get—OAuth authentication is used, and the user is prompted to allow the third-party application to access their information. When you use this access method, the canvas app must initiate the OAuth authentication flow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Post—OAuth authentication is used, but when the administrator installs the canvas app, they implicitly allow access for users. Therefore, the user isn’t prompted to allow the third party to access their user information. When you use this access method, the authentication is posted directly to the canvas app URL.</td>
</tr>
<tr>
<td>canvasUrl</td>
<td>string</td>
<td>Required. The URL of the third-party app that’s exposed as a canvas app.</td>
</tr>
<tr>
<td>lifecycleClass</td>
<td>string</td>
<td>The name of the Canvas.CanvasLifecycleHandler Apex class, if you’ve implemented this class for custom parameters. This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>locations</td>
<td>CanvasLocationOptions (enumeration of type string)</td>
<td>Indicates where the canvas app can appear to the user. The valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Aura—Reserved for future use.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- AppLauncher—Reserved for future use.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Chatter—The canvas app can appear in the app navigation list on the Chatter tab.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ChatterFeed—The canvas app can appear as a Chatter feed item.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- MobileNav—The canvas app can appear in a mobile card in the Salesforce app. This value is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>none</td>
<td>None</td>
<td>The canvas app can appear only in the Canvas App Previewer.</td>
</tr>
<tr>
<td>opencti</td>
<td>OpenCTI</td>
<td>The canvas app can appear in the call control tool.</td>
</tr>
<tr>
<td>pagelayout</td>
<td>PageLayout</td>
<td>The canvas app can appear on a page layout.</td>
</tr>
<tr>
<td>publisher</td>
<td>Publisher</td>
<td>The canvas app can appear as a global action.</td>
</tr>
<tr>
<td>servicedesk</td>
<td>ServiceDesk</td>
<td>The canvas app can appear in the footer or sidebars of a Salesforce console.</td>
</tr>
<tr>
<td>userprofile</td>
<td>UserProfile</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>visualforce</td>
<td>Visualforce</td>
<td>The canvas app can appear on a Visualforce page.</td>
</tr>
<tr>
<td>options</td>
<td>CanvasOptions (enumeration of type string)</td>
<td>Indicates whether to hide the share button and header in the publisher for your canvas app, and whether the app is a canvas personal app. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- HideShare—The Share button is hidden in the publisher for the related canvas app. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- HideHeader—The header is hidden in the publisher for the related canvas app. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- PersonalEnabled—End users can install the app as a canvas personal app. Available in API version 32.0 and later.</td>
</tr>
<tr>
<td>samlinitiationmethod</td>
<td>SamlInitiationMethod (enumeration of type string)</td>
<td>If you’re using SAML single sign-on (SSO), indicates which provider initiates the SSO flow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- IdpInitiated—Identity provider initiated. Salesforce makes the initial request to start the SSO flow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SpInitiated—Service provider initiated. The canvas app starts the SSO flow after it’s invoked.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- None—The canvas app isn’t using SAML SSO. This field is available in API version 31.0 and later.</td>
</tr>
</tbody>
</table>

**ConnectedAppIpRange**

The list of IP addresses that can access the app without requiring the user to authenticate.
### DescriptionField TypeField Name
Use this field to identify the purpose of the range, such as which part of a network corresponds to this range. This field is available in API version 31.0 and later.

**string**

```
string description
```

Required. The first address in the IP range, inclusive.

**string**

```
string startAddress
```

Required. The last address in the IP range, inclusive.

**string**

```
string endAddress
```

### ConnectedAppOauthConfig

Represents the field names that configure how your application communicates with Salesforce.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callbackUrl</td>
<td>string</td>
<td>Required. The endpoint that Salesforce calls back to your application during OAuth; it's the OAuth redirect_uri.</td>
</tr>
<tr>
<td>certificate</td>
<td>string</td>
<td>The PEM-encoded certificate string, if the app uses a certificate.</td>
</tr>
</tbody>
</table>
| consumerKey    | string     | A value used by the consumer for identification to Salesforce. Referred to as `client_id` in OAuth 2.0.

In API version 32.0 and later, you can set this field's value only during creation. After you define and save the value, it can't be edited. The value must be alphanumeric, can't contain special characters or spaces, and must be between 8 and 256 characters. Consumer keys must be globally unique.

| consumerSecret | string     | A value that is combined with the consumerKey and used by the consumer for identification to Salesforce. Referred to as `client_secret` in OAuth 2.0. Typically, Salesforce generates this value when you create the connected app. However, you can customize the shared secret value during creation. After you save the value, it can't be edited. When set, the value is not returned in metadata API requests.

The value must be alphanumeric (no special characters and no spaces) and a minimum of 8 characters (maximum of 256 characters). If you specify a secret already in use for another connected app in the organization, an error occurs.

This field is available in API version 32.0 and later.

| idTokenConfig   | ConnectedAppOauthIdToken | Specifies the ID token configuration for the connected app OAuth settings. Available in API version 43.0 and later. |
| isAdminApproved | boolean                 | If set to `false` (default setting), anyone in the org can authorize the app. Users must approve the app the first time they access it.

If set to `true`, only users with the appropriate profile or permission set can access the app. These users don't have to approve the app before they can access it. Manage profiles for the app by editing each
profile's Connected App Access list. Manage permission sets for the app by editing each permission set's Assigned Connected App list. This setting is not available in Group Edition. Available in API version 46.0 and later.

Note: Connected app consumers can edit this setting when deploying a connected app in their org.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopes</td>
<td>ConnectedAppOauthAccessScope (enumeration of type string)</td>
<td>The scopes refer to permissions given by the user running the connected app. When deploying metadata, valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Address—Allows access to the logged-in user's street address (the same behavior as deploying Basic).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Api—Allows access to the logged-in user's account over the APIs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Basic—Allows access to your identity URL service (the same behavior as deploying Address, Email, Phone, and Profile).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Chatter—Allows access to only the Chatter REST API resources.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CustomApplications—Provides access to custom applications, such as those using Visualforce.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CustomPermissions—Allows access to the custom permissions in an organization associated with the connected app, and shows whether the current user has each permission enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Email—Allows access to the logged-in user’s email address (the same behavior as deploying Basic).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Full—Allows access to all data accessible by the logged-in user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• OfflineAccess—Allows the app to interact with the user's data while the user is offline, and get a refresh token (the same behavior as deploying RefreshToken).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• OpenID—Allows access to the logged-in user's unique identifier for OpenID Connect apps.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Phone—Allows access to the logged-in user’s phone number value (the same behavior as deploying Basic).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Profile—Allows access to the logged-in user's profile (the same behavior as deploying Basic).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RefreshToken—Allows a refresh token to be returned if you’re eligible to receive one (the same behavior as deploying OfflineAccess).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Wave—Allows access to the Analytics REST API resources. Available in API version 35.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Web—Allows use of the access_token on the web. This usage also includes visualforce, allowing access to Visualforce pages.</td>
</tr>
</tbody>
</table>

When retrieving metadata, valid values are:
### Field Types

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Api —</td>
<td>Allows access to the logged-in user's account over the APIs.</td>
</tr>
<tr>
<td>Basic —</td>
<td>Allows access to the user’s identity URL service, and includes Address, Email, Phone, and Profile.</td>
</tr>
<tr>
<td>Chatter —</td>
<td>Allows access to only the Chatter REST API resources.</td>
</tr>
<tr>
<td>CustomApplications —</td>
<td>Allows access to custom applications, such as those using Visualforce.</td>
</tr>
<tr>
<td>Full —</td>
<td>Allows access to all data accessible by the logged-in user.</td>
</tr>
<tr>
<td>OpenID —</td>
<td>Allows access to the logged in user’s unique identifier for OpenID Connect apps.</td>
</tr>
<tr>
<td>RefreshToken —</td>
<td>Allows a refresh token to be returned if you are eligible to receive one, and is synonymous with allowing OfflineAccess.</td>
</tr>
<tr>
<td>Wave —</td>
<td>Allows access to the Analytics REST API resources. Available in API version 35.0 and later.</td>
</tr>
<tr>
<td>Web —</td>
<td>Allows usage of the access_token on the web. This usage also includes visualforce, allowing access to Visualforce pages.</td>
</tr>
</tbody>
</table>

**singleLogoutUrl**  
**string**  
The single logout endpoint. This URL is the endpoint where Salesforce sends a logout request when users log out of Salesforce.

### ConnectedAppOauthIdToken

Specifies the ID token configuration for the connected app OAuth settings. Available in API version 43.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>idTokenAudience</td>
<td>string</td>
<td>The audiences that this ID token is intended for. The value is an array of case-sensitive strings. If no audiences are specified, the OAuth 2.0 client_id of the relying party is returned as the default audience. Otherwise, the other audiences are returned with the client_id in the aud value.</td>
</tr>
<tr>
<td>idTokenIncludeAttributes</td>
<td>boolean</td>
<td>Indicates whether attributes are included in the ID token.</td>
</tr>
<tr>
<td>idTokenIncludeCustomPerms</td>
<td>boolean</td>
<td>Indicates whether custom permissions are included in the ID token.</td>
</tr>
<tr>
<td>idTokenIncludeStandardClaims</td>
<td>boolean</td>
<td>Indicates whether standard claims about the authentication event are included in the ID token.</td>
</tr>
<tr>
<td>idTokenValidity</td>
<td>int</td>
<td>The length of time that the ID token is valid for after it’s issued. The value can be from 1 to 720 minutes. The default is 2 minutes.</td>
</tr>
</tbody>
</table>

### ConnectedAppSamlConfig

Specifies how an app uses single sign-on.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acsUrl</td>
<td>string</td>
<td>Required. The assertion consumer service URL from the service provider.</td>
</tr>
<tr>
<td>certificate</td>
<td>string</td>
<td>The PEM-encoded certificate string, if the app uses a certificate.</td>
</tr>
<tr>
<td>entityUrl</td>
<td>string</td>
<td>Required. The entity ID from your service provider.</td>
</tr>
<tr>
<td>encryptionCertificate</td>
<td>string</td>
<td>The name of the certificate to use for encrypting SAML assertions to the service provider. This certificate is saved in the organization’s Certificate and Key Management list. Available in API version 30.0 and later.</td>
</tr>
</tbody>
</table>
| encryptionType      | SamlEncryptionType (enumeration of type string) | When Salesforce is the identity provider, the SAML configuration can specify the encryption method used for encrypting SAML assertions to the service provider. The service provider detects the encryption method in the SAML assertion for decryption. Valid values are:  
  - AES_128—128-bit key.  
  - AES_256—256-bit key.  
  - Triple_Des—Triple Data Encryption Algorithm.  
  Available in API version 30.0 and later. |
| issuer              | string              | A URI that sends the SAML response. A service provider can use this URI to determine which identity provider sent the response. Available in API version 29.0 and later.                                              |
| samlIdpSLOBinding   | SamlIdpSLOBinding (enumeration of type string) | The SAML HTTP binding type from the service provider used for single logout. Available in API version 40.0 and later. Valid values are:  
  - PostBinding  
  - RedirectBinding |
| samlNameIdFormat    | SamlNameIdFormatType (enumeration of type string) | Indicates the format the service provider (SP) requires for the user’s single sign-on identifier. Available in API version 29.0 and later. Valid values are:  
  - Unspecified—No format given. This option is the default.  
  - EmailAddress—Used if the subject type is the user’s name or federation ID (an ID internal to the SP).  
  - Persistent—Used with the user ID and persistent ID subject types.  
  - Transient—Used when the subject type is a custom attribute and can change every time the user logs in. |
| samlSloUrl          | string              | The SAML single-log out endpoint of the connected app service provider (SP). This endpoint is where SAML LogoutRequests and LogoutResponses are sent when users log out of Salesforce. The SP provides this endpoint. Available in API version 40.0 and later. |
| samlSubjectCustomAttr | string              | If the samlSubjectType is CustomAttr, include that custom value here; otherwise, leave empty. Available in API version 29.0 and later.                                                                     |
### Declarative Metadata Sample Definition

The following is an example package manifest used to deploy or retrieve the ConnectedApp metadata for an organization.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>PortalTestApp</members>
    <name>ConnectedApp</name>
  </types>
  <version>29.0</version>
</Package>
```

The following is an example of a ConnectedApp component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ConnectedApp xmlns="http://soap.sforce.com/2006/04/metadata">
  <fullName>AConnectedApp</fullName>
  <attributes>
    <formula>$User.CompanyName</formula>
    <key>companyName</key>
  </attributes>
  <contactEmail>user@company.com</contactEmail>
  <mobileStartUrl>https://m.connectedapp.company.com</mobileStartUrl>
  <label>A ConnectedApp</label>
  <oauthConfig>
    <callbackUrl>https://callback.yourcompany.com</callbackUrl>
    <consumerKey>**********************************************</consumerKey>
    <isAdminApproved>true</isAdminApproved>
    <scopes>Basic</scopes>
    <scopes>Api</scopes>
    <scopes>Web</scopes>
    <scopes>Full</scopes>
    <scopes>Chatter</scopes>
    <scopes>CustomApplications</scopes>
    <scopes>RefreshToken</scopes>
    <scopes>OpenID</scopes>
    <scopes>CustomPermissions</scopes>
    <scopes>Wave</scopes>
  </oauthConfig>
</ConnectedApp>
```
<scopes>Eclair</scopes>
</oauthConfig>
<permissionSetName>Advanced</permissionSetName>
<permissionSetName>Service</permissionSetName>
<profileName>Solution Manager</profileName>
<profileName>System Administrator</profileName>
<plugin>ConnectedAppPluginExample</plugin>
<pluginExecutionUser>admin@salesforce.org</pluginExecutionUser>
<startUrl>https://connectedapp.company.com</startUrl>
<ipRanges>
  <endAddress>10.x.x.xx</endAddress>
  <startAddress>10.x.x.yy</startAddress>
</ipRanges>
</ConnectedApp>

Usage

If you’re constructing a SAML-enabled connected app using Metadata API, and must set the IdP-Initiated Login URL for your service provider, you have two options:

You can use the service provider app ID with the `app` parameter in the following format. This value is displayed in the Salesforce user interface. From Setup, enter Connected Apps in the Quick Find box, then select Connected Apps, then click the name of the connected app to see its detail page.

https://<Salesforce_base_URL>/idp/login?app=<app_id>

Or, if you’re configuring the connected app using Metadata API only, you can use the `apiName` parameter of the service provider app in the following format. The `apiName` parameter is the `fullName` inherited from the Metadata type.

https://<Salesforce_base_URL>/idp/login?apiName=<fullName>

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

ContentAsset

Represents the metadata for creating an asset file. Asset files enable a Salesforce file to be used for org setup and configuration purposes. This type extends the `MetadataWithContent` metadata type and inherits its `content` and `fullName` fields.

File Suffix and Directory Location

ContentAsset components have the suffix .asset and are stored in the `contentassets` folder.

Version

ContentAsset components are available in API version 38.0 and later.
**Special Access Rules**

The system prevents metadata retrieval if the total size of the asset’s file content exceeds 30 MB. All pre-existing limits for packaging apply to asset files.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>format</td>
<td>ContentAssetFormat (enumeration of type string)</td>
<td>Describes the format of the asset file. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Original—A single asset file version.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ZippedVersions—Contains multiple versions of the asset file.</td>
</tr>
<tr>
<td>isVisibleByExternalUsers</td>
<td>boolean</td>
<td>Indicates whether unauthenticated users can see the asset file (true) or not (false). If not specified, the default value is false. This field is available in API version 44.0 and later.</td>
</tr>
<tr>
<td>language</td>
<td>string</td>
<td>Required. The language of the asset file label.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The label for the asset file record, which displays in Setup.</td>
</tr>
<tr>
<td>originNetwork</td>
<td>string</td>
<td>For deploys, the name of the community the file is assigned upon creation. For retrievals, the name of the community the file is assigned to populates the field value. If null, file was not assigned to a community.</td>
</tr>
<tr>
<td>relationships</td>
<td>ContentAssetRelationships[]</td>
<td>The list of ContentAssetLinks that describe whether the asset file should be shared with the org.</td>
</tr>
<tr>
<td>versions</td>
<td>ContentAssetVersions</td>
<td>Required. Captures basic information about the file version(s) included the asset metadata. Typically the file has only one version.</td>
</tr>
</tbody>
</table>

**ContentAssetRelationships**

Represents the relationships between an asset file and the locations it's linked with.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>organization</td>
<td>ContentAssetLink[]</td>
<td>Carries information about sharing the asset file with the org. Maps to ContentDocumentLink.</td>
</tr>
</tbody>
</table>

**ContentAssetLink**

Represents a relationship link for an asset file, and includes details about the level of access for the link.
### Field Name | Field Type | Description
--- | --- | ---
**access** | ContentAssetAccess (enumeration of type string) | Required. The permission granted to the user of the shared file, determined by the permission the user already has. Valid values are:
- VIEWER
- COLLABORATOR
- INFERRED

**name** | string | Reserved for future use.

### ContentAssetVersions
Represents information about all file versions included in the asset metadata.

| Field Name | Field Type | Description |
|--- | --- | ---
| version | ContentAssetVersion[] | A list of file versions for the asset.

### ContentAssetVersion
Represents information about one file version included in the asset metadata.

| Field Name | Field Type | Description |
|--- | --- | ---
| number | string | Required. The version number. This field is based on, or sets, the ContentVersion. |
| pathOnClient | string | Required. Describes the original filename of the file. This field maps to ContentVersion.PathOnClient. It provides the data for the ContentVersion Title field. |
| zipEntry | string | If the asset file has more than one version, format is ZippedVersions. In this case, zipEntry is the name of the file within the zip. If the asset file has only one version, this field is empty. |

### Declarative Metadata Sample Definition
The following is an example of a ContentAsset component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ContentAsset xmlns="http://soap.sforce.com/2006/04/metadata">
  <masterLabel>some asset</masterLabel>
  <relationships>
    <organization>
      <access>VIEWER</access>
    </organization>
  </relationships>
  <versions>
    <version>
      <number>1</number>
    </version>
  </versions>
</ContentAsset>
```
For assets that include just one version, the format field can be omitted or specified with the value as Original. File assets with more than one version have versions wrapped in a zip file.

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>MyAsset</members>
    <name>ContentAsset</name>
  </types>
  <version>46.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**CorsWhitelistOrigin**

Represents an origin in the CORS whitelist.

**File Suffix and Directory Location**

CorsWhitelistOrigin components have the suffix .corswhitelistorigin and are stored in the corswhitelistorigins folder.

**Version**

CorsWhitelistOrigin components are available in API version 32.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>urlPattern</td>
<td>String</td>
<td>A URL pattern for the origin. The origin URL pattern must include the HTTPS protocol and a domain name, and may include a port. The wildcard character (<em>) is supported and must be in front of a second-level domain name. For example, https://</em>.example.com adds all subdomains of example.com to the whitelist.</td>
</tr>
</tbody>
</table>
The origin URL pattern can be an IP address. However, an IP address and a domain that resolve to the same address are not the same origin and you must add them to the CORS whitelist as separate entries.

Declarative Metadata Sample Definition

The following is an example package manifest used to deploy or retrieve the CorsWhitelistOrigin metadata for an organization.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>*</members>
        <name>CorsWhitelistOrigin</name>
    </types>
    <version>32.0</version>
</Package>
```

The following is an example of a CorsWhitelistOrigin component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CorsWhitelistOrigin xmlns="http://soap.sforce.com/2006/04/metadata">
    <developerName>CorsWhitelistEntry1</developerName>
    <urlPattern>https://*.example.com</urlPattern>
</CorsWhitelistOrigin>
```

Usage

CORS (cross-origin resource sharing) is a W3C recommendation that enables Web browsers to request resources from origins other than their own. For example, using CORS, a JavaScript script at https://www.example.com could request a resource from https://www.salesforce.com.

If a browser that supports CORS makes a request to an origin in the Salesforce CORS whitelist, Salesforce returns the origin in the Access-Control-Allow-Origin HTTP header, along with any additional CORS HTTP headers. If the origin is not whitelisted, Salesforce returns HTTP status code 404.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

CspTrustedSite

Represents a CSP Trusted Site. The Lightning Component framework uses Content Security Policy (CSP) to impose restrictions on content. The main objective is to help prevent cross-site scripting (XSS) and other code injection attacks. To use third-party APIs that make requests to an external (non-Salesforce) server or to use a WebSocket connection, add a CSP Trusted Site.
Declarative Metadata File Suffix and Directory Location

CspTrustedSite components are stored in the cspTrustedSites directory of the corresponding package directory. The file name matches the unique name of the trusted site, and the extension is .cspTrustedSite.

Version

CspTrustedSite components are available in API version 39.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>CspTrustedSiteContext (enumeration of type string)</td>
<td>Declares the scope of trust for the listed third-party host.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• All whitelists the host for both Lightning Experience and Lightning Communities experience.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Communities whitelists the host for Lightning Communities experience only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• (Default) LEX whitelists the host for Lightning Experience only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 44.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description explaining what this trusted site is used for.</td>
</tr>
<tr>
<td>endpointUrl</td>
<td>string</td>
<td>Required. The URL for the trusted site.</td>
</tr>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Required. Indicates if the trusted site is active (true) or not (false).</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

A sample XML definition of a trusted site is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CspTrustedSite xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>Used for Lightning component callout to mapping web service</description>
  <endpointUrl>https://www.maptestsite.net/</endpointUrl>
  <isActive>true</isActive>
  <context>LEX</context>
</CspTrustedSite>
```

Usage

CSP is a W3C standard that defines rules to control the source of content that can be loaded on a page. All CSP rules work at the page level, and apply to all components and libraries. By default, the framework's headers allow content to be loaded only from secure (HTTPS) URLs and forbid XHR requests from JavaScript.

When you define a CSP Trusted Site, the site's URL is added to the list of allowed sites for the following directives in the CSP header.
This change to the CSP header directives allows Lightning components to load resources, such as images, styles, and fonts, from the site. It also allows client-side code to make requests to the site.

**Important:** You can’t load JavaScript resources from a third-party site, even if it’s a CSP Trusted Site. To use a JavaScript library from a third-party site, add it to a static resource, and then add the static resource to your component. After the library is loaded from the static resource, you can use it as normal.

---

### CustomApplication

CustomApplication represents a custom or standard application. In API version 29.0 and earlier, CustomApplication represents only a custom application. An application is a list of tab references, with a description and a logo. This type extends the Metadata metadata type and inherits its `fullName` field.

### File Suffix and Directory Location

Custom and standard applications have the suffix `.app` and are stored in the `applications` folder.

**Note:** Retrieving a component of this metadata type in a project makes the component appear in any Profile and PermissionSet components that are retrieved in the same package.

### Version

Custom applications are available in API version 10.0 and later. Standard applications are available in API version 30.0 and later.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionOverrides</td>
<td>AppActionOverride[]</td>
<td>Represents an action override for an application. Use it to create, update, edit, or delete action overrides. This field is available for Lightning Experience in API version 38.0 and later.</td>
</tr>
<tr>
<td>brand</td>
<td>AppBrand</td>
<td>The color scheme and logo used for the app. This field is available for Lightning Experience in API version 38.0 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consoleConfig</td>
<td>ServiceCloudConsoleConfig</td>
<td>Represents configuration settings for a Salesforce console app. This field is available in API version 42.0 and later.</td>
</tr>
<tr>
<td>defaultLandingTab</td>
<td>string</td>
<td>The fullName of a standard tab or custom tab that opens when this application is selected.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The optional description text of the application.</td>
</tr>
</tbody>
</table>
| formFactors          | FormFactor (enumeration of type string) | Indicates the form factors for which the app is visible for Lightning Experience. Valid values are:  
  - Null (no value)—For a desktop using Salesforce Classic  
  - Small—Reserved for future use  
  - Medium—Reserved for future use  
  - Large—For a desktop using Lightning Experience  
This field is available in API version 38.0 and later.  
  
  **Note:** As of version 38.0, formFactors is set to Large for existing Salesforce Classic apps, except for Salesforce Classic consoles. Salesforce Classic apps installed from packages created before version 38.0 also have formFactors set to Large. For Salesforce Classic apps in packages created with 38.0 or later, you must set formFactors to Large for Salesforce Classic apps to appear in the Lightning Experience desktop. |
<p>| isNavAutoTempTabsDisabled | boolean                           | Indicates whether the navigation automatically creates temporary tabs settings. Available in API version 43.0 and later.                                                                                         |
| isNavPersonalizationDisabled | boolean                       | Indicates whether navigation personalization is disabled. Available in API version 43.0 and later.                                                                                                           |
| isServiceCloudConsole | boolean                           | Indicates if the application is a Salesforce Classic console app. For Lightning Experience console apps, this field is null and the navType field is set to Console.                                                |
| label                | string                             | The name of the application.                                                                                                                                                                                 |
| logo                 | string                             | The optional reference to the image document for a Salesforce app or Salesforce console.                                                                                                                     |
| navType              | NavType (enumeration of type string) | Not updateable. Indicates the type of navigation the app uses. The value Standard is for Lightning Experience. The value Console is for Salesforce Classic. A null value means Salesforce Classic. |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>preferences</td>
<td>AppPreferences</td>
<td>This field is available in API version 38.0 and later. Represents the preferences for a Salesforce console app. This field is available in API version 42.0 and later.</td>
</tr>
<tr>
<td>profileActionOverrides</td>
<td>AppProfileActionOverride[]</td>
<td>A list of the Lightning Experience record page ProfileActionOverrides that are assigned to this custom app. When a user invokes the custom app, a matching ProfileActionOverride assignment takes precedence over existing overrides for the record page specified in ActionOverride. This lets you override a record page for the custom app by record type and profile. In API version 45.0 and later, you can override a home page for the custom app by profile.</td>
</tr>
<tr>
<td>setupExperience</td>
<td>string</td>
<td>The type of Setup experience associated with the app. Valid values are: all—Represents the full Setup tree. essentials—Represents the Essentials Setup tree, which contains a subset of Setup items configured for Essentials edition. service—Represents the Service Setup tree, which contains a subset of Setup items configured for Service Console. A null value is equivalent to all. Previous valid values AllSetup, ServiceSetup, and EssentialsSetup have been deprecated. This field is available in API version 39.0 and later.</td>
</tr>
<tr>
<td>subscriberTabs</td>
<td>string[]</td>
<td>Represents the list of tabs appended by a subscriber to a Lightning app installed from a managed package. Records in a subscriber tab always open as primary tabs. This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>tabs</td>
<td>string</td>
<td>The list of tabs included in this application. In API version 12.0, the fullName for built-in tabs like Home, Account, and Reports, is the name of the tab (Home, for example). In API version 13.0 and later, built-in tabs are prefixed with standard-. For example, to reference the Account tab you would use standard-Account. In API version 42.0, this field was renamed from tab to tabs.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| uiType      | UiType (enumeration of type string) | Not updateable. Identifies the type of custom app. The value is:  
|             |                                   | • Aloha for Salesforce Classic  
|             |                                   | • Lightning for Lightning Experience  
|             |                                   | This field is available in API version 38.0 and later.                                                                                      |
| utilityBar  | string                            | The developer name of the UtilityBar associated with this app.  
|             |                                   | Note: We recommend assigning a UtilityBar to only one Lightning App, because UtilityBars are shared. Sharing means that if you change the UtilityBar in one app, it automatically changes in all apps it’s part of.  
|             |                                   | This field is available in API version 38.0 and later.                                                                                      |
| workspaceConfig | AppWorkspaceConfig               | Represents how records open in a Salesforce console app. Required if isServiceCloudConsole is true. In API version 42.0, this field was renamed to workspaceConfig from workspaceMappings. |

**AppActionOverride**

Represents an action override for an application. Use it to create, update, edit, or delete action overrides. AppActionOverride inherits from ActionOverride and extends it by one field, pageOrSobjectType. Available for Lightning Experience in API version 38.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>string</td>
<td>The only valid value is view for API version 43.0 and earlier. The value tab is supported for API version 44.0 and later.</td>
</tr>
<tr>
<td>comment</td>
<td>string</td>
<td>Any comments you want associated with the override.</td>
</tr>
<tr>
<td>content</td>
<td>string</td>
<td>Set this field if type is set to flexipage. It refers to the name of the page to use as the override. To reference installed components, use the format of Component_namespace__Component_name</td>
</tr>
<tr>
<td>formFactor</td>
<td>FormFactor (enumeration of type string)</td>
<td>The size of the page being overridden. If the type field is set to flexipage, set this field to Large to override the View action with a Lightning page in Lightning Experience. The Large value represents the Lightning Experience desktop environment and is valid only for the flexipage and lightningcomponent types. The Small value represents the Salesforce app on a phone or tablet. The Medium value is reserved for</td>
</tr>
</tbody>
</table>
null value (which is the same as specifying no value) represents Salesforce Classic.

This field is available in API version 37.0 and later and is part of the feature for creating and editing record pages in Lightning Experience.

**Note:** Lightning component overrides return different FormFactor values depending on the API version used.

- In API version 41.0 and earlier, Lightning component overrides return only the null value (no value), representing the Salesforce Classic environment.
- In API version 42.0, if you specify different Lightning component overrides for Lightning Experience and mobile, one component is selected randomly for both overrides and its FormFactor value is returned. If there’s a conflict between Lightning components, and a Visualforce page override is also specified for Salesforce Classic, the Visualforce page takes precedence.
- In API version 43.0 and later, a Lightning component override for Lightning Experience returns the Large value and a Lightning component override for mobile returns the Small value, as expected.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pageOrSobjectType</td>
<td>string</td>
<td>The name of the sObject type being overridden. Valid values are standard and custom. This value must be standard-home when actionName is tab.</td>
</tr>
<tr>
<td>skipRecordTypeSelect</td>
<td>boolean</td>
<td>Set this field to true if you prefer that any new records created by this action aren’t forwarded to the record type selection page. This field is only valid if the actionName is a “create” type (like new), and type is set to visualforce.</td>
</tr>
<tr>
<td>type</td>
<td>ActionOverrideType (enumeration of type string)</td>
<td>Required. Represents the type of action override. The valid values are Flexipage and Default.</td>
</tr>
</tbody>
</table>

**AppBrand**

The color scheme and logo used for the app. Available for Lightning apps in API version 38.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>footerColor</td>
<td>string</td>
<td>Optional. Determines the footer color in the app. Specify the color with a hexadecimal code, such as #0000FF for blue.</td>
</tr>
<tr>
<td>headerColor</td>
<td>string</td>
<td>Optional. Determines the header color in the app. Specify the color with a hexadecimal code, such as #0000FF for blue.</td>
</tr>
</tbody>
</table>
### Metadata Types

#### CustomApplication

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>logo</td>
<td>string</td>
<td>The optional reference to the image document for the application.</td>
</tr>
<tr>
<td>logoVersion</td>
<td>int</td>
<td>An optional version number for the logo.</td>
</tr>
<tr>
<td>shouldOverrideOrgTheme</td>
<td>boolean</td>
<td>Indicates whether to override the global theme for the org. When true, the color scheme and logo that the user has set are used. When false, the global theme for the org is used, even if the user has set a color scheme and logo.</td>
</tr>
</tbody>
</table>

#### AppComponentList

Represents custom console components (Visualforce pages) assigned to a Salesforce console app. In API version 42.0, this type was renamed from CustomApplicationComponents to AppComponentList.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignment</td>
<td>string</td>
<td>Required. Determines how custom console components are aligned in the footer of a Salesforce console app.</td>
</tr>
<tr>
<td>components</td>
<td>string[]</td>
<td>The name of a custom console component assigned to a Salesforce console app. In API version 42.0, this field was renamed from customApplicationComponent to components.</td>
</tr>
</tbody>
</table>

#### AppPreferences

Represents the preferences for a Salesforce console app. Available in API version 42.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableCustomizeMyTabs</td>
<td>boolean</td>
<td>Indicates if a Salesforce console app has Customize My Tabs enabled. If enabled, users can hide, display, and organize items in the navigation tab.</td>
</tr>
<tr>
<td>enableKeyboardShortcuts</td>
<td>boolean</td>
<td>Indicates if a Salesforce console app has keyboard shortcuts enabled. Shortcuts let users perform actions by pressing a combination of keys instead of having to use a mouse. After keyboard shortcuts are enabled, several default shortcuts are available for customization. Before you can create custom shortcuts, a developer must define the shortcut’s action with the addEventListener() method in the Salesforce Console Integration Toolkit. You can’t create keyboard shortcuts for actions performed outside of the console. This field is required if isServiceCloudConsole is true.</td>
</tr>
<tr>
<td>enableListViewHover</td>
<td>boolean</td>
<td>Indicates if a Salesforce console app has list view hovers enabled. If set to true, summary information is displayed about a record in a responsive list when the user hovers over a record name. For cases, hover over the subject field.</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
---|---|---
enableListViewReskin | boolean | Indicates if Salesforce console apps use responsive list views instead of Salesforce Classic lists views.
enableMultiMonitorComponents | boolean | Indicates if a Salesforce console app has multi-monitor components enabled, which lets users move portions of a console from their browsers to locations on their screens. This field is required if isServiceCloudConsole is true.
enablePinTabs | boolean | Indicates if a Salesforce console app has pinned tabs enabled, which lets users pin primary tabs to the tab bar for quick access.
enableTabHover | boolean | Indicates if a Salesforce console app has tab hover enabled. If enabled, summary information is displayed about a record in an overlay when the user hovers over a tab.
enableTabLimits | boolean | Indicates whether limits are enabled on the number of primary tabs and subtabs that can be opened in a Salesforce console session. When true, values for tabLimitConfig are required.
saveUserSessions | boolean | Indicates if a Salesforce console app saves user sessions automatically. If enabled, when console users close their browsers or log out of Salesforce, any previously open tabs display when users log in again. Required if isServiceCloudConsole is true.

### AppProfileActionOverride

Represents a ProfileActionOverride for a custom app. This type inherits from ProfileActionOverride on page 671 and extends it by one field, profile. Available for Lightning Experience in API version 39.0 and later. In API version 45.0 and later, you can override a home page for the custom app by profile.

| Field Name | Field Type | Description |
---|---|---|
actionName | string | The name of the action. The only valid values are Tab and View. If pageOrSobjectType is record-home, this field must be View. The View action is supported only when ProfileActionOverride is being specified as part of a CustomApplication. In API version 45.0 and later, this action is supported only when ProfileActionOverride is being specified as part of a CustomApplication, pageOrSobjectType is standard-home, and this field is Tab.
content | string | Read-only. Represents the name of the Lightning page being used as the override.
formFactor | FormFactor (enumeration of type string) | The size of the page being overridden. The Large value represents the Lightning Experience desktop environment.
### Metadata Types

#### CustomApplication

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pageOrObjectType</td>
<td>string</td>
<td>The name of the page being overridden. The only valid values are record-home and standard-home. If the actionName is Tab, this field must be standard-home</td>
</tr>
<tr>
<td>profile</td>
<td>string</td>
<td>The profile associated with the ProfileActionOverride.</td>
</tr>
<tr>
<td>recordType</td>
<td>string</td>
<td>The record type associated with the override. If pageOrObjectType is standard-home, this field must be null. This field is required when actionName is set to View.</td>
</tr>
<tr>
<td>type</td>
<td>ActionOverrideType (enumeration of type string)</td>
<td>Read-only. The type of action override. The only valid value is flexipage.</td>
</tr>
</tbody>
</table>

#### AppWorkspaceConfig

Represents how records open in a Salesforce console app. Required if isServiceCloudConsole is true. Available for Salesforce Classic console apps in API version 25.0 and later. Available for Lightning console apps in API version 41.0 and later. In API version 42.0, this type was renamed from WorkspaceMappings to AppWorkspaceConfig.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mappings</td>
<td>WorkspaceMapping[]</td>
<td>Represents how records for a specific tab open in a Salesforce console app. Required for each tab specified in the CustomApplication. In API version 42.0, this field was renamed from workspaceMapping to mappings.</td>
</tr>
</tbody>
</table>

#### CustomShortcut

Represents custom keyboard shortcuts assigned to a Salesforce console app. Before you can create custom shortcuts, a developer must define the shortcut’s action with the addEventListener() method in the Salesforce Console Integration Toolkit. You can’t create keyboard shortcuts for actions performed outside of the console. Available in API version 28.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>string</td>
<td>Required. The action performed in the console when a user presses the keyboard shortcut.</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Indicates whether the keyboard shortcut is active (true) or not (false).</td>
</tr>
<tr>
<td>keyCommand</td>
<td>string</td>
<td>Required. The combination of keys a user presses to trigger the keyboard shortcut. Keyboard shortcuts aren’t case-sensitive, but they display as uppercase on setup pages in the Salesforce user interface so that they’re easier to read. Each key command can include up to four modifier keys followed by one non-modifier key. Modifier and non-modifier keys are separated by the + key. Modifier keys can occur in any order, but you must place</td>
</tr>
</tbody>
</table>
non-modifier keys at the end of the key command sequence. For example, 
SHIFT+CTRL+ALT+META +A.

Valid modifier keys are:

- SHIFT
- CTRL
- ALT
- META (represents the COMMAND key on Macs)

Valid non-modifier keys are letters A through Z and numbers 0 through 9. Other valid keys are:

- TAB
- ENTER
- PAUSE/BREAK
- CAPS LOCK
- ESC
- SPACE
- PAGE UP
- PAGE DOWN
- END
- HOME
- LEFT ARROW
- UP ARROW
- RIGHT ARROW
- DOWN ARROW
- PRINT SCREEN
- INSERT
- DELETE
- RIGHT WINDOW
- NUMPAD 0
- NUMPAD 1
- NUMPAD 2
- NUMPAD 3
- NUMPAD 4
- NUMPAD 5
- NUMPAD 6
- NUMPAD 7
- NUMPAD 8
- NUMPAD 9
- MULTIPLY
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• ADD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SUBTRACT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DECIMAL POINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DIVIDE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• F12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NUM LOCK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SCROLL LOCK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• =</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• –</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• .</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• /</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• [</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ’</td>
</tr>
</tbody>
</table>

| description | string | The optional description text for the keyboard shortcut.                     |
| eventName   | string  | Required. Code available to developers who want to add custom shortcut functions to the console via the Salesforce Console Integration Toolkit. |

**DefaultShortcut**

Represents default keyboard shortcuts assigned to a Salesforce console app. Once you enable keyboard shortcuts for a console, several default shortcuts are available for customization. These include opening and closing tabs, moving between tabs, and saving records. Available in API version 28.0 and later.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| action     | string     | Required. The action performed in the console when a user presses the keyboard shortcut. Valid values are:  
  - FOCUS_CONSOLE  
  - FOCUS_NAVIGATOR_TAB  
  - FOCUS_DETAIL_VIEW  
  - FOCUS_PRIMARY_TAB_PANEL  
  - FOCUS_SUBTAB_PANEL  
  - FOCUS_LIST_VIEW  
  - FOCUS_FIRST_LIST_VIEW  
  - FOCUS_SEARCH_INPUT  
  - MOVE_LEFT  
  - MOVE_RIGHT  
  - UP_ARROW  
  - DOWN_ARROW  
  - OPEN_TAB_SCROLLER_MENU  
  - OPEN_TAB  
  - CLOSE_TAB  
  - ENTER  
  - EDIT  
  - SAVE  
  For a list and description of the default keyboard shortcuts, see “Default Keyboard Shortcuts for a Salesforce Console in Salesforce Classic” in the Salesforce Help. |
| active     | boolean    | Required. Indicates whether the keyboard shortcut is active (true) or not (false). |
| keyCommand | string     | Required. The combination of keys a user presses to trigger the keyboard shortcut. Keyboard shortcuts aren’t case-sensitive, but they display as uppercase on setup pages in the Salesforce user interface so that they’re easier to read.  
  Each key command can include up to four modifier keys followed by one non-modifier key. Modifier and non-modifier keys are separated by the + key. Modifier keys can occur in any order, but you must place non-modifier keys at the end of the key command sequence. For example, SHIFT+CTRL+ALT+META +A.  
  Valid modifier keys are:  
  - SHIFT  
  - CTRL  
  - ALT  
  - META (represents the COMMAND key on Macs) |
Valid non-modifier keys are letters A through Z and numbers 0 through 9. Other valid keys are:

- TAB
- ENTER
- PAUSE/BREAK
- CAPS LOCK
- ESC
- SPACE
- PAGE UP
- PAGE DOWN
- END
- HOME
- LEFT ARROW
- UP ARROW
- RIGHT ARROW
- DOWN ARROW
- PRINT SCREEN
- INSERT
- DELETE
- RIGHT WINDOW
- NUMPAD 0
- NUMPAD 1
- NUMPAD 2
- NUMPAD 3
- NUMPAD 4
- NUMPAD 5
- NUMPAD 6
- NUMPAD 7
- NUMPAD 8
- NUMPAD 9
- MULTIPLY
- ADD
- SUBTRACT
- DECIMAL POINT
- DIVIDE
- F1
- F2
- F3
### KeyboardShortcuts

Represents keyboard shortcuts assigned to a Salesforce console app. Required if `isServiceCloudConsole` is `true`. Available in API version 28.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customShortcuts</td>
<td>CustomShortcut[]</td>
<td>Represents custom keyboard shortcuts assigned to a Salesforce console app. Before you can create custom shortcuts, a developer must define the shortcut’s action with the <code>addEventListener()</code> method in the Salesforce Console Integration Toolkit. You can’t create keyboard shortcuts for actions performed outside of the console. In API version 42.0, this field was renamed from <code>customShortcut</code> to <code>customShortcuts</code>.</td>
</tr>
<tr>
<td>defaultShortcuts</td>
<td>DefaultShortcut[]</td>
<td>Represents default keyboard shortcuts assigned to a Salesforce console app. Once you enable keyboard shortcuts for a console, several default</td>
</tr>
</tbody>
</table>
shortcuts are available for customization. These include opening and closing tabs, moving between tabs, and saving records.

For a list and description of the default keyboard shortcuts, see “Default Keyboard Shortcuts for a Salesforce Console in Salesforce Classic” in the Salesforce Help.

In API version 42.0, this field was renamed from defaultShortcut to defaultShortcuts.

**ListPlacement**

Represents how lists display in a Salesforce console app. Required if isServiceCloudConsole is true. Available in API version 25.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>int</td>
<td>Height of the list in pixels or percentage. Required if location is top.</td>
</tr>
<tr>
<td>location</td>
<td>string</td>
<td>Required. Location of the list on the screen. Valid values are: full, top, left</td>
</tr>
<tr>
<td>units</td>
<td>string</td>
<td>Required. Represents if height or width is in pixels or percentage.</td>
</tr>
<tr>
<td>width</td>
<td>int</td>
<td>Width of the list in pixels or percentage. Required if location is left.</td>
</tr>
</tbody>
</table>

**LiveAgentConfig**

Represents your organization’s settings for using Chat in the Salesforce Console.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableLiveChat</td>
<td>boolean</td>
<td>Specifies whether Chat is enabled in your organization (true) or not (false).</td>
</tr>
<tr>
<td>openNewAccountSubtab</td>
<td>boolean</td>
<td>Specifies whether to open a new Account subtab in a Salesforce console app automatically (true) or not (false) when an agent accepts a chat.</td>
</tr>
<tr>
<td>openNewCaseSubtab</td>
<td>boolean</td>
<td>Specifies whether to open a new Case subtab in a Salesforce console app automatically (true) or not (false) when an agent accepts a chat.</td>
</tr>
<tr>
<td>openNewContactSubtab</td>
<td>boolean</td>
<td>Specifies whether to open a new Contact subtab in a Salesforce console app automatically (true) or not (false) when an agent accepts a chat.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>openNewLeadSubtab</td>
<td>boolean</td>
<td>Specifies whether to open a new Lead subtab in a Salesforce console app automatically (true) or not (false) when an agent accepts a chat.</td>
</tr>
<tr>
<td>openNewVFPageSubtab</td>
<td>boolean</td>
<td>Specifies whether to open a new Visualforce page as a subtab in a Salesforce console app automatically (true) or not (false) when an agent accepts a chat.</td>
</tr>
<tr>
<td>pageNamesToOpen</td>
<td>string [array of strings]</td>
<td>Specifies the Visualforce pages to open in subtabs when an agent accepts a chat in a Salesforce console app. This field is available in API version 42.0 and later.</td>
</tr>
<tr>
<td>showKnowledgeArticles</td>
<td>boolean</td>
<td>Specifies whether to display the Knowledge component while using Chat in a Salesforce console app (true) or not (false).</td>
</tr>
</tbody>
</table>

**PushNotification**

Represents a set of push notifications, which are visual indicators on lists and detail pages that show when a record or field has changed during a user’s session. Available for use if isServiceCloudConsole is true. Available in API version 28.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldNames</td>
<td>string ]</td>
<td>Required. The name of the field, or fields, that trigger push notifications for the selected object.</td>
</tr>
<tr>
<td>objectName</td>
<td>string</td>
<td>Required. Name of the object that triggers push notifications.</td>
</tr>
</tbody>
</table>

**ServiceCloudConsoleConfig**

Represents configuration settings for a Salesforce console app. Available in API version 42.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentList</td>
<td>AppComponentList</td>
<td>Represents custom console components (Visualforce pages) assigned to a Salesforce console app.</td>
</tr>
<tr>
<td>detailPageRefreshMethod</td>
<td>string</td>
<td>Determines how detail pages refresh in a Salesforce console app. Required if isServiceCloudConsole is true. The valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• none</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• autoRefresh</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• flag</td>
</tr>
<tr>
<td>footerColor</td>
<td>string</td>
<td>Determines the footer color in a Salesforce console app. Specify the color with a hexadecimal code, such as #0000FF for blue.</td>
</tr>
<tr>
<td>headerColor</td>
<td>string</td>
<td>Determines the header color in a Salesforce console app. Specify the color with a hexadecimal code, such as #0000FF for blue.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>keyboardShortcuts</td>
<td>KeyboardShortcuts</td>
<td>Represents the keyboard shortcuts for a Salesforce console app. Keyboard shortcuts let users perform actions by pressing a combination of keys instead of having to use a mouse.</td>
</tr>
<tr>
<td>listPlacement</td>
<td>ListPlacement</td>
<td>Represents how lists display in a Salesforce console app. Required if isServiceCloudConsole is true.</td>
</tr>
</tbody>
</table>
| listRefreshMethod     | string                | Determines how lists refresh in a Salesforce console app. Required if isServiceCloudConsole is true. The valid values are:  
  • none  
  • refreshList  
  • refreshListRows |
| liveAgentConfig       | LiveAgentConfig       | Represents the configurations for using Chat in the Salesforce Console.        |
| primaryTabColor       | string                | Determines the primary tab color in a Salesforce console app. Specify the color with a hexadecimal code, such as #0000FF for blue. |
| pushNotifications     | PushNotification[]    | Represents push notifications for a Salesforce console app. Push notifications are visual indicators on lists and detail pages that show when a record or field has changed during a user’s session. For example, assume that two support agents are working on the same case. If one agent changes the Priority, a push notification displays to the other agent so the agent notices the change and doesn’t duplicate the effort. |
| tabLimitConfig        | TabLimitConfig        | Represents the maximum number of primary tabs and subtabs allowed in one Salesforce console session. Required if enableTabLimits is true. |
| whiteListedDomains    | string[]              | Any external domains that users can access from within a Salesforce console app. For example, www.yourdomain.com. |

**TabLimitConfig**

Represents the maximum number of primary tabs and subtabs allowed in one Salesforce console session. Required if enableTabLimits is true. Available in API version 36.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| maxNumberOfPrimaryTabs             | string     | The maximum number of primary tabs allowed in one console session. Valid values are:  
  • 5  
  • 10  
  • 20  
  • 30 |
### CustomApplicationMetadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxNumberOfSubTabs</td>
<td>string</td>
<td>The maximum number of subtabs allowed in one console session. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 15</td>
</tr>
</tbody>
</table>

### WorkspaceMapping

Represents how records for a specific tab open in a Salesforce console app. Required for each tab specified in the CustomApplication. Available in API version 25.0 and later for Salesforce Classic console apps. Available in API version 41.0 and later for Lightning console apps.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>string</td>
<td>The name of the field that specifies the primary tab in which to display tab as a subtab. If not specified, tab opens as a primary tab.</td>
</tr>
<tr>
<td>tab</td>
<td>string</td>
<td>Required. Name of the tab.</td>
</tr>
</tbody>
</table>

### Retrieving Apps

To retrieve apps in your organization, use the CustomApplication type name in the package.xml manifest file. You can either retrieve all apps or specify which apps to retrieve in the types section of package.xml.

To retrieve all apps in your organization—custom and standard apps, specify the wildcard character (*), as follows.

```xml
<types>
  <members>*</members>
  <name>CustomApplication</name>
</types>
```

**Note:** In API version 29.0 and earlier, use of the wildcard returns only all custom applications but not standard applications.

To retrieve a custom app, specify the app name.

```xml
<types>
  <members>MyCustomApp</members>
  <name>CustomApplication</name>
</types>
```

To retrieve a standard app, add the standard__ prefix to the app name. For example, to retrieve the Chatter standard app, specify standard__Chatter.

```xml
<types>
  <members>standard__Chatter</members>
  <name>CustomApplication</name>
</types>
```
To retrieve an app that is part of an installed package, add the package namespace prefix followed by two underscores and the app name. For example, if the package namespace is `myInstalledPackageNS` and the app name is `PackageApp`, specify `myInstalledPackageNS__PackageApp`, as follows.

```xml
<types>
  <members>myInstalledPackageNS__PackageApp</members>
  <name>CustomApplication</name>
</types>
```

Declarative Metadata Sample Definition

The following is the definition of a custom app:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomApplication xmlns="http://soap.sforce.com/2006/04/metadata">
  <defaultLandingTab>Myriad_Publishing__c</defaultLandingTab>
  <description>App to manage Myriad Publishing</description>
  <label>Myriad</label>
  <logo>MyriadFolder/Myriad_Logo.jpg</logo>
  <tab>standard-Chatter</tab>
  <tab>standard-File</tab>
  <tab>Myriad_Publishing__c</tab>
  <tab>standard-report</tab>
  <tab>standard-Dashboard</tab>
</CustomApplication>
```

The following is a definition of a standard app (Chatter):

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomApplication xmlns="http://soap.sforce.com/2006/04/metadata">
  <defaultLandingTab>standard-home</defaultLandingTab>
  <label>Collaboration</label>
  <tab>standard-Chatter</tab>
  <tab>standard-UserProfile</tab>
  <tab>standard-OtherUserProfile</tab>
  <tab>standard-CollaborationGroup</tab>
  <tab>standard-File</tab>
</CustomApplication>
```

Declarative Metadata Sample Definition—Salesforce Console

The following is the definition of a custom app where `isServiceCloudConsole` is true:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomApplication xmlns="http://soap.sforce.com/2006/04/metadata">
  <consoleConfig>
    <componentList>
      <alignment>left</alignment>
      <components>MyComponent</components>
    </componentList>
    <detailPageRefreshMethod>autoRefresh</detailPageRefreshMethod>
    <keyboardShortcuts>
      <customShortcuts>
        <action>MyCustomShortcutAction</action>
      </customShortcuts>
    </keyboardShortcuts>
  </consoleConfig>
</CustomApplication>
```
<active>true</active>
<keyCommand>X</keyCommand>
<description>Custom Shortcut example</description>
<eventName>myCustomShortcutExample</eventName>
</customShortcuts>
<defaultShortcuts>
  <action>FOCUS_CONSOLE</action>
  <active>true</active>
  <keyCommand>ESC</keyCommand>
</defaultShortcuts>
<defaultShortcuts>
  <action>FOCUS_NAVIGATOR_TAB</action>
  <active>true</active>
  <keyCommand>V</keyCommand>
</defaultShortcuts>
<defaultShortcuts>
  <action>FOCUS_DETAIL_VIEW</action>
  <active>true</active>
  <keyCommand>SHIFT+S</keyCommand>
</defaultShortcuts>
<defaultShortcuts>
  <action>FOCUS_PRIMARY_TAB_PANEL</action>
  <active>true</active>
  <keyCommand>P</keyCommand>
</defaultShortcuts>
<defaultShortcuts>
  <action>FOCUS_SUBTAB_PANEL</action>
  <active>true</active>
  <keyCommand>S</keyCommand>
</defaultShortcuts>
<defaultShortcuts>
  <action>FOCUS_LIST_VIEW</action>
  <active>true</active>
  <keyCommand>N</keyCommand>
</defaultShortcuts>
<defaultShortcuts>
  <action>FOCUS_FIRST_LIST_VIEW</action>
  <active>true</active>
  <keyCommand>SHIFT+F</keyCommand>
</defaultShortcuts>
<defaultShortcuts>
  <action>FOCUS_SEARCH_INPUT</action>
  <active>true</active>
  <keyCommand>R</keyCommand>
</defaultShortcuts>
<defaultShortcuts>
  <action>MOVE_LEFT</action>
  <active>true</active>
  <keyCommand>LEFT ARROW</keyCommand>
</defaultShortcuts>
<defaultShortcuts>
  <action>MOVE_RIGHT</action>
  <active>true</active>
  <keyCommand>RIGHT ARROW</keyCommand>
<defaultShortcuts><action>UP_ARROW</action><active>true</active><keyCommand>UP ARROW</keyCommand></defaultShortcuts><defaultShortcuts><action>DOWN_ARROW</action><active>true</active><keyCommand>DOWN ARROW</keyCommand></defaultShortcuts><defaultShortcuts><action>OPEN_TAB_SCROLLER_MENU</action><active>true</active><keyCommand>D</keyCommand></defaultShortcuts><defaultShortcuts><action>OPEN_TAB</action><active>true</active><keyCommand>T</keyCommand></defaultShortcuts><defaultShortcuts><action>CLOSE_TAB</action><active>true</active><keyCommand>C</keyCommand></defaultShortcuts><defaultShortcuts><action>ENTER</action><active>true</active><keyCommand>ENTER</keyCommand></defaultShortcuts><defaultShortcuts><action>EDIT</action><active>true</active><keyCommand>E</keyCommand></defaultShortcuts><defaultShortcuts><action>SAVE</action><active>true</active><keyCommand>CTRL+S</keyCommand></defaultShortcuts></keyboardShortcuts><listPlacement><location>left</location><units>percent</units><width>20</width></listPlacement><listRefreshMethod>refreshList</listRefreshMethod><pushNotifications><fieldNames>CreatedBy</fieldNames><objectName>Campaign</objectName></pushNotifications><pushNotifications><fieldNames>CustomField1__c</fieldNames></pushNotifications>
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:

CustomTab
CustomApplicationComponent

Represents a custom console component (Visualforce page) assigned to a CustomApplication that is marked as a Salesforce console. Custom console components extend the capabilities of Salesforce console apps. See “Customize a Console with Custom Components in Salesforce Classic” in the Salesforce online help.

File Suffix and Directory Location

Custom application components have the suffix .customApplicationComponent and are stored in the customApplicationComponents folder.

Version

Custom applications are available in API version 25.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>buttonIconUrl</td>
<td>string</td>
<td>The address of a page that hosts an icon for the button.</td>
</tr>
<tr>
<td>buttonStyle</td>
<td>string</td>
<td>The inline style used to define how the button looks.</td>
</tr>
<tr>
<td>buttonText</td>
<td>string</td>
<td>The label on the button used to launch the custom console component.</td>
</tr>
<tr>
<td>buttonWidth</td>
<td>int</td>
<td>The pixel width of the button as it should display in the Salesforce console.</td>
</tr>
<tr>
<td>height</td>
<td>int</td>
<td>The pixel height of the window used to display the custom console component.</td>
</tr>
<tr>
<td>isHeightFixed</td>
<td>boolean</td>
<td>Required. Indicates whether users can change the custom console component height (false) or not (true).</td>
</tr>
<tr>
<td>isHidden</td>
<td>boolean</td>
<td>Required. Indicates whether the custom console component is hidden from users (true) or not (false).</td>
</tr>
<tr>
<td>isWidthFixed</td>
<td>boolean</td>
<td>Required. Indicates whether users can change the component width (false) or not (true).</td>
</tr>
<tr>
<td>visualforcePage</td>
<td>string</td>
<td>Required. Name of the Visualforce page that represents the custom console component.</td>
</tr>
<tr>
<td>width</td>
<td>int</td>
<td>The pixel width of the window used to display the custom console component.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is the definition of a custom application component:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomApplicationComponent xmlns="http://soap.sforce.com/2006/04/metadata">
    <buttonIconUrl>http://www.salesforce.com</buttonIconUrl>
    <buttonStyle>buttonStyleCSS</buttonStyle>
    <buttonText>buttonText</buttonText>
    <buttonWidth>200</buttonWidth>
    <height>200</height>
    <isHeightFixed>false</isHeightFixed>
    <isHidden>false</isHidden>
    <isWidthFixed>false</isWidthFixed>
    <visualforcePage>MyVisualforcePage</visualforcePage>
    <width>50</width>
</CustomApplicationComponent>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

CustomFeedFilter

Represents a custom feed filter that limits the feed view to feeds from the Cases object. The custom feed filter shows only feed items that satisfy the criteria specified in the CustomFeedFilter definition. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

CustomFeedFilter components have the suffix .feedFilter and are stored in the feedFilters folder.

Version

CustomFeedFilter components are available in API version 35.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the custom feed filter. For example, specify what feed items that filter shows.</td>
</tr>
<tr>
<td>criteria</td>
<td>FeedFilterCriterion []</td>
<td>The criterion that defines which feed items are shown when the filter is applied. The feed filter displays all feed items that satisfy the criteria.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The API label of the custom feed filter.</td>
</tr>
</tbody>
</table>
FeedFilterCriterion

Represents the conditions that a feed item must satisfy to be displayed when a feed filter is applied.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>feedItemType</td>
<td>FeedItemType (enumeration of type string)</td>
<td>Required. The type of feed items that the filter shows. The feed item type can be one of the following values: AttachArticleEvent, CallLogPost, CanvasPost, CaseCommentPost, ChangeStatusPost, ChatTranscriptPost, ContentPost, CreateRecordEvent, EmailMessageEvent, LinkPost, MilestoneEvent, QuestionPost, PollPost, ReplyPost, SocialPost, TextPost</td>
</tr>
<tr>
<td>feedItemVisibility</td>
<td>FeedItemVisibility (enumeration of type string)</td>
<td>The visibility of feed items that the filter shows. For example, you can show only poll posts that are visible internally. Valid values are: AllUsers, InternalUsers</td>
</tr>
<tr>
<td>relatedSObjectType</td>
<td>string</td>
<td>The API name of the object that the feed item refers to. This field is typically used with the CreateRecordEvent feed item type. For example, a feed filter can show CreateRecordEvent feed items for the Cases object.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example of a CustomFeedFilter component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomFeedFilter xmlns="http://soap.sforce.com/2006/04/metadata">
  <criteria>
    <feedItemType>CreateRecordEvent</feedItemType>
    <relatedSObjectType>MyCO01__c</relatedSObjectType>
  </criteria>
  <criteria>
    <feedItemType>CreateRecordEvent</feedItemType>
    <relatedSObjectType>Case</relatedSObjectType>
  </criteria>
  <criteria>
    <feedItemType>PollPost</feedItemType>
    <feedItemVisibility>InternalUsers</feedItemVisibility>
  </criteria>
  <label>Sample Custom Feed Filter</label>
</CustomFeedFilter>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>myCaseFeedFilter</members>
    <name>CustomFeedFilter</name>
  </types>
  <version>46.0</version>
</Package>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

CustomHelpMenuSection

Represents the section of the Lightning Experience help menu that the admin added to display custom, org-specific help resources for the org. The custom section contains help resources added by the admin. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

CustomHelpMenuSection components have the suffix .customHelpMenuSection and are stored in the customHelpMenuSections folder.

Version

CustomHelpMenuSection components are available in API version 45.0 and later.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customHelpMenuItems</td>
<td>CustomHelpMenuItems[]</td>
<td>Items included in the custom section. Specify up to 15 items.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Name of the custom section. Only one custom section can be added to the Lightning Experience help menu. Specify up to 80 characters.</td>
</tr>
</tbody>
</table>

CustomHelpMenuItems

Items included in the custom section. Specify up to 15 items.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>linkURL</td>
<td>string</td>
<td>Required. The URL for the resource.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The name of the resource. Specify up to 100 characters.</td>
</tr>
<tr>
<td>sortOrder</td>
<td>int</td>
<td>Required. The order of the item within the custom section. Valid values are 1 through 15.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a CustomHelpMenuSection component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomHelpMenuSection xmlns="http://soap.sforce.com/2006/04/metadata">
  <masterLabel>MyOrgCustomHelp</masterLabel>
  <customHelpMenuItems>
    <linkUrl>https://www.yourcompanyhelp.com/gettingstarted</linkUrl>
    <masterLabel>Getting Started</masterLabel>
    <sortOrder>1</sortOrder>
  </customHelpMenuItems>
  <customHelpMenuItems>
    <linkUrl>https://www.yourcompanyhelp.com/features</linkUrl>
    <masterLabel>Feature to Start Using Right Away</masterLabel>
    <sortOrder>2</sortOrder>
  </customHelpMenuItems>
  <customHelpMenuItems>
    <linkUrl>https://www.yourcompanyhelp.com/salestips</linkUrl>
    <masterLabel>Tips for Sales Team Members</masterLabel>
    <sortOrder>3</sortOrder>
  </customHelpMenuItems>
</CustomHelpMenuSection>
```

The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
```

280
CustomLabels

The CustomLabels metadata type allows you to create custom labels that can be localized for use in different languages, countries, and currencies.

This type extends the Metadata metadata type and inherits its fullName field. Custom labels are custom text values, up to 1,000 characters in length, that can be accessed from Apex classes or Visualforce pages. For more information, see “Custom Labels” in Salesforce Help.

Declarative Metadata File Suffix and Directory Location

Master custom label values are stored in the CustomLabels.labels file. Translations for custom labels can be retrieved through Translations in Metadata API. Translations are stored in files under the translations folder with the name format of localeCode.translation, where localeCode is the locale code of the translation language. The supported locale codes are listed in Language on page 898.

Version

CustomLabels components are available in API version 14.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The name of the custom label bundle. Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See createMetadata() to see an example of this field specified for a call.</td>
</tr>
<tr>
<td>labels</td>
<td>CustomLabel[]</td>
<td>A list of custom labels.</td>
</tr>
</tbody>
</table>

CustomLabel

This metadata type represents a custom label. This type extends the Metadata metadata type and inherits its fullName field.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>categories</td>
<td>string</td>
<td>A comma-separated list of categories for the label. This field can be used in filter criteria when creating custom label list views. Maximum of 255 characters.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The name of the custom label. Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See createMetadata() to see an example of this field specified for a call.</td>
</tr>
<tr>
<td>language</td>
<td>string</td>
<td>Required. The language of the translated custom label.</td>
</tr>
<tr>
<td>protected</td>
<td>boolean</td>
<td>Required. Indicates whether this component is protected (true) or not (false). Protected components cannot be linked to or referenced by components created in the installing organization.</td>
</tr>
<tr>
<td>shortDescription</td>
<td>string</td>
<td>Required. An easily recognizable term to identify this custom label. This description is used in merge fields.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Required. The translated custom label. Maximum of 1000 characters.</td>
</tr>
</tbody>
</table>

**Usage**

Use CustomLabels with the wildcard character (*) for members in the package.xml manifest file to retrieve all custom labels that are defined in your organization. CustomLabels doesn’t support retrieving one or more custom labels by name. To retrieve specific labels by name, use CustomLabel and specify the label names as members.

**Declarative Metadata Sample Definition**

A sample XML definition of a custom label components is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomLabels xmlns="http://soap.sforce.com/2006/04/metadata">
  <labels>
    <fullName>quoteManual</fullName>
    <value>This is a manual quote.</value>
    <language>en_US</language>
    <protected>false</protected>
    <shortDescription>Manual Quote</shortDescription>
  </labels>
  <labels>
    <fullName>quoteAuto</fullName>
    <value>This is an automatically generated quote.</value>
    <language>en_US</language>
    <protected>false</protected>
    <shortDescription>Automatic Quote</shortDescription>
  </labels>
</CustomLabels>
```

This is a sample manifest file for retrieving all custom labels in the organization by using the CustomLabels type.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
```

282
This is a sample manifest file for retrieving two custom labels by name. Notice it uses the CustomLabel singular type.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <fullName>MyPkg</fullName>
  <types>
    <members>quoteManual</members>
    <members>quoteAuto</members>
    <name>CustomLabel</name>
  </types>
  <version>46.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- Translations

**Custom Metadata Types (CustomObject)**

Represents the metadata associated with a custom metadata type.

For more information, see the Custom Metadata Types Implementation Guide.

**File Suffix and Directory Location**

A custom metadata type is defined as a custom object and is stored in the objects folder. Custom metadata types have a suffix of __mdt (instead of __c for custom objects). Custom metadata type field names have a suffix of __c, like other custom fields. Custom metadata type field names must be dot-qualified with the name of the custom metadata type to which they belong.

Names of custom metadata types must be unique within their namespace. All custom metadata types belong to the CustomMetadata namespace and can optionally belong to a second namespace. In your organization, you can use custom metadata types with your namespace and also other organizations’ namespaces.

**Version**

Custom metadata type components are available in API version 31.0 and later.
Special Access Rules

To create custom metadata types, you must have the “Author Apex” permission. Apex code can create, read, and update (but not delete) custom metadata records, as long as the metadata is subscriber-controlled and visible from within the code’s namespace. You can edit records in memory but not upsert or delete them. Apex code can deploy custom metadata records, but not via a DML operation. Moreover, DML operations aren’t allowed on custom metadata in the Partner or Enterprise APIs. Customers who install a managed custom metadata type can’t add new custom fields to it. With unpackaged metadata, both developer-controlled and subscriber-controlled access behave the same: like subscriber-controlled access. Refer to Trust, but Verify: Apex Metadata API and Security to learn more.

Note: Audit fields (CreatedDate, CreatedBy, LastModifiedDate, LastModifiedBy, SystemModStamp) remain uneditable.

Fields

Custom metadata types can contain the following CustomObject fields.

To make the fields on your custom metadata types unique and indexable, mark your fields as Unique and ExternalId.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the custom metadata type. This field can contain a maximum of 1,000 characters.</td>
</tr>
<tr>
<td>fields</td>
<td>CustomField[]</td>
<td>Represents one or more custom fields in the custom metadata type.</td>
</tr>
<tr>
<td>gender</td>
<td>Gender</td>
<td>Indicates the gender of the noun that represents the object. This field is used for languages where words need different treatment depending on their gender.</td>
</tr>
</tbody>
</table>

Protected | boolean | When a custom metadata type is released in a managed package, access is limited in specific ways.  
• Code that’s in the same managed package as custom metadata records can read the records.  
• Code that’s in the same managed package as custom metadata types can read the records that belong to that type.  
• Code that’s in a managed package that doesn’t contain either the type or the protected record can’t read the protected records.  
• Code that the subscriber creates and code that’s in an unmanaged package can’t read the protected records.  
• The developer can modify protected records with a package upgrade or by using the Metadata Apex classes (if the Apex code is in the same namespace as either the records or their type). The subscriber can’t read or modify protected records. The developer name of a protected record can’t be changed after release.  
• The subscriber can’t create records of a protected type. |
Records that are hidden by these access rules are also unavailable to REST, SOAP, SOQL, and Setup.

A label that represents the object throughout the Salesforce Setup user interface. Custom metadata types are visible only through the recently used objects list on the Lightning Platform Home Page and in the packaging user interface.

The plural version of the label value.

Indicates whether the noun starts with a vowel, a consonant, or a special character. This field is used for languages where words need different treatment depending on their first character.

When this field is present, this component is not a custom object, but a custom setting or custom metadata type. This field returns the visibility of the custom setting or custom metadata type. The following values are valid:

- **Public**—If the custom setting or custom metadata type is packaged, it’s accessible to all subscribing organizations.
- **Protected**—If the custom setting or custom metadata type is in a managed package, it’s only accessible to the developer organization; subscribing organizations can’t access it.

The default value is **Public**.

---

### Declarative Metadata Sample Definition

In this example, Picklists R Us creates its Reusable Picklist custom metadata type by deploying a file in the objects folder, named `ReusablePicklistOption__mdt.object`, with these contents.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
    <fields>
        <fullName>AlphaSort__c</fullName>
        <defaultValue>false</defaultValue>
        <externalId>false</externalId>
        <label>Sorted Alphabetically</label>
        <type>Checkbox</type>
    </fields>
    <label>Reusable Picklist</label>
    <pluralLabel>Reusable Picklist</pluralLabel>
    <visibility>Public</visibility>
</CustomObject>
```
This excerpt from a package.xml file shows the use of dot notation and the __mdt suffix. If you’re using a namespace, for example picklist1234, the full name of ReusablePicklistOption__mdt would be picklist1234__ReusablePicklistOption__mdt.

```
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  ...
  <types>
    <members>PicklistTest__c.PicklistTestField__c</members>
    <members>ReusablePicklistOption__mdt.Picklist__c</members>
    <members>ReusablePicklistOption__mdt.SortOrder__c</members>
    <members>PicklistUsage__mdt.Field__c</members>
    <members>PicklistUsage__mdt.Picklist__c</members>
    <members>PicklistUsage__mdt.SObjectType__c</members>
    <members>ReusablePicklist__mdt.AlphaSort__c</members>
    <name>CustomField</name>
  </types>
  ...
  <types>
    <members>PicklistTest__c</members>
    <members>ReusablePicklistOption__mdt</members>
    <members>PicklistUsage__mdt</members>
    <members>ReusablePicklist__mdt</members>
    <name>CustomObject</name>
  </types>
  ...
  <version>46.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**CustomMetadata**

Represents a record of a custom metadata type.

This type extends the Metadata metadata type and inherits its fullName field.

**File Suffix and Directory Location**

CustomMetadata components have the suffix .md and are stored in the customMetadata folder. Unlike custom metadata types, custom metadata records don’t have a double-underscore suffix. Custom metadata record names are prepended with their custom metadata type name, excluding the __mdt suffix but including the namespace of any types in an installed managed package.

**Version**

CustomMetadata components are available in API version 31.0 and later.
Special Access Rules

To create custom metadata records, you must have the “Customize Application” permission.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the custom metadata record. This field can contain a maximum of 1,000 characters.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>A label that represents the object throughout the Salesforce Setup user interface. Custom metadata records are currently visible only through the packaging user interface.</td>
</tr>
<tr>
<td>protected</td>
<td>boolean</td>
<td>Boolean. Indicates whether the type is protected (true) or not (false).</td>
</tr>
<tr>
<td>values</td>
<td>CustomMetadataValue[]</td>
<td>Represents one or more values for custom fields on the custom metadata record.</td>
</tr>
</tbody>
</table>

CustomMetadataValue

Represents a value for a custom field on the custom metadata record.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>Required. The non-object-qualified name of a custom field in the custom metadata type. This value corresponds to the name of a field on the custom metadata record’s custom metadata type. Include the namespace (if the type is from a managed package) and the __c suffix. The name of the custom metadata type isn’t required. For example, picklist1234__AlphaSort__c.</td>
</tr>
<tr>
<td>value</td>
<td>Any type</td>
<td>The value on a custom metadata record. Where fields are EntityDefinition and FieldDefinition, the qualified API names of the entity and the field it points to. This value can be null. For more information, see Usage on page 290.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definitions

The following is an example of a CustomMetadata component. In this example, the sample app TravelApp deploys a Planets picklist, specifies its sort order, and adds picklist items to it.
Assuming Picklists R Us’s namespace is picklist1234, to define the Planets picklist, TravelApp deploys a file in the customMetadata folder, named picklist1234__ReusablePicklist.Planets.md, with these contents. The xsi:type attribute specifies the type for the value of the AlphaSort__c checkbox field.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomMetadata xmlns="http://soap.sforce.com/2006/04/metadata"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <description>All the planets in the solar system. Does not include asteroids.</description>
    <label>Planets</label>
    <values>
        <field>picklist1234__AlphaSort__c</field>
        <value xsi:type="xsd:boolean">false</value>
    </values>
</CustomMetadata>
```

Picklists R Us creates its Reusable Picklist Option custom metadata type by deploying a file in the objects folder, named ReusablePicklist_option.object, with these contents.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
    <fields>
        <fullName>Picklist__c</fullName>
        <externalId>false</externalId>
        <label>Picklist</label>
        <length>40</length>
        <required>true</required>
        <type>Text</type>
        <unique>false</unique>
    </fields>
    <fields>
        <fullName>SortOrder__c</fullName>
        <externalId>false</externalId>
        <label>Non-Alphabetical Sort Order</label>
        <precision>3</precision>
        <scale>0</scale>
        <required>false</required>
        <type>Number</type>
        <unique>false</unique>
    </fields>
    <label>Reusable Picklist Option</label>
    <pluralLabel>Reusable Picklist Options</pluralLabel>
</CustomObject>
```

To define the Mars picklist item, TravelApp deploys a file, named picklist1234__ReusablePicklistOption.Mars.md, with these contents. This component file specifies types that apply to the ReusablePicklistOption__mdt custom fields.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomMetadata xmlns="http://soap.sforce.com/2006/04/metadata"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <label>Mars</label>
    <values>
        <field>picklist1234__Picklist__c</field>
    </values>
</CustomMetadata>
```
To define the Motel 6 picklist item, TravelApp deploys a file, named `picklist1234__ReusablePicklistOption.Motel6.md`, with these contents.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomMetadata xmlns="http://soap.sforce.com/2006/04/metadata"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <label>Motel 6</label>
    <values>
        <field>picklist1234__Picklist__c</field>
        <value xsi:type="xsd:string">Hotels</value>
    </values>
</CustomMetadata>
```

Because the `SortOrder__c` field isn’t required, this file doesn’t require a value for `SortOrder__c`. Alternatively, the file could have explicitly specified a value with `xsi:nil` to ensure that `SortOrder__c` was cleared of any previous value.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomMetadata xmlns="http://soap.sforce.com/2006/04/metadata"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <label>Motel 6</label>
    <values>
        <field>picklist1234__Picklist__c</field>
        <value xsi:type="xsd:string">Hotels</value>
    </values>
    <values>
        <field>picklist1234__SortOrder__c</field>
        <value xsi:nil="true" />
    </values>
</CustomMetadata>
```

This excerpt from a `package.xml` file illustrates the inclusion of custom metadata types and their namespaces in custom metadata records’ names. Assume that Picklists R Us’s namespace is `picklist1234`.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<package xmlns="http://soap.sforce.com/2006/04/metadata">
    ...
    <types>
        <members>picklist1234__ReusablePicklist.Hotels</members>
        <members>picklist1234__ReusablePicklist.Planets</members>
        <members>picklist1234__ReusablePicklistOption.Bellagio</members>
        <members>picklist1234__ReusablePicklistOption.Motel6</members>
        <members>picklist1234__ReusablePicklistOption.Mercury</members>
        <members>picklist1234__ReusablePicklistOption.Venus</members>
        <members>picklist1234__ReusablePicklistOption.Earth</members>
        <members>picklist1234__PicklistUsage.BookedHotel</members>
    </types>
</package>
```
TravelApp, Inc’s package.xml file uses a wildcard to install custom metadata, as is shown in this excerpt from their package.xml file. Unless you want to deploy or retrieve specific records, using a wildcard is easier than listing all of your custom metadata records in your package.xml file.

If the custom metadata is from a managed package, the name after the dot in the package.xml file—between the two dots in the file name—is qualified by the managed package’s namespace. For example, assuming TravelApp uses the namespace travelApp1234, the first member element in the TravelApp package.xml file appears to Galactic Tours as:

Here’s another example. In this case, we have an instance of custom metadata record, whose EntityDefinition field points to a custom object named SalesAgreement__c. The FieldDefinition field points to the custom field CustomerReference__c on SalesAgreement__c. You can deploy new custom metadata records and retrieve existing ones with EntityDefinition and FieldDefinition fields using qualified API names of custom and standard entities and their fields.

Usage

When specifying the value field in the CustomMetadataValue subtype, specify an appropriately typed object that’s based on your field type definition. In declarative metadata definitions for CustomMetadataValue, use the xsi:type attribute of the value element. For example, to specify a boolean value:  

<table>
<thead>
<tr>
<th>Custom metadata value</th>
<th>Custom field definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>xsi:type=&quot;xsd:boolean&quot;</td>
<td>Checkbox</td>
</tr>
<tr>
<td>xsi:type=&quot;xsd:date&quot;</td>
<td>Date</td>
</tr>
<tr>
<td>xsi:type=&quot;xsd:dateTime&quot;</td>
<td>Date/Time</td>
</tr>
<tr>
<td>xsi:type=&quot;xsd:picklist&quot;</td>
<td>Picklist</td>
</tr>
</tbody>
</table>
Custom metadata value                  Custom field definition
---                                   ------------------
  xsi:type="xsd:string"              Text
  xsi:type="xsd:string"              Phone
  xsi:type="xsd:string"              TextArea
  xsi:type="xsd:string"              URL
  xsi:type="xsd:string"              Email
  xsi:type="xsd:int"                 Number/Percent, with scale equal to 0
  xsi:type="xsd:double"              Number/Percent, with scale not equal to 0

You can also omit the xsi:type attribute. For example, `<value>true</value>`.

Although this attribute must be specified for any CustomMetadataValue, you can use an element with the xsi:nil attribute set to true to explicitly set the field's value to null. For example, `<value xsi:nil="true"/>`.

Using null field values differs from leaving out the CustomMetadataValue for a particular field entirely. If you leave out the CustomMetadataValue, the value of the field doesn't change. The field's value is null for newly deployed custom metadata records and left at its previous value for updated custom metadata records.

When you retrieve CustomMetadataValue objects, the value field of the returned object holds a value of the correct type, specified by xsi:type in the case of declarative metadata definitions.

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### CustomNotificationType

Represents the metadata associated with a custom notification type. Custom notification types allow you to send a custom desktop or mobile notification via a process or invocable API call.

For more information about custom notifications, see Custom Notification Actions. This type extends the Metadata metadata type and inherits its fullName field.

### Declarative Metadata File Suffix and Directory Location

The file suffix is .notiftype for the notification type definition. Notification types are stored in the notificationtypes directory of the corresponding package directory.

### Version

CustomNotificationType components are available in API version 46.0 and later.
### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customNotifTypeName</td>
<td>string</td>
<td>Required. Specifies a notification type name. Maximum number of characters: 80.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Specifies a general description of the notification type, which is displayed with the notification type name. Maximum number of characters: 255.</td>
</tr>
<tr>
<td>desktop</td>
<td>boolean</td>
<td>Required. Indicates whether desktop notifications are enabled (true) or not (false).</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Specifies the label for the notification type.</td>
</tr>
<tr>
<td>mobile</td>
<td>boolean</td>
<td>Required. Indicates whether mobile in-app and push notifications are enabled (true) or not (false).</td>
</tr>
</tbody>
</table>

#### Declarative Metadata Sample Definition

The following is a definition of a custom notification type.

```xml
<CustomNotificationType xmlns="http://soap.sforce.com/2006/04/metadata">
  <customNotifTypeName>Custom Notification</customNotifTypeName>
  <desktop>true</desktop>
  <masterLabel>Custom Notification</masterLabel>
  <mobile>true</mobile>
</CustomNotificationType>
```

#### Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### CustomObject

Represents a custom object that stores data unique to your org or an external object that maps to data stored outside your org. This type extends the `Metadata` metadata type and inherits its `fullName` field.

You must specify all relevant fields when you create or update a custom object. You cannot update a single field on the object. For more information about custom objects, see Administer Custom Objects in Salesforce Help.

You can also use this metadata type to work with customizations of standard objects, such as accounts. For an example, see Standard Objects on page 18.

All metadata components have a `fullName` field, which must be fully specified for any custom object.

For example, the following are fully specified names for a standard object and a custom object respectively:

- `Account`
- `MyCustomObject__c`
And the following is a fully specified name for an external object:

```
MyExternalObject__x
```

For sample Java code that creates a custom object, see Step 3: Walk Through the Java Sample Code on page 7.

### Declarative Metadata File Suffix and Directory Location

Custom object names are automatically appended with __c. The file suffix is .object for the custom object or standard object file.

External object names are automatically appended with __x. The file suffix is .object for the external object file.

Custom, standard, and external objects are stored in the `objects` folder in the corresponding package directory.

**Note:** Retrieving a component of this metadata type in a project makes the component appear in any Profile and PermissionSet components that are retrieved in the same package.

### Version

Custom objects are available in API version 10.0 and later. External objects are available in API version 32.0 and later.

### Fields

Unless otherwise noted, all fields are createable, filterable, and nillable.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionOverrides</td>
<td>ActionOverride[]</td>
<td>A list of action overrides on the object. This field is available in API version 18.0 and later.</td>
</tr>
<tr>
<td>allowInChatterGroups</td>
<td>boolean</td>
<td>Indicates whether records of this custom object type can be added to Chatter groups. This field is available in API version 34.0 and later.</td>
</tr>
<tr>
<td>businessProcesses</td>
<td>BusinessProcess[]</td>
<td>A list of business processes associated with the object. This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>compactLayoutAssignment</td>
<td>string</td>
<td>The compact layout assigned to the object. This field is available in API version 29.0 and later. This field is available for external objects in API version 42.0 and later.</td>
</tr>
<tr>
<td>compactLayouts</td>
<td>CompactLayout[]</td>
<td>A list of compact layouts associated with the object. This field is available in API version 29.0 and later. This field is available for external objects in API version 42.0 and later.</td>
</tr>
<tr>
<td>customHelp</td>
<td>string</td>
<td>The s-control that contains the help content if the object has customized help content. This field is available in API version 14.0 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>customHelpPage</td>
<td>string</td>
<td>The Visualforce page that contains the help content if the object has customized help content. This field is available in API version 16.0 and later.</td>
</tr>
</tbody>
</table>
| customSettingsType      | CustomSettingsType (enumeration of type string) | When this field is present, this component is not a custom object, but a custom setting. This field returns the type of custom setting. The following string values are valid:  
  • **List**—static data stored in cache, accessed as part of your application, and available org-wide.  
  • **Hierarchy**—static data stored in cache, accessed as part of your application, and available based on a hierarchy of user, profile, or org. This value is the default.  
  This field is available in API version 17.0 and later. |
| customSettingsVisibility| CustomSettingsVisibility (enumeration of type string) | When this field is present, this component is not a custom object, but a custom setting. This field returns the visibility of the custom setting. The following string values are valid:  
  • **Public**—if the custom setting is packaged, it is accessible to all subscribing orgs.  
  • **Protected**—if the custom setting is in a managed package, it is only accessible to the developer org. Subscribing orgs can't access it. This value is the default.  
  This field is available in API versions 17.0 through 33.0. In versions 34.0 and later, use the visibility field instead of this field. |
<p>| dataStewardGroup        | string                         | Deprecated in API version 46.0 and later.                                                                                                                                                                 |
| dataStewardUser         | string                         | Deprecated in API version 46.0 and later.                                                                                                                                                                 |
| deploymentStatus        | DeploymentStatus (enumeration of type string) | Indicates the deployment status of the object.                                                                                                                                                            |
| deprecated              | boolean                        | Reserved for future use.                                                                                                                                                                                  |
| description             | string                         | A description of the object. Maximum of 1000 characters.                                                                                                                                                 |
| enableActivities        | boolean                        | Indicates whether the object is enabled for activities (true) or not (false). Not available for external objects.                                                                                         |
| enableBulkApi           | boolean                        | When enabled, the object is classified as an Enterprise Application object for usage tracking. When enabled, enableSharing and enableStreamingApi must also be enabled. This field is available in API version 31.0 and later. |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableDivisions</td>
<td>boolean</td>
<td>Indicates whether the object is enabled for divisions (true) or not (false). For more information about the Division object, see the SOAP API Developer Guide.</td>
</tr>
<tr>
<td>enableEnhancedLookup</td>
<td>boolean</td>
<td>Indicates whether the object is enabled for enhanced lookups (true) or not (false). In API version 28.0 and later, this field can also be used for the Account, Contact, and User objects. Enhanced lookups provide an updated lookup dialog interface that lets users filter, sort, and page through search results and customize search result columns. For more information about enhanced lookups, see “Enable Enhanced Lookups” in Salesforce Help.</td>
</tr>
<tr>
<td>enableFeeds</td>
<td>boolean</td>
<td>Indicates whether the object is enabled for feed tracking (true) or not (false). For more information, see “Customize Chatter Feed Tracking” in Salesforce Help. This field is available in API version 18.0 and later.</td>
</tr>
<tr>
<td>enableHistory</td>
<td>boolean</td>
<td>Indicates whether the object is enabled for history tracking (true) or not (false). Also available for standard objects in API version 29.0 and later. History tracking on the Account object includes person account history tracking.</td>
</tr>
<tr>
<td>enableReports</td>
<td>boolean</td>
<td>Indicates whether the object is enabled for reports (true) or not (false). Support for external objects is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>enableSearch</td>
<td>boolean</td>
<td>Indicates whether the object’s records can be found via SOSL and Salesforce searches. Corresponds to Allow Search in the user interface. By default, search is disabled for new custom objects. This field is available for custom objects in API version 35.0 and later. By default, search is disabled for new external objects. However, you can validate and sync an external data source to automatically create external objects. Syncing always enables search on the external object when search is enabled on the external data source, and vice versa. This field is available for external objects in API version 37.0 and later.</td>
</tr>
<tr>
<td>enableSharing</td>
<td>boolean</td>
<td>When enabled, the object is classified as an Enterprise Application object for usage tracking. When enabled, enableBulkApi and enableStreamingApi must also be enabled. This field is available in API version 31.0 and later.</td>
</tr>
</tbody>
</table>

CustomObjectMetadata Types

295
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableStreamingApi</td>
<td>boolean</td>
<td>When enabled, the object is classified as an Enterprise Application object for usage tracking. When enabled, enableBulkApi and enableSharing must also be enabled. This field is available in API version 31.0 and later.</td>
</tr>
</tbody>
</table>
| eventType                  | PlatformEventType (enumeration of type string)  | This field applies only to platform events. Indicates the event type. The values are:  
  - HighVolume—For a high-volume platform event.  
  - StandardVolume—Deprecated. Creating a platform event with this event type is not supported and returns an error.                                                                                                                                                                               |
| externalDataSource         | string                                          | Required and available for external objects only. The name of the external data source that stores the data for the external object. The data source is represented by the ExternalDataSource component. This field is available in API version 32.0 and later.                                                                                             |
| externalName               | string                                          | Required and available for external objects only. The name of the table in the external data source that contains the data for the external object. This field is available in API version 32.0 and later.                                                                                                                   |
| externalRepository         | string                                          | Available for Salesforce Connect external objects only. Corresponds to Display URL Reference Field in the user interface. The external object’s Display URL standard field values are automatically generated from the external system. For example, with the OData 2.0 adapter for Salesforce Connect, the value is based on the link href that’s defined on the OData producer. You can override the default values with the values of a custom field on the same external object. Select the field name, and make sure that the custom field’s values are valid URLs. This field is available in API version 32.0 and later. |
| externalSharingModel       | SharingModel (enumeration of type string)       | Indicates the external org-wide defaults for the object, which determines the access level for external users, such as portal and community users. You can set external org-wide defaults for these objects:  
  - Account  
  - Asset                                                                                                                                                                                                                                                                 |


<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>CustomField[]</td>
<td>Represents one or more fields in the object.</td>
</tr>
<tr>
<td>fieldSets</td>
<td>FieldSet</td>
<td>Defines the field set that exists on this object.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See <code>createMetadata()</code> to see an example of this field specified for a call. This value cannot be <code>null</code>.</td>
</tr>
<tr>
<td>gender</td>
<td>Gender</td>
<td>Indicates the gender of the noun that represents the object. This is used for languages where words need different treatment depending on their gender.</td>
</tr>
<tr>
<td>household</td>
<td>boolean</td>
<td>This field supports relationship groups, a feature available only with Salesforce for Wealth Management. For more information, see “Salesforce for Wealth Management” in Salesforce Help.</td>
</tr>
<tr>
<td>historyRetentionPolicy</td>
<td>HistoryRetentionPolicy</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>indexes</td>
<td>Index[]</td>
<td>Defines the index for a custom big object.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Label that represents the object throughout the Salesforce user interface.</td>
</tr>
<tr>
<td>listViews</td>
<td>ListView[]</td>
<td>Represents one or more list views associated with the object.</td>
</tr>
<tr>
<td>namedFilter</td>
<td>NamedFilter[]</td>
<td>Represents the metadata associated with a lookup filter. This metadata type is used to create, update, or delete lookup filter definitions. This component has been removed as of API version 30.0 and is only available in previous API versions. The metadata associated with a lookup filter is now represented by the lookupFilter field in the CustomField component.</td>
</tr>
</tbody>
</table>

Your org might have other objects whose external org-wide defaults can be modified. This field is available in API version 31.0 and later.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nameField</td>
<td>CustomField</td>
<td>Required for custom objects. On external objects, the name field can instead be specified by setting <code>isNameField</code> to <code>true</code> in the CustomField component. The field that this object’s name is stored in. Every custom object must have a name, usually a string or autonumber. Identifier for the custom object record. This name appears in page layouts, related lists, lookup dialogs, search results, and key lists on tab home pages. By default, this field is added to the custom object page layout as a required field.</td>
</tr>
<tr>
<td>pluralLabel</td>
<td>string</td>
<td>Plural version of the <code>label</code> value.</td>
</tr>
</tbody>
</table>
| publishBehavior             | PlatformEventPublishBehavior      | This field applies only to platform events. Indicates when platform event messages are published in a Lightning Platform transaction. This field applies to event messages published through the Lightning Platform, such as Apex, Process Builder, and Flow Builder, but not through Salesforce APIs. Valid values are:  
  • PublishAfterCommit — The event message is published only after a transaction commits successfully. If the transaction fails, the event message isn't published.  
  • PublishImmediately — The event message is published when the publish call executes, regardless of whether the transaction succeeds.  
If you don’t specify this field, the default value used is PublishImmediately.  
This field is available in API version 46.0 and later. |
| recordTypes                 | RecordType[]                      | An array of one or more record types defined for this object.  
| recordTypeTrackFeedHistory  | boolean                           | Indicates whether the record type is enabled for feed tracking (`true`) or not (`false`). To set this field to `true`, the `enableFeeds` field on the associated `CustomObject` must also be `true`. For more information, see “Customize Chatter Feed Tracking” in Salesforce Help.  
This field is available in API version 19.0 and later. |
| recordTypeTrackHistory      | boolean                           | Indicates whether history tracking is enabled for this record type (`true`) or not (`false`). To set `recordTypeTrackHistory` to `true`, the |
Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableHistory</td>
<td></td>
<td>enableHistory field on the associated custom object must also be true. This field is available in API version 19.0 and later.</td>
</tr>
<tr>
<td>searchLayouts</td>
<td>SearchLayouts</td>
<td>The Search Layouts related list information for the object.</td>
</tr>
<tr>
<td>sharingModel</td>
<td>SharingModel (enumeration of type string)</td>
<td>Indicates the org-wide defaults for the object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Using API version 29.0 and earlier, this field is read-only and can’t be set using the Metadata API; you must use the Salesforce user interface. Using API version 30.0 and later, you can set this field for internal users using the API and the Salesforce user interface.</td>
</tr>
<tr>
<td>sharingReasons</td>
<td>SharingReason[]</td>
<td>The reasons why the object is being shared.</td>
</tr>
<tr>
<td>sharingRecalculations</td>
<td>SharingRecalculation[]</td>
<td>A list of custom sharing recalculations associated with the object.</td>
</tr>
<tr>
<td>startsWith</td>
<td>StartsWith (enumeration of type string)</td>
<td>Indicates whether the noun starts with a vowel, consonant, or is a special character. This is used for languages where words need different treatment depending on the first character. Valid values are listed in StartsWith.</td>
</tr>
<tr>
<td>validationRules</td>
<td>ValidationRule[]</td>
<td>An array of one or more validation rules on the object.</td>
</tr>
<tr>
<td>visibility</td>
<td>SetupObjectVisibility (enumeration of type string)</td>
<td>This field returns the visibility of the custom object, custom setting, or custom metadata type. The following values are valid.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Public—If the custom object, custom setting, or custom metadata type is packaged, it’s accessible to all subscribing orgs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Protected—If the custom object, custom setting, or custom metadata type is in a managed package, it’s accessible only to the developer org. Subscribing orgs can’t access it.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is Public.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 34.0 and later. For custom settings, this field replaces the customSettingsVisibility field.</td>
</tr>
<tr>
<td>webLinks</td>
<td>WebLink[]</td>
<td>An array of one or more weblinks defined for the object.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Additional Components**

CustomObject definitions can include additional components defined in the custom object for declarative metadata. The following components are defined in the CustomObject:
Declarative Metadata Sample Definition

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <deploymentStatus>Deployed</deploymentStatus>
  <description>test object with one field for eclipse ide testing</description>
  <fields>
    <fullName>Comments__c</fullName>
    <description>add your comments about this object here</description>
    <inlineHelpText>This field contains comments made about this object</inlineHelpText>
    <label>Comments</label>
    <length>32000</length>
    <type>LongTextArea</type>
    <visibleLines>30</visibleLines>
  </fields>
  <label>MyFirstObject</label>
  <nameField>
    <label>MyFirstObject Name</label>
    <type>Text</type>
  </nameField>
  <pluralLabel>MyFirstObjects</pluralLabel>
  <sharingModel>ReadWrite</sharingModel>
</CustomObject>
```

The following is the metadata definition of an external object for Salesforce Connect.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <actionOverrides>
    <actionName>CancelEdit</actionName>
    <type>Default</type>
  </actionOverrides>
  <actionOverrides>
    <actionName>Delete</actionName>
    <type>Default</type>
  </actionOverrides>
</CustomObject>
```
<actionOverrides>
  <actionName>Edit</actionName>
  <type>Default</type>
</actionOverrides>

<actionOverrides>
  <actionName>Follow</actionName>
  <type>Default</type>
</actionOverrides>

<actionOverrides>
  <actionName>List</actionName>
  <type>Default</type>
</actionOverrides>

<actionOverrides>
  <actionName>New</actionName>
  <type>Default</type>
</actionOverrides>

<actionOverrides>
  <actionName>SaveEdit</actionName>
  <type>Default</type>
</actionOverrides>

<actionOverrides>
  <actionName>Tab</actionName>
  <type>Default</type>
</actionOverrides>

<actionOverrides>
  <actionName>View</actionName>
  <type>Default</type>
</actionOverrides>

<deploymentStatus>InDevelopment</deploymentStatus>
<description>Products</description>
<enableFeeds>false</enableFeeds>
<externalDataSource>OData</externalDataSource>
<externalIndexAvailable>false</externalIndexAvailable>
<externalName>Products</externalName>

<fields>
  <fullName>DiscontinuedDate__c</fullName>
  <description>DiscontinuedDate</description>
  <externalDeveloperName>DiscontinuedDate</externalDeveloperName>
  <isFilteringDisabled>false</isFilteringDisabled>
  <isRequired>false</isRequired>
  <isNameField>false</isNameField>
  <isSortingDisabled>false</isSortingDisabled>
  <label>DiscontinuedDate</label>
  <type>DateTime</type>
</fields>

<fields>
  <fullName>ID__c</fullName>
  <description>ID</description>
  <externalDeveloperName>ID</externalDeveloperName>
  <isFilteringDisabled>false</isFilteringDisabled>
  <isRequired>false</isRequired>
  <isNameField>false</isNameField>
</fields>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>External Developer Name</th>
<th>Is Filtering Disabled</th>
<th>Is Name Field</th>
<th>Is Sorting Disabled</th>
<th>Label</th>
<th>Length</th>
<th>Required</th>
<th>Precision</th>
<th>Scale</th>
<th>Type</th>
<th>Unique</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td></td>
<td></td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>ID</td>
<td>10</td>
<td>false</td>
<td></td>
<td></td>
<td>Number</td>
<td>false</td>
</tr>
<tr>
<td>Name</td>
<td>Name</td>
<td>Name</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>Name</td>
<td>128</td>
<td>false</td>
<td></td>
<td></td>
<td>Text</td>
<td>false</td>
</tr>
<tr>
<td>Price</td>
<td>Price</td>
<td>Price</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>Price</td>
<td>16</td>
<td>false</td>
<td>2</td>
<td></td>
<td>Number</td>
<td>false</td>
</tr>
<tr>
<td>Products</td>
<td>Products</td>
<td>Products</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>Products</td>
<td>20</td>
<td>false</td>
<td></td>
<td></td>
<td>ExternalLookup</td>
<td>false</td>
</tr>
<tr>
<td>Rating</td>
<td>Rating</td>
<td>Rating</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>Rating</td>
<td></td>
<td>false</td>
<td></td>
<td></td>
<td>Text</td>
<td>false</td>
</tr>
</tbody>
</table>
<CustomObject>
  <fields>
    <fullName>ReleaseDate__c</fullName>
    <description>ReleaseDate</description>
    <externalDeveloperName>ReleaseDate</externalDeveloperName>
    <externalId>false</externalId>
    <isFilteringDisabled>false</isFilteringDisabled>
    <isNameField>false</isNameField>
    <isSortingDisabled>false</isSortingDisabled>
    <label>ReleaseDate</label>
    <required>false</required>
    <type>DateTime</type>
    <unique>false</unique>
  </fields>
</CustomObject>
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file for Field Sets and Record Types but not for other components. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- CustomField
- Metadata
- Picklist (Including Dependent Picklist)
- SearchLayouts
- WebLink
- CustomObjectTranslation
- ListView
- CompactLayout

ActionOverride

Represents an action override on a standard or custom object. Use it to create, update, edit, or delete action overrides. You can access ActionOverride only by accessing its encompassing CustomObject.

Declarative Metadata File Suffix and Directory Location

Action overrides are defined as part of a standard or custom object.

Version

Action overrides are available in API version 18.0 and later. Beginning in Summer ‘13, action overrides can be applied to both standard and custom objects. Previously, action overrides only applied to custom objects.

Fields

Unless otherwise noted, all fields are creatable, filterable, and nillable.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>string</td>
<td>Required. The possible values are the same as the actions you can override:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• accept</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• clone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• delete</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• edit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• list</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• new</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tab</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• view</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>comment</td>
<td>string</td>
<td>Any comments you want associated with the override.</td>
</tr>
<tr>
<td>content</td>
<td>string</td>
<td>Set this field if type is set to flexipage, lightningcomponent, scontrol, or visualforce. It refers to the name of the Lightning page, Lightning component, s-control, or Visualforce page to use as the override. To reference installed components, use this format: <code>Component_namespace__Component_name</code>.</td>
</tr>
<tr>
<td>formFactor</td>
<td>FormFactor (enumeration of type string)</td>
<td>The size of the page being overridden. If the type field is set to flexipage, set this field to Large to override the View action with a Lightning page in Lightning Experience. The Large value represents the Lightning Experience desktop environment and is valid only for the flexipage and lightningcomponent types. The Small value represents the Salesforce app on a phone or tablet. The Medium value is reserved for future use. The null value (which is the same as specifying no value) represents Salesforce Classic. This field is available in API version 37.0 and later and is part of the feature for creating and editing record pages in Lightning Experience.</td>
</tr>
<tr>
<td>skipRecordTypeSelect</td>
<td>boolean</td>
<td>Set this field to true if you prefer that any new records created by this action override aren’t forwarded to the record type selection page. This field is only valid if the actionName is a &quot;create&quot; type (like new), and type is set to visualforce. This field is available in API version 21.0 and later.</td>
</tr>
<tr>
<td>type</td>
<td>ActionOverrideType (enumeration of type string)</td>
<td>Required. Represents the type of action override. Valid values are described in ActionOverrideType.</td>
</tr>
</tbody>
</table>

**Note:** Lightning component overrides return different FormFactor values depending on the API version used.

- In API version 41.0 and earlier, Lightning component overrides return only the null value (no value), representing the Salesforce Classic environment.
- In API version 42.0, if you specify different Lightning component overrides for Lightning Experience and mobile, one component is selected randomly for both overrides and its FormFactor value is returned. If there’s a conflict between Lightning components, and a Visualforce page override is also specified for Salesforce Classic, the Visualforce page takes precedence.
- In API version 43.0 and later, a Lightning component override for Lightning Experience returns the Large value and a Lightning component override for mobile returns the Small value, as expected.
**ActionOverrideType**

ActionOverrideType is an enumeration of type string that defines which kind of action override to use. The valid values are:

- **default**—The override uses a custom override provided by an installed package. If there isn’t one available, the standard Salesforce behavior is used.
- **flexipage**—The override uses behavior from a Lightning page, and is only valid for the View action in Lightning Experience.
- **lightningcomponent**—The override uses behavior from a Lightning component.
- **scontrol**—The override uses behavior from an s-control.
- **standard**—The override uses regular Salesforce behavior.
- **visualforce**—The override uses behavior from a Visualforce page.

**Note:** Existing s-controls can be used as overrides for Salesforce Classic under certain conditions. However, s-controls have been deprecated since the Spring ’09 release. We recommend using Visualforce pages instead.

**Declarative Metadata Sample Definitions**

You can define action overrides, as in these examples for the Edit action.

**A Visualforce page override for Salesforce Classic:**

```xml
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <actionOverrides>
    <actionName>edit</actionName>
    <type>visualforce</type>
    <content>myEditVFPage</content>
    <comment>This edit action is a lot safer.</comment>
  </actionOverrides>
</CustomObject>
```

**Note:** This example includes no value for FormFactor. Using no value is the same as using the null value, which represents Salesforce Classic.

**A Lightning component override for Lightning Experience:**

```xml
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <actionOverrides>
    <actionName>edit</actionName>
    <type>lightningcomponent</type>
    <content>myEditLightningComponent</content>
    <formFactor>Large</formFactor>
    <comment>This edit action is a lot safer.</comment>
  </actionOverrides>
</CustomObject>
```

**A Lightning component override for the Salesforce app:**

```xml
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <actionOverrides>
    <actionName>edit</actionName>
    <type>lightningcomponent</type>
    <content>myEditLightningComponent</content>
    <formFactor>Small</formFactor>
    <comment>This edit action is a lot safer.</comment>
  </actionOverrides>
</CustomObject>
```
When overrides are included in a managed package, the overrides are represented as `default` type in the metadata. Calling `retrieve()` presents the following:

```xml
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <actionOverrides>
    <actionName>edit</actionName>
    <type>default</type>
  </actionOverrides>
</CustomObject>
```

If you subscribe to a managed package with default overrides, you can replace the default override behavior by editing the XML. For example, to replace the Visualforce page override with the Salesforce standard page for Salesforce Classic, use:

```xml
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <actionOverrides>
    <actionName>edit</actionName>
    <type>standard</type>
  </actionOverrides>
</CustomObject>
```

To set a Lightning page action override on the View standard button in Lightning Experience, use:

```xml
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <actionOverrides>
    <actionName>View</actionName>
    <content>myLightningPage</content>
    <formFactor>Large</formFactor>
    <type>flexipage</type>
  </actionOverrides>
</CustomObject>
```

**Wildcard Support in the Manifest File**

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:

  - CustomObject

**BusinessProcess**

The `BusinessProcess` metadata type enables you to display different picklist values for users based on their profile. This type extends the `Metadata` metadata type and inherits its `fullName` field.

Multiple business processes allow you to track separate sales, support, and lead lifecycles. A sales, support, lead, or solution process is assigned to a record type. The record type determines the user profiles that are associated with the business process. For more information, see "Managing Multiple Business Processes" in the Salesforce online help.
Declarative Metadata File Suffix and Directory Location

Business processes are defined as part of the custom object or standard object definition. See CustomObject for more information.

Version

BusinessProcess components are available in API version 17.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Description for the business process.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The name used as a unique identifier for API access. This field is inherited from the Metadata component, but the string it contains is created differently than the fullName strings for other types. For a fullName string BusinessProcess, the fullName is created combining the Entity Name and Business Process Name. For example, for a business process called “Bulk Orders” for opportunities, the fullName would be Opportunity.Bulk Orders.</td>
</tr>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Indicates if the business process is active (true) or not (false).</td>
</tr>
<tr>
<td>namespacePrefix</td>
<td>string</td>
<td>The namespace of the developer organization where the package was created.</td>
</tr>
<tr>
<td>values</td>
<td>PicklistValue[]</td>
<td>A list of picklist values associated with this business process.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

A sample XML definition of a lead business process is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
    ....
    <businessProcesses>
        <fullName>HardwareLeadProcess</fullName>
        <description>Lead Process for hardware division</description>
        <isActive>true</isActive>
        <values>
            <fullName>Closed - Converted</fullName>
            <default>false</default>
        </values>
        <values>
            <fullName>CustomLeadStep1</fullName>
            <default>false</default>
        </values>
        <values>
            <fullName>CustomLeadStep2</fullName>
        </values>
    </businessProcesses>
</CustomObject>
```
**Metadata Types**

```xml
<fullName>Open - Not Contacted</fullName>
<default>false</default>
</values>
<values>
  <fullName>Working - Contacted</fullName>
  <default>true</default>
</values>
</businessProcesses>
```

---

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file only when a `RecordType` on page 338 is specified. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**SEE ALSO:**
- CustomObject
- CompactLayout

**CompactLayout**

Represents the metadata associated with a compact layout. This type extends the `Metadata` metadata type and inherits its `fullName` field.

A compact layout displays a record’s key fields at a glance in both the Salesforce mobile app and Lightning Experience.

Compact layouts support all field types except:

- text area
- long text area
- rich text area
- multi-select picklist

For more information on compact layouts, see “Compact Layouts” in the Salesforce Help.

**File Suffix and Directory Location**

Compact layouts are defined as part of the custom object, standard object, or external object definition. See `CustomObject` for more information.

**Version**

CompactLayout components are available in API version 29.0 and later. CompactLayout components are available for external objects in API version 42.0 and later.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>string</td>
<td>The fields assigned to the compact layout. Their order represents the prioritization given to them when defining the compact layout.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Label that represents the object throughout the Salesforce user interface.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a CompactLayout component:

```xml
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
   <actionOverrides>
      <actionName>Accept</actionName>
      <type>Default</type>
   </actionOverrides>
   <actionName>Clone</actionName>
   <type>Default</type>
   <actionName>Delete</actionName>
   <type>Default</type>
   <actionName>Edit</actionName>
   <type>Default</type>
   <actionName>List</actionName>
   <type>Default</type>
   <actionName>New</actionName>
   <type>Default</type>
   <actionName>Tab</actionName>
   <type>Default</type>
   <actionName>View</actionName>
   <type>Default</type>
   <compactLayouts>
      <fullName>testCompactLayout</fullName>
      <compactLayoutItems>textfield__c</compactLayoutItems>
      <label>testCompactLayoutLabel</label>
   </compactLayouts>
   <compactLayoutAssignment>SYSTEM</compactLayoutAssignment>
   <deploymentStatus>Deployed</deploymentStatus>
</CustomObject>
```
**Metadata Types**

**CustomField**

```xml
<fields>
    <fullName>textfield__c</fullName>
    <externalId>false</externalId>
    <label>textfield</label>
    <length>255</length>
    <required>false</required>
    <type>Text</type>
    <unique>false</unique>
</fields>

<nameField>
    <label>customObj Name</label>
    <type>Text</type>
</nameField>

<pluralLabel>customObjs</pluralLabel>

<recordTypes>
    <fullName>RT1</fullName>
    <active>true</active>
    <label>RT1</label>
    <compactLayoutAssignment>testCompactLayout</compactLayoutAssignment>
</recordTypes>

<recordTypes>
    <fullName>RT2</fullName>
    <active>true</active>
    <label>RT2</label>
</recordTypes>

<searchLayouts/>

<sharingModel>ReadWrite</sharingModel>
</CustomObject>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**CustomField**

Represents the metadata associated with a field. Use this metadata type to create, update, or delete custom field definitions on standard, custom, and external objects or standard field definitions on standard objects. This type extends the Metadata metadata type and inherits its fullName field.

Only standard fields that you can customize are supported, that is, standard fields to which you can add help text or enable history tracking or Chatter feed tracking. Other standard fields aren’t supported, including system fields (such as CreatedById or LastModifiedDate) and autonumber fields. Some standard picklist fields aren’t supported. See Unsupported Metadata Types.

Specify the full name whenever you create or update a field. For example, a custom field on a custom object:

```
MyCustomObject__c.MyCustomField__c
```
An example of a custom field on a standard object:

Account.MyAcctCustomField__c

An example of a standard field on a standard object:

Account.Phone

An example of a custom field on an external object:

MyExternalObject__x.MyCustomField__c

Note: In Metadata API, external objects are represented by the CustomObject metadata type.

The following custom field types aren’t available for external objects.

- Auto-number (available only with the cross-org adapter for Salesforce Connect)
- Currency (available only with the cross-org adapter for Salesforce Connect)
- Formula
- Location
- Master-detail relationship
- Picklist and multi-select picklist (available only with the cross-org adapter for Salesforce Connect)
- Roll-up summary
- Text (encrypted)
- Text Area (rich)

Declarative Metadata File Suffix and Directory Location

Custom fields are user-defined fields and are part of the custom object or standard object definition. See CustomObject for more information. Standard fields are predefined on standard objects.

Note: Retrieving a component of this metadata type in a project makes the component appear in any Profile and PermissionSet components that are retrieved in the same package.

Retrieving Fields on Custom or Standard Objects

When you retrieve a custom or standard object, you return everything associated with the object, except for standard fields that aren’t customizable. You can also retrieve only specific fields for an object by explicitly naming the object and fields in package.xml. The following definition in package.xml will create the files objects/MyCustomObject__c.object and objects/Account.object, each containing the requested field definitions.

```xml
<types>
  <members>MyCustomObject__c.MyCustomField__c</members>
  <members>Account.MyCustomAccountField__c</members>
  <members>Account.Phone</members>
  <name>CustomField</name>
</types>
```

Version

Custom and standard fields are available in API version 10.0 and later.
## Fields

Unless otherwise noted, all fields are createable, filterable, and nillable.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>businessOwnerGroup</td>
<td>reference</td>
<td>Indicates the group associated with this field. The business owner group understands the importance of the field’s data to your company, and might be responsible for determining the minimum security classification. This field is available in API version 45.0 and later.</td>
</tr>
<tr>
<td>businessOwnerUser</td>
<td>reference</td>
<td>Indicates the person associated with this field. The business owner understands the importance of the field’s data to your company, and might be responsible for determining the minimum security classification. This field is available in API version 45.0 and later.</td>
</tr>
<tr>
<td>businessStatus</td>
<td>picklist</td>
<td>Indicates whether the field is in use. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Active</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DeprecateCandidate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Hidden</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 45.0 and later.</td>
</tr>
<tr>
<td>caseSensitive</td>
<td>boolean</td>
<td>Indicates whether the field is case-sensitive (true) or not (false).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For indirect lookup relationship fields on external objects, this attribute affects how this custom field’s values are matched against the values of the referenceTargetField.</td>
</tr>
<tr>
<td>customDataType</td>
<td>string</td>
<td>Deprecated in the Spring ’19 (API version 45.0) release.</td>
</tr>
<tr>
<td>defaultValue</td>
<td>string</td>
<td>If specified, represents the default value of the field.</td>
</tr>
<tr>
<td>deleteConstraint</td>
<td>DeleteConstraint (enumeration of type string)</td>
<td>Provides deletion options for lookup relationships. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SetNull</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This is the default. If the lookup record is deleted, the lookup field is cleared.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Restrict</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Prevents the record from being deleted if it’s in a lookup relationship.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cascade</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Deletes the lookup record as well as associated lookup fields.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For more information on lookup relationships, see “Object Relationships” in the Salesforce Help.</td>
</tr>
<tr>
<td>deprecated</td>
<td>boolean</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the field.</td>
</tr>
<tr>
<td>displayFormat</td>
<td>string</td>
<td>The display format.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>displayLocationInDecimal</td>
<td>boolean</td>
<td>Indicates how the geolocation values of a Location custom field appears in the user interface. If true, the geolocation values appear in decimal notation. If false, the geolocation values appear as degrees, minutes, and seconds.</td>
</tr>
<tr>
<td>encrypted</td>
<td>boolean</td>
<td><strong>Note:</strong> This entry is about Shield Platform Encryption, not Classic Encryption.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Indicates whether this field is encrypted (true) or not (false). This field is available in API version 34.0 through 43.0.</td>
</tr>
<tr>
<td>encryptionScheme</td>
<td>EncryptionScheme (enumeration of type string)</td>
<td><strong>Note:</strong> This entry is about Shield Platform Encryption, not Classic Encryption.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For encrypted fields, determines which encryption scheme a field takes. Valid values are None, ProbabilisticEncryption, CaseSensitiveDeterministicEncryption, and CaseInsensitiveDeterministicEncryption. This field is available in API version 44.0 and later.</td>
</tr>
<tr>
<td>externalDeveloperName</td>
<td>string</td>
<td>Available only for external objects. Name of the table column on the external data source that maps to this custom field in Salesforce. Corresponds to External Column Name in the user interface. This field is available in API version 32.0 and later.</td>
</tr>
<tr>
<td>externalId</td>
<td>boolean</td>
<td>Indicates whether the field is an external ID field (true) or not (false).</td>
</tr>
<tr>
<td>fieldManageability</td>
<td>FieldManageability (enumeration of type string)</td>
<td>Determines who can update the field after it’s released in a managed package. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Locked—The field can’t be updated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DeveloperControlled—The creator of the record can update the field with a package upgrade.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SubscriberControlled—Anyone with proper permissions can update the field. The field can’t be updated with a package upgrade.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Available only for fields on custom metadata types. If the field type is MetadataRelationship, and the manageability of the entity definition field is:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Subscriber-controlled, then the Field Definition field must be subscriber-controlled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Upgradeable, then the Field Definition field must be either upgradeable or subscriber-controlled.</td>
</tr>
<tr>
<td>formula</td>
<td>string</td>
<td>If specified, represents a formula on the field.</td>
</tr>
<tr>
<td>formulaTreatBlankAs</td>
<td>TreatBlanksAs (enumeration of type string)</td>
<td>Indicates how to treat blanks in a formula. Valid values are BlankAsBlank and BlankAsZero.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See <code>createMetadata()</code> to see an example of this field specified for a call. This value cannot be null.</td>
</tr>
<tr>
<td>globalPicklist</td>
<td>string</td>
<td>(This field is available in API version 37.0 only and removed from later versions.) If this custom field is a picklist that's based on a global picklist, <code>globalPicklist</code> is the name of the global picklist whose value set this picklist inherits. A custom picklist that's based on a global picklist is restricted. You can only add or remove values by editing the global picklist.</td>
</tr>
<tr>
<td>indexed</td>
<td>boolean</td>
<td>Indicates if the field is indexed. If this field is unique or the <code>externalId</code> is set true, the <code>isIndexed</code> value is set to true. This field has been deprecated as of version 14.0 and is only provided for backward compatibility.</td>
</tr>
<tr>
<td>inlineHelpText</td>
<td>string</td>
<td>Represents the content of field-level help. For more information, see &quot;Define Field-Level Help&quot; in the Salesforce Help.</td>
</tr>
<tr>
<td>isAIPredictionField</td>
<td>boolean</td>
<td>Available for Number type custom fields when you use Einstein Prediction Builder. Denotes whether the field can store and display Einstein prediction data on an object. Use Einstein Prediction Builder to determine the data for the target field. This field is available in API version 43.0 and later.</td>
</tr>
<tr>
<td>isFilteringDisabled</td>
<td>boolean</td>
<td>Available only for external objects. Indicates whether the custom field is available in filters. This field is available in API version 32.0 and later.</td>
</tr>
<tr>
<td>isNameField</td>
<td>boolean</td>
<td>Available only for external object fields of type text. For each external object, you can specify one field as the name field. If you set this to <code>true</code>, make sure that the external table column identified by the <code>externalDeveloperName</code> attribute contains name values. This field is available in API version 32.0 and later.</td>
</tr>
<tr>
<td>isSortingDisabled</td>
<td>boolean</td>
<td>Available only for external objects. Indicates whether the custom field is sortable. This field is available in API version 32.0 and later.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Label for the field. You cannot update the label for standard picklist fields, such as the <code>Industry</code> field for accounts.</td>
</tr>
<tr>
<td>length</td>
<td>int</td>
<td>Length of the field.</td>
</tr>
<tr>
<td>lookupFilter</td>
<td><code>LookupFilter</code></td>
<td>Represents the metadata associated with a lookup filter. This metadata type is used to create, update, or delete lookup filter definitions. This component has been removed as of API version 30.0 and is only available in previous API versions. The metadata</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
lookupFilter | DescriptionField | associated with a lookup filter is now represented by the lookupFilter field in the CustomField component. This field is available in API version 30.0 and later. 
- **Note:** LookupFilter is not supported on the article type object.

maskChar | EncryptedFieldMaskChar | For encrypted fields, specifies the character to be used as a mask. Valid values are enumerated in EncryptedFieldMaskChar. For more information on encrypted fields, see “Classic Encryption for Custom Fields” in the Salesforce Help.

maskType | EncryptedFieldMaskType | For encrypted text fields, specifies the format of the masked and unmasked characters in the field. Valid values are enumerated in EncryptedFieldMaskType. For more information on encrypted fields, see “Classic Encryption for Custom Fields” in the Salesforce Help.

metadataRelationship | string | In custom metadata relationships, represents the controlling field that specifies the standard or custom object in an entity definition metadata relationship. Required when creating a field definition metadata relationship on a custom metadata type. The object specified in the controlling field determines the values available in its dependent field definition. For example, specifying the Account object filters the available fields in the field definition to Account fields only. This field is available in API version 39.0 and later.

ControllingField | | (Deprecated. Use this field in API version 37.0 and earlier only. In later versions, use valueSet instead.) If specified, the field is a picklist, and this field enumerates the picklist values and labels.

picklist | Picklist | Indicates whether existing rows will be populated (`true`) or not (`false`).

populateExistingRows | boolean | The precision for number values. Precision is the number of digits in a number. For example, the number 256.99 has a precision of 5.

referenceTargetField | string | Available only for indirect lookup relationship fields on external objects. Specifies the custom field on the parent object to match against this indirect lookup relationship field, whose values come from an external data source. The specified custom field on the
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>parent object must have both <code>externalId</code> and <code>unique</code> set to <code>true</code>. This field is available in API version 32.0 and later.</td>
</tr>
<tr>
<td><code>referenceTo</code></td>
<td>string</td>
<td>If specified, indicates a reference this field has to another object.</td>
</tr>
<tr>
<td><code>relationshipLabel</code></td>
<td>string</td>
<td>Label for the relationship.</td>
</tr>
<tr>
<td><code>relationshipName</code></td>
<td>string</td>
<td>If specified, indicates the value for one-to-many relationships. For example, in the object <code>MyObject</code> that had a relationship to <code>YourObject</code>, the relationship name might be <code>YourObjects</code>.</td>
</tr>
<tr>
<td><code>relationshipOrder</code></td>
<td>int</td>
<td>This field is valid for all master-detail relationships, but the value is only non-zero for junction objects. A junction object has two master-detail relationships, and is analogous to an association table in a many-to-many relationship. Junction objects must define one parent object as primary (0), the other as secondary (1). The definition of primary or secondary affects delete behavior and inheritance of look and feel, and record ownership for junction objects. For more information, see the Salesforce Help. 0 or 1 are the only valid values, and 0 is always the value for objects that are not junction objects.</td>
</tr>
<tr>
<td><code>reparentableMasterDetail</code></td>
<td>boolean</td>
<td>Indicates whether the child records in a master-detail relationship on a custom object can be reparented to different parent records. The default value is <code>false</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 25.0 and later.</td>
</tr>
<tr>
<td><code>required</code></td>
<td>boolean</td>
<td>Indicates whether the field requires a value on creation (<code>true</code>) or not (<code>false</code>).</td>
</tr>
<tr>
<td><code>scale</code></td>
<td>int</td>
<td>The scale for the field. Scale is the number of digits to the right of the decimal point in a number. For example, the number 256.99 has a scale of 2.</td>
</tr>
<tr>
<td><code>securityClassification</code></td>
<td>picklist</td>
<td>Indicates the sensitivity of the data contained in the field. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Public</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Internal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Confidential</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Restricted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MissionCritical</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 45.0 and later.</td>
</tr>
<tr>
<td><code>startingNumber</code></td>
<td>int</td>
<td>If specified, indicates the starting number for the field. When you create records, <code>Starting Number</code>'s value increments to store the number that will be assigned to the next auto-number field created.</td>
</tr>
</tbody>
</table>
### Field Type: Description

**Note:**
- You can’t retrieve the starting number of an auto-number field through Metadata API. To specify a **Starting Number** while deploying, add a `startingNumber` tag for your field to your package.xml file. For example: `<startingNumber>42</startingNumber>`
- If you deploy without specifying a **Starting Number** value in your package.xml file, the default starting number for standard fields is 0. The default starting number for custom fields is 1.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stripMarkup</td>
<td>boolean</td>
<td>Set to <code>true</code> to remove markup, or <code>false</code> to preserve markup. Used when converting a rich text area to a long text area.</td>
</tr>
<tr>
<td>summarizedField</td>
<td>string</td>
<td>Represents the field on the detail row that is being summarized. This field cannot be null unless the <code>summaryOperation</code> value is <code>count</code>.</td>
</tr>
<tr>
<td>summaryFilterItems</td>
<td>FilterItem[]</td>
<td>Represents the set of filter conditions for this field if it is a summary field. This field will be summed on the child if the filter conditions are met.</td>
</tr>
<tr>
<td>summaryForeignKey</td>
<td>string</td>
<td>Represents the master-detail field on the child that defines the relationship between the parent and the child.</td>
</tr>
<tr>
<td>summaryOperation</td>
<td>SummaryOperations (enumeration of type string)</td>
<td>Represents the sum operation to be performed. Valid values are enumerated in <code>SummaryOperations</code>.</td>
</tr>
<tr>
<td>trackFeedHistory</td>
<td>boolean</td>
<td>Indicates whether the field is enabled for feed tracking (<code>true</code>) or not (<code>false</code>). To set this field to <code>true</code>, the <code>enableFeeds</code> field on the associated <code>CustomObject</code> must also be <code>true</code>. For more information, see “Customize Chatter Feed Tracking” in the Salesforce Help. This field is available in API version 18.0 and later.</td>
</tr>
<tr>
<td>trackHistory</td>
<td>boolean</td>
<td>Indicates whether history tracking is enabled for the field (<code>true</code>) or not (<code>false</code>). Also available for standard object fields (picklist and lookup fields only) in API version 30.0 and later. To set <code>trackHistory</code> to <code>true</code>, the <code>enableHistory</code> field on the associated standard or custom object must also be <code>true</code>. For more information, see “Field History Tracking” in the Salesforce Help. Field history tracking isn’t available for external objects.</td>
</tr>
<tr>
<td>trackTrending</td>
<td>boolean</td>
<td>Indicates whether historical trending data is captured for the field (<code>true</code>) or not (<code>false</code>). An object is enabled for historical trending</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>trueValueIndexed</td>
<td>boolean</td>
<td>This is only relevant for a checkbox field. If set, true values are built into the index. This field has been deprecated as of API version 14.0 and is only provided for backward compatibility.</td>
</tr>
<tr>
<td>type</td>
<td>FieldType</td>
<td>Indicates the field type for the field. Valid values are enumerated in FieldType. For standard fields on standard objects, the type field is optional. This field is included for some standard field types, such as Picklist or Lookup, but not for others. The type field is included for custom fields.</td>
</tr>
<tr>
<td>unique</td>
<td>boolean</td>
<td>Indicates whether the field is unique (true) or not (false).</td>
</tr>
<tr>
<td>valueSet</td>
<td>ValueSet</td>
<td>Represents the set of values that make up a picklist on a custom field. Each value is defined as a CustomValue on page 379. If this custom field is a picklist that uses a global value set, valueSet is the name of the global value set whose values this picklist inherits. A custom picklist that uses a global value set is restricted. You can only add or remove values by editing the global value set. Note: A ValueSet component has either a valueSetDefinition or a valueName specified, but never both. This field is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>visibleLines</td>
<td>int</td>
<td>Indicates the number of lines displayed for the field.</td>
</tr>
<tr>
<td>writeRequiresMasterRead</td>
<td>boolean</td>
<td>Sets the minimum sharing access level required on the master record to create, edit, or delete child records. This field applies only to master-detail or junction object custom field types.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- true — Allows users with “Read” access to the master record permission to create, edit, or delete child records. This setting makes sharing less restrictive.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- false — Allows users with “Read/Write” access to the master record permission to create, edit, or delete child records. This setting is more restrictive than true, and is the default value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For junction objects, the most restrictive access from the two parents is enforced. For example, if you set to true on both master-detail fields, but users have “Read” access to one master record and “Read/Write” access to the other master record, users won’t be able to create, edit, or delete child records.</td>
</tr>
</tbody>
</table>
Fields use additional data types. For more information, see Metadata Field Types on page 351.

**EncryptedFieldMaskChar**

This field type is used in `maskChar`. It is a string with two valid values: asterisk or X. For more information on encrypted fields, see Classic Encryption for Custom Fields in the Salesforce online help.

**EncryptedFieldMaskType**

This field type is used in `maskType`. Valid values are:

- **all**
  All characters in the field are hidden. This option is equivalent to the Mask All Characters option in Salesforce.

- **creditCard**
  The first 12 characters are hidden and the last four display. This option is equivalent to the Credit Card Number option in Salesforce.

- **ssn**
  The first five characters are hidden and the last four display. This option is equivalent to the Social Security Number option in Salesforce.

- **lastFour**
  All characters are hidden but the last four display. This option is equivalent to the Last Four Characters Clear option in Salesforce.

- **sin**
  All characters are hidden but the last four display. This option is equivalent to the Social Insurance Number option in Salesforce.

- **nino**
  All characters are hidden. Salesforce automatically inserts spaces after each pair of characters if the field contains nine characters. This option is equivalent to the National Insurance Number option in Salesforce.

For more information on encrypted fields, see Classic Encryption for Custom Fields in the Salesforce online help.

**LookupFilter**

Represents the metadata associated with a lookup filter. Replaces the NamedFilter component, which was removed as of API version 30.0. LookupFilter is available in API version 30.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Indicates whether or not the lookup filter is active.</td>
</tr>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Specifies advanced filter conditions.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of what this filter does.</td>
</tr>
<tr>
<td>errorMessage</td>
<td>string</td>
<td>The error message that appears if the lookup filter fails.</td>
</tr>
<tr>
<td>filterItems</td>
<td>FilterItem[]</td>
<td>Required. The set of filter conditions. You can have up to 10 FilterItems per lookup filter.</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>infoMessage</td>
<td>string</td>
<td>The information message displayed on the page. Use to describe things the user might not understand, such as why certain items are excluded in the lookup filter.</td>
</tr>
<tr>
<td>isOptional</td>
<td>boolean</td>
<td>Required. Indicates whether or not the lookup filter is optional.</td>
</tr>
</tbody>
</table>

Lookup filters use additional data types. For more information, see [Metadata Field Types](#).

## FilterItem

Represents one entry in a set of filter criteria.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>Represents the field specified in the filter.</td>
</tr>
<tr>
<td>operation</td>
<td>FilterOperation (enumeration of type string)</td>
<td>Represents the filter operation for this filter item. Valid values are enumerated in FilterOperation.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Represents the value of the filter item being operated upon, for example, if the filter is <code>my_number_field__c &gt; 1</code>, the value of <code>value</code> is 1.</td>
</tr>
<tr>
<td>valueField</td>
<td>string</td>
<td>Specifies if the final column in the filter contains a field or a field value. Approval processes don’t support <code>valueField</code> entries in filter criteria.</td>
</tr>
</tbody>
</table>

### FilterOperation

This is an enumeration of type string that lists different filter operations. Valid values are:

- equals
- notEqual
- lessThan
- greaterThan
- lessOrEqual
- greaterOrEqual
- contains
- notContain
- startsWith
- includes
- excludes
- within (DISTANCE criteria only)
**SummaryOperations**

Represents the type of a `summaryOperation`. Valid values are:

- Count
- Min
- Max
- Sum

**Declarative Metadata Sample Definition**

The following example shows a field definition for a custom field that is named `Comments__c`.

```xml
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
    ....
    <fields>
        <fullName>Comments__c</fullName>
        <description>Add your comments about this object here</description>
        <inlineHelpText>This field contains help text for this object</inlineHelpText>
        <label>Comments</label>
        <length>32000</length>
        <type>LongTextArea</type>
        <visibleLines>30</visibleLines>
    </fields>
    ....
</CustomObject>
```

The following is the definition for two fields on the Account standard object—a custom field (`MyCustomAccountField__c`), and a standard field (`Phone`) that has history tracking enabled.

```xml
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
    <fields>
        <fullName>MyCustomAccountField__c</fullName>
        <description>A custom field on the Account standard object.</description>
        <externalId>false</externalId>
        <inlineHelpText>Some help text.</inlineHelpText>
        <label>MyCustomAccountField</label>
        <length>100</length>
        <required>false</required>
        <trackFeedHistory>false</trackFeedHistory>
        <trackHistory>false</trackHistory>
        <type>Text</type>
        <unique>false</unique>
    </fields>
    ....
</CustomObject>
```
**Wildcard Support in the Manifest File**

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- CustomObject
- Picklist (Including Dependent Picklist)
- Metadata
- NamedFilter

**FieldSet**

Represents a field set. A field set is a grouping of fields. For example, you could have a field set that contains fields describing a user’s first name, middle name, last name, and business title.

Field sets can be referenced on Visualforce pages dynamically. If the page is added to a managed package, administrators can add, remove, or reorder fields in a field set to modify the fields presented on the Visualforce page without modifying any code.

**Version**

FieldSet components are available in API version 21.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>availableFields</td>
<td>FieldSetItem[]</td>
<td>An array containing all the possible fields in the field set.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Required. A description provided by the developer that describes the field set. This is required.</td>
</tr>
<tr>
<td>displayedFields</td>
<td>FieldSetItem[]</td>
<td>An array containing all the fields that are presented on the Visualforce page. The order in which a field is listed determines the order of appearance on the page.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The label used to reference the field set.</td>
</tr>
</tbody>
</table>

**FieldSetItem**

FieldSetItem represents an individual field in a field set.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>Required. The name of a field in a standard or custom object.</td>
</tr>
<tr>
<td>isFieldManaged</td>
<td>boolean</td>
<td>Read-only. Denotes whether the field was added to the field set via a managed or unmanaged package.</td>
</tr>
</tbody>
</table>
### Declarative Metadata Sample Definition

A sample XML definition of a FieldSet component is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <fieldSets>
    <fullName>FieldSetNames</fullName>
    <availableFields>
      <field>MiddleName__c</field>
    </availableFields>
    <availableFields>
      <field>Title__c</field>
    </availableFields>
    <description>FieldSet containing how to properly address someone</description>
    <displayedFields>
      <field>FirstName__c</field>
    </displayedFields>
    <displayedFields>
      <field>LastName__c</field>
    </displayedFields>
    <label>FieldSet Names</label>
  </fieldSets>
</CustomObject>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### HistoryRetentionPolicy

Represents the policy for retaining field history data. By setting a policy, you can specify the number of months you want to maintain field history in Salesforce and the number of years that you want to retain field history in the archive.

This component is only available to users with the “RetainFieldHistory” permission.

### Declarative Metadata File Suffix and Directory Location

Field history retention policies are defined as part of a standard or custom object. You can set field history retention policies for objects individually. See CustomObject for more information.

### Version

Available in API version 31.0 and later.
### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>archiveAfterMonths</td>
<td>int</td>
<td>Required. The number of months that you want to keep field history data in Salesforce before archiving. You can set a minimum of 1 month and a maximum of 18 months. If you don’t set a number, the default is 18 months. (That is, Salesforce maintains data for 18 months before archiving.)</td>
</tr>
<tr>
<td>archiveRetentionYears</td>
<td>int</td>
<td>Required. The number of years that you want to retain data in the archive. You can set a minimum of zero years, and a maximum of 10 years. If no number is set, the default is 10 years.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A text description for the history retention.</td>
</tr>
<tr>
<td>gracePeriodDays</td>
<td>int</td>
<td>The number of days of extra time after the <code>archiveAfterMonths</code> period before the data is archived. The <code>gracePeriodDays</code> interval applies only to the first time that the data is archived; because all the data is copied the first time, the operation may take longer than subsequent times when only the data that changed since the last archival operation is copied. The <code>gracePeriodDays</code> provides extra time for the administrator to prepare the organization before the initial archive operation. You can set a minimum of zero days and a maximum of 10 days. If no number is set, the default is 1 day.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

This sample shows the definition of a history retention policy for a custom object:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <historyRetentionPolicy>
    <archiveAfterMonths>6</archiveAfterMonths>
    <archiveRetentionYears>5</archiveRetentionYears>
    <description>My field history retention</description>
  </historyRetentionPolicy>
</CustomObject>
```

### Index

Represents an index defined within a custom big object. Use this metadata type to define the composite primary key (index) for a custom big object. This type extends the Metadata metadata type and inherits its `fullName` field.

### File Suffix and Directory Location

Indexes are user-defined and are part of the custom object definition for big objects. See CustomObject for more information.
Version

The Index type is available in API version 41.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>IndexField[]</td>
<td>The definition of the fields in the index.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. This name is used to refer to the big object in the user interface. Available in API version 41.0 and later.</td>
</tr>
</tbody>
</table>

IndexField

Defines which fields make up the index, their order, and sort direction. The order in which the fields are defined determines the order fields are listed in the index.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The API name for the field that's part of the index. This value must match the fullName value for the corresponding field in the fields section and be marked as required.</td>
</tr>
</tbody>
</table>

⚠️ Warning: When querying a big object record via SOQL and passing the results as arguments to the delete API, if any index field name has a leading or trailing white space, you can't delete the big object record.

| sortDirection | string | Required. The sort direction of the field in the index. Valid values are ASC for ascending order and DESC for descending order. |

Declarative Metadata Sample Definition

The following is an example of an index contained within the definition of a custom big object, Customer_Interactions__b.object.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <deploymentStatus>Deployed</deploymentStatus>

  // Define the fields within the big object
  <fields>
    <fullName>Purchase__c</fullName>
    <label>Purchase</label>
    <length>16</length>
    <required>false</required>
    <type>Text</type>
    <unique>false</unique>
  </fields>
</CustomObject>
```
<fields>
  <fullName>Order_Number__c</fullName>
  <label>Order Number</label>
  <length>16</length>
  <required>false</required>
  <type>Text</type>
  <unique>true</unique>
</fields>

<fields>
  <fullName>Platform__c</fullName>
  <label>Platform</label>
  <length>16</length>
  <required>true</required>
  <type>Text</type>
  <unique>false</unique>
</fields>

<fields>
  <fullName>Account__c</fullName>
  <label>User Account</label>
  <referenceTo>Account</referenceTo>
  <relationshipName>User_Account</relationshipName>
  <required>true</required>
  <type>Lookup</type>
</fields>

<fields>
  <fullName>Order_Date__c</fullName>
  <label>Order Date</label>
  <required>true</required>
  <type>DateTime</type>
</fields>

// Define the index
<indexes>
  <fullName>CustomerInteractionsIndex</fullName>
  <label>Customer Interactions Index</label>
  <fields>
    <name>Account__c</name>
    <sortDirection>DESC</sortDirection>
  </fields>
  <fields>
    <name>Platform__c</name>
    <sortDirection>ASC</sortDirection>
  </fields>
  <fields>
    <name>Order_Date__c</name>
    <sortDirection>DESC</sortDirection>
  </fields>
</indexes>

<label>Customer Interaction</label>
Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- CustomObject
- Metadata

ListView

ListView allows you to see a filtered list of records, such as contacts, accounts, or custom objects.

This type extends the Metadata metadata type and inherits its fullName field. See “Create a Custom List View in Salesforce Classic” in the Salesforce online help.

Note: List views with the Visible only to me Restrict Visibility option are not accessible in Metadata API. Each of these list views is associated with a particular user.

Declarative Metadata File Suffix and Directory Location

List views are stored within a CustomObject component. The component can represent a custom object or a standard object, such as an account.

Version

ListView components for custom objects are available in API version 14.0 and later. ListView components for standard objects, such as accounts, are available in API version 17.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>This field represents an Advanced Option for a filter. Advanced Options in filters allow you to build up filtering conditions that use a mixture of AND and OR boolean operators across multiple filter line items. For example, (1 AND 2) OR 3 finds records that match both the first two filter line items or the third.</td>
</tr>
<tr>
<td>columns</td>
<td>string[]</td>
<td>The list of fields in the list view. The field name relative to the object name, for example MyCustomField__c, is specified for each custom field.</td>
</tr>
</tbody>
</table>

Note: Field names in the ListView columns don’t always match their API name counterparts. In particular, if person accounts is enabled in your organization, standard fields
merged from a contact into an account start with the PC_ prefix, while the corresponding API name starts with the Person prefix. For example, the ListView column name is PC_Email for a corresponding API field name of PersonEmail.

division  | string  | If your organization uses divisions to segment data and you have the "Affected by Divisions" permission, records in the list view must match this division. This field is only available if you are searching all records.

This field is available in API version 17.0 and later.

filterScope  | FilterScope (enumeration of type string)  | Required. This field indicates whether you are filtering by owner or viewing all records.

filters  | ListViewFilter[]  | The list of filter line items.

fullName  | string  | Required. Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See createMetadata() to see an example of this field specified for a call.

label  | string  | Required. The list view name.

language  | Language  | The language used for filtering if your organization uses the Translation Workbench and you are using the startsWith or contains operator. The values entered as search terms must be in the same language as the filter language.

For a list of valid language values, see Language.

This field is available in API version 17.0 and later.

queue  | string  | The name of a queue. Objects are sometimes assigned to a queue so that the users who have access to the queue can monitor and manage them. When you create a queue, a corresponding list view is automatically created. See "Create Queues" in the Salesforce online help.

sharedTo  | SharedTo  | Sharing access for the list view.

This field is available in API version 17.0 and later.

**ListViewFilter**

ListViewFilter represents a filter line item.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>string</td>
<td>Required. Represents the field specified in the filter.</td>
</tr>
</tbody>
</table>
**Metadata Types**

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operation</td>
<td>FilterOperation (enumeration of type string)</td>
<td>Required. The operation used by the filter, such as equals. The valid values are listed in FilterOperation.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Represents the value of the filter item being operated upon, for example, if the filter is my_number_field__c &gt; 1, the value of value is 1.</td>
</tr>
</tbody>
</table>

### FilterScope

This is an enumeration of type string that represents the filtering criteria for the records. The valid values are listed in the table below:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Everything</td>
<td>All records, for example All Opportunities.</td>
</tr>
<tr>
<td>Mine</td>
<td>Records owned by the user running the list view, for example My Opportunities.</td>
</tr>
<tr>
<td>MineAndMyGroups</td>
<td>Records owned by the user running the list view, and records assigned to the user’s queues.</td>
</tr>
<tr>
<td>Queue</td>
<td>Records assigned to a queue.</td>
</tr>
<tr>
<td>Delegated</td>
<td>Records delegated to another user for action: for example, a delegated task. This option is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>MyTerritory</td>
<td>Records in the territory of the user seeing the list view. This option is available if territory management is enabled for your organization. This option is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>MyTeamTerritory</td>
<td>Records in the territory of the team of the user seeing the list view. This option is available if territory management is enabled for your organization. This option is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>Team</td>
<td>Records assigned to a team. This option is available in API version 17.0 and later.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

A sample XML definition of a list view in a custom object is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  ...<listViews>
    <fullName>All_Mileages</fullName>
    <filterScope>all</filterScope>
    <label>All Mileages</label>
  </listViews>
  <listViews>
    <fullName>My_Mileages</fullName>
    <booleanFilter>1 AND 2</booleanFilter>
    <columns>NAME</columns>
    <columns>CREATED_DATE</columns>
</listViews>
```
Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
   CustomObject
   Sample package.xml Manifest Files

NamedFilter

Represents the metadata associated with a lookup filter. This metadata type is used to create, update, or delete lookup filter definitions. This component has been removed as of API version 30.0 and is only available in previous API versions. The metadata associated with a lookup filter is now represented by the lookupFilter field in the CustomField component.

This type extends the Metadata metadata type and inherits its fullName field. You can also use this metadata type to work with customizations of lookup filters on standard fields.

Note: The namedFilter appears as a child of the target object of the associated lookup field.

Declarative Metadata File Suffix and Directory Location

Lookup filters are defined as part of the custom object or standard object definition. See CustomObject for more information.

Note: Retrieving a component of this metadata type in a project makes the component appear in any Profile and PermissionSet components that are retrieved in the same package.

Version

Lookup filters are available in API version 17.0 and later.

Fields

Unless otherwise noted, all fields are createable, filterable, and nillable.
Metadata Types

### Field Name | Field Type | Description
--- | --- | ---
active | boolean | Required. Indicates whether or not the lookup filter is active.
booleanFilter | string | Specifies advanced filter conditions.
description | string | A description of what this filter does.
errorMessage | string | The error message that appears if the lookup filter fails.
field | string | Required. The `fullName` of the custom or standard field associated with the lookup filter. You can associate one relationship field with each lookup filter, and vice-versa.

**Note:** You cannot update a field associated with a lookup filter.

filterItems | FilterItems[] | Required. The set of filter conditions.
infoMessage | string | The information message displayed on the page. Use to describe things the user might not understand, such as why certain items are excluded in the lookup filter.

fullName | string | Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See `createMetadata()` to see an example of this field specified for a call. This value cannot be `null`.
isOptional | boolean | Required. Indicates whether or not the lookup filter is optional.
name | string | Required. The name of the lookup filter. If you create this field in the user interface, a name is automatically assigned. If you create this field through Metadata API, you must include the `name` field.
sourceObject | string | The object that contains the lookup field that uses this lookup filter. Set this field if the lookup filter references fields on the source object.

Lookup filters use additional data types. For more information, see [Metadata Field Types](#).

### FilterItems

FilterItems contains the following properties:

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>Represents the field specified in the filter.</td>
</tr>
<tr>
<td>operation</td>
<td><code>FilterOperation</code> (enumeration of type string)</td>
<td>Represents the filter operation for this filter item. Valid values are enumerated in <code>FilterOperation</code>.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Represents the value of the filter item being operated upon, for example, if the filter is my_number_field__c &gt; 1, the value of value is 1.</td>
</tr>
</tbody>
</table>

**FilterOperation**

This is an enumeration of type string that lists different filter operations. Valid values are:

- equals
- notEqual
- lessThan
- greaterThan
- lessOrEqual
- greaterOrEqual
- contains
- notContain
- startsWith
- includes
- excludes

**Declarative Metadata Sample Definition**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
....
  <namedfilters>
    <fullName>nf_Acc</fullName>
    <active>true</active>
    <booleanFilter>1 OR 2</booleanFilter>
    <field>Account.lk__c</field>
    <filterItems>
      <field>Account.Phone</field>
      <operation>notEqual</operation>
      <value>x</value>
    </filterItems>
    <filterItems>
      <field>Account.Fax</field>
      <operation>notEqual</operation>
      <value>y</value>
    </filterItems>
  </namedfilters>
  <name>Acc</name>
  <sourceObject>Account</sourceObject>
....
</CustomObject>
```
Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- CustomObject
- Picklist (Including Dependent Picklist)
- Metadata
- CustomField

Picklist (Including Dependent Picklist)

Deprecated. Represents a picklist (or dependent picklist) definition for a custom field in a custom object or a custom or standard field in a standard object, such as an account.

Version

Use this type in API version 37.0 and earlier only. In later versions, Picklist is replaced by ValueSet. Picklists for custom fields in custom objects are available in API version 12.0 and later. Picklists for custom or standard fields in standard objects, such as accounts, are available in API version 16.0 and later.

Declarative Metadata File Suffix and Directory Location

Picklist definitions are included in the custom object and field with which they are associated.

Fields

Picklist contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>controllingField</td>
<td>string</td>
<td>The fullName of the controlling field if this is a dependent picklist. A dependent picklist works in conjunction with</td>
</tr>
<tr>
<td></td>
<td></td>
<td>a controlling picklist or checkbox to filter the available options. The value chosen in the controlling field</td>
</tr>
<tr>
<td></td>
<td></td>
<td>affects the values available in the dependent field. This field is available in API version 14.0 and later.</td>
</tr>
<tr>
<td>picklistValues</td>
<td>PicklistValue[]</td>
<td>Required. Represents a set of values for a picklist.</td>
</tr>
<tr>
<td>restrictedPicklist</td>
<td>boolean</td>
<td>Indicates whether the picklist’s value list is restricted. With a restricted picklist, only an admin can add or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>change values; users can’t load or remove values through the API. By default this value is false. This field is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>available in API version 37.0 and later.</td>
</tr>
<tr>
<td>sorted</td>
<td>boolean</td>
<td>Indicates whether values should be sorted (true), or not (false). By default this value is false.</td>
</tr>
</tbody>
</table>
Java Sample

The following sample uses a picklist. For a complete sample of using a picklist with record types and profiles, see Profile on page 657.

```java
public void setPicklistValues() {
    // Create a picklist
    Picklist expenseStatus = new Picklist();
    PicklistValue unsubmitted = new PicklistValue();
    unsubmitted.setFullName("Unsubmitted");
    PicklistValue submitted = new PicklistValue();
    submitted.setFullName("Submitted");
    PicklistValue approved = new PicklistValue();
    approved.setFullName("Approved");
    PicklistValue rejected = new PicklistValue();
    rejected.setFullName("Rejected");
    expenseStatus.setPicklistValues(new PicklistValue[]
        {unsubmitted, submitted, approved, rejected});

    CustomField expenseStatusField = new CustomField();
    expenseStatusField.setFullName("
        ExpenseReport__c.ExpenseStatus__c"");
    expenseStatusField.setLabel("Expense Report Status");
    expenseStatusField.setType(FieldType.Picklist);
    expenseStatusField.setPicklist(expenseStatus);
    try {
        AsyncResult[] ars =
            metadataConnection.create(new Metadata[] {expenseStatusField});
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

Declarative Metadata Sample Definition

The following sample shows usage for picklists, including dependent picklists, in a custom object. The isAmerican__c checkbox controls the list of manufacturers shown in the manufacturer__c picklist. The manufacturer__c checkbox in turn controls the list of models shown in the model__c picklist.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
    <deploymentStatus>Deployed</deploymentStatus>
    <enableActivities>true</enableActivities>
    <fields>
        <fullName>isAmerican__c</fullName>
        <defaultValue>false</defaultValue>
        <label>American Only</label>
        <type>Checkbox</type>
    </fields>
    <fields>
        <fullName>manufacturer__c</fullName>
        <label>Manufacturer</label>
        <picklist>
            <controllingField>isAmerican__c</controllingField>
        </picklistValues>
    </fields>
</CustomObject>
```
<full_name>Chrysler</full_name>
<controlling_field_values>checked</controlling_field_values>
<default>false</default>
</picklist_values>
<picklist_values>
<full_name>Ford</full_name>
<controlling_field_values>checked</controlling_field_values>
<default>false</default>
</picklist_values>
<picklist_values>
<full_name>Honda</full_name>
<controlling_field_values>unchecked</controlling_field_values>
<default>false</default>
</picklist_values>
<picklist_values>
<full_name>Toyota</full_name>
<controlling_field_values>unchecked</controlling_field_values>
<default>false</default>
</picklist_values>
<sorted>false</sorted>
</picklist>
</fields>
<fields>
<full_name>model__c</full_name>
<label>Model</label>
<picklist>
<controlling_field>manufacturer__c</controlling_field>
<picklist_values>
<full_name>Mustang</full_name>
<controlling_field_values>Ford</controlling_field_values>
<default>false</default>
</picklist_values>
<picklist_values>
<full_name>Taurus</full_name>
<controlling_field_values>Ford</controlling_field_values>
<default>false</default>
</picklist_values>
<picklist_values>
<full_name>PT Cruiser</full_name>
<controlling_field_values>Chrysler</controlling_field_values>
<default>false</default>
</picklist_values>
<picklist_values>
<full_name>Pacifica</full_name>
<controlling_field_values>Chrysler</controlling_field_values>
<default>false</default>
</picklist_values>
<picklist_values>
<full_name>Accord</full_name>
<controlling_field_values>Honda</controlling_field_values>
<default>false</default>
</picklist_values>
</picklist>
The following sample shows usage for the standard `Stage` field in opportunities.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
    <fields>
        <fullName>StageName</fullName>
        <picklist>
            <picklistValues>
                <fullName>Prospecting</fullName>
                <default>false</default>
                <forecastCategory>Pipeline</forecastCategory>
                <probability>10</probability>
            </picklistValues>
            <picklistValues>
                <fullName>Qualification</fullName>
                <default>false</default>
                <forecastCategory>Pipeline</forecastCategory>
                <probability>10</probability>
            </picklistValues>
            <picklistValues>
                <fullName>Needs Analysis</fullName>
                <default>false</default>
                <forecastCategory>Pipeline</forecastCategory>
                <probability>20</probability>
            </picklistValues>
            ...  
        </picklist>
    </fields>
</CustomObject>
```
Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

RecordType

Represents the metadata associated with a record type. Record types let you offer different business processes, picklist values, and page layouts to different users. Use this metadata type to create, update, or delete record type definitions for a custom object.

For more information, see “Tailor Business Processes to Different Users” in the Salesforce Help. This type extends the Metadata metadata type and inherits its fullName field.

When an object is accessible to a user, all record type information for the object is readable by that user. We strongly recommend against storing sensitive information in the record type description, name, or label. Instead, store sensitive information in a separate object or fields to which you’ve applied appropriate access controls.

Note: Retrieving a component of this metadata type in a project makes the component appear in any Profile and PermissionSet components that are retrieved in the same package.

Version

Record types are available in API version 12.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Indicates whether the record type is active.</td>
</tr>
<tr>
<td>businessProcess</td>
<td>string</td>
<td>The fullName of the business process associated with the record type. This field is required in record types for lead, opportunity, solution, and case, and not allowed otherwise. See BusinessProcess on page 307. This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>compactLayoutAssignment</td>
<td>string</td>
<td>Represents the compact layout that is assigned to the record type. This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Record type description. Maximum of 255 characters.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Record type name. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. If this field contained characters before version 14.0 that are no longer allowed, the characters were stripped out of this field, and the previous value of the field was saved in the label field.</td>
</tr>
</tbody>
</table>
Inherited from the Metadata component, this field is not defined in the WSDL for this component. It must be specified when creating, updating, or deleting. See create() to see an example of this field specified for a call. This value cannot be null.

**label**
- **Field Type**: string
- **Description**: Required. Descriptive label for the record type. The list of characters allowed in the fullName field has been reduced for versions 14.0 and later. This field contains the value contained in the fullName field before version 14.0.

**picklistValues**
- **Field Type**: RecordTypePicklistValue[]
- **Description**: Represents a set of values for a picklist.

### RecordTypePicklistValue

RecordTypePicklistValue represents the combination of picklists and valid values that define a record type:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>picklist</td>
<td>string</td>
<td>Required. The name of the picklist.</td>
</tr>
<tr>
<td>values</td>
<td>PicklistValue</td>
<td>One or more of the picklist values in the picklist. Each value defined is available in the record type that contains this component.</td>
</tr>
</tbody>
</table>

### Java Sample

The following sample uses two record types. For the complete sample that includes profiles and picklists, see Profile on page 657.

```java
public void recordTypeSample() {
    try {
        // Employees and managers have different access
        // to the state of the expense sheet
        RecordType edit = new RecordType();
        edit.setFullName("ExpenseReport__c.Edit");
        edit.setLabel("ExpenseReport__c.Label");
        PicklistValue unsubmitted = new PicklistValue();
        unsubmitted.setFullName("Unsubmitted");
        PicklistValue submitted = new PicklistValue();
        submitted.setFullName("Submitted");
        RecordTypePicklistValue editStatuses = new RecordTypePicklistValue();
        editStatuses.setPicklist("ExpenseStatus__c");
        editStatuses.setValues(new PicklistValue[] {unsubmitted, submitted});
        edit.setPicklistValues(new RecordTypePicklistValue[] {editStatuses});
        AsyncResult[] arsEdit = metadataConnection.create(new Metadata[] {edit});
    }
```
Declarative Metadata Sample Definition

The definition of a record type in a custom object is shown below:

```
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  . . .
  <recordTypes>
    <fullName>My First Recordtype</fullName>
  </recordTypes>
  . . .
</CustomObject>
```

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SearchLayouts

Represents the metadata associated with the Search Layouts for an object. You can customize which fields to display for users in search results, search filter fields, lookup dialogs, and recent record lists on tab home pages. You can access SearchLayouts only by accessing its encompassing CustomObject.

For more information, see “Customize Search Layouts” and “Customize Search Layouts for Custom Objects” in the Salesforce online help.

Version

Search layouts for custom objects are available in API version 14.0 and later. The ability to modify search layouts for standard objects (except events and tasks) is available in API version 27.0 and later.
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customTabListAdditionalFields</td>
<td>string[]</td>
<td>The list of fields displayed in the Recent Object Name list view on a tab associated with the object. The name field is mandatory and is always displayed as the first column header, so it is not included in this list; all additional fields are included. The field name relative to the object name, for example MyCustomField__c, is specified for each custom field.</td>
</tr>
<tr>
<td>excludedStandardButtons</td>
<td>string[]</td>
<td>The list of standard buttons excluded from the search layout.</td>
</tr>
<tr>
<td>listViewButtons</td>
<td>string[]</td>
<td>The list of buttons available in list views for an object. The list of standard buttons excluded from the search layout.</td>
</tr>
<tr>
<td>lookupDialogsAdditionalFields</td>
<td>string[]</td>
<td>The list of fields displayed in a lookup dialog for the object. The name field is mandatory and is always displayed as the first column header, so it is not included in this list; all additional fields are included. The field name relative to the object name, for example MyCustomField__c, is specified for each custom field.</td>
</tr>
<tr>
<td>lookupFilterFields</td>
<td>string[]</td>
<td>The list of fields that can be used to filter enhanced lookups for an object. Enhanced lookups are optionally enabled by your administrator. The field name relative to the object name, for example MyCustomField__c, is specified for each custom field.</td>
</tr>
</tbody>
</table>

Salesforce objects often include one or more lookup fields that allow users to associate two records together in a relationship. For example, a contact record includes an Account lookup field that represents the relationship between the contact and the organization with which the contact is associated. A lookup search dialog helps you search for the record associated with the one being edited. Lookup filter fields allow you to filter your lookup search by a customized list of fields in the object.

This field is equivalent to the Lookup Dialogs in the Search Layouts related list on the object detail page in the application user interface. For more information, see “Standard and Enhanced Lookups in Salesforce Classic” in the Salesforce online help.

This field is equivalent to the Lookup Filter Fields in the Search Layouts related list on the object detail page.
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lookupPhoneDialogsAdditionalFields</td>
<td>string[]</td>
<td>The list of phone-related fields displayed in a lookup dialog for the object. The name field is mandatory and is always displayed as the first column header, so it is not included in this list; all additional fields are included. The field name relative to the object name, for example MyCustomField__c, is specified for each custom field. This list enables integration of the fields with a softphone dial pad. This field is equivalent to the Lookup Phone Dialogs in the Search Layouts related list on the object detail page in the application user interface.</td>
</tr>
<tr>
<td>massQuickActions</td>
<td>string[]</td>
<td>The list of actions that you can use to perform mass quick action on records. Use this field to add an existing create or update action. You can perform mass quick action on custom objects and all standard objects that support quick actions and have a search layout in Lightning Experience. This includes but isn’t limited to cases, leads, accounts, campaigns, contacts, opportunities, and work orders.</td>
</tr>
<tr>
<td>searchFilterFields</td>
<td>string[]</td>
<td>The list of fields that can be used to filter a search for the object. The field name relative to the object name, for example MyCustomField__c, is specified for each custom field. This field is equivalent to the Search Filter Fields in the Search Layouts related list on the object detail page in the application user interface.</td>
</tr>
<tr>
<td>searchResultsAdditionalFields</td>
<td>string[]</td>
<td>The list of fields displayed in a search result for the object. The name field is mandatory and is always displayed as the first column header, so it is not included in this list; all additional fields are included. The field name relative to the object name, for example MyCustomField__c, is specified for each custom field. This field is equivalent to the Search Results in the Search Layouts related list on the object detail page in the application user interface.</td>
</tr>
<tr>
<td>searchResultsCustomButtons</td>
<td>string[]</td>
<td>The list of custom buttons available in a search result for the object. The actions associated with the buttons can be applied to any of the records returned in the search result.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

A sample definition of search layouts in an object is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?><CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  ...
  <searchLayouts>
    <listViewButtons>New</listViewButtons>
    <listViewButtons>Accept</listViewButtons>
    <listViewButtons>ChangeOwner</listViewButtons>
    <lookupDialogsAdditionalFields>firstQuote__c</lookupDialogsAdditionalFields>
    <lookupDialogsAdditionalFields>finalQuote__c</lookupDialogsAdditionalFields>
    <massQuickActions>Create_MQA_Contact</massQuickActions>
    <searchResultsAdditionalFields>CREATEDBY_USER</searchResultsAdditionalFields>
  </searchLayouts>
  ...
</CustomObject>
```

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:

- CustomObject

SharingReason

Represents an Apex sharing reason, which is used to indicate why sharing was implemented for a custom object. Apex managed sharing allows developers to use Apex to programmatically share custom objects. When you use Apex managed sharing to share a custom object, only users with the “Modify All Data” permission can add or change the sharing on the custom object’s record, and the sharing access is maintained across record owner changes. For more information, see “Sharing Settings” in the Salesforce online help.

Use SharingReason to create, update, or delete sharing reason definitions for a custom object. This type extends the Metadata metadata type and inherits its `fullName` field.

Version

Sharing reasons are available in API version 14.0 and later.
Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. Sharing reason name. The __c suffix is appended to custom sharing reasons. Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See createMetadata() to see an example of this field specified for a call.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Descriptive label for the sharing reason. Maximum of 40 characters.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The definition of a sharing reason in a custom object:

```
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  . . .
  <sharingReasons>
    <fullName>recruiter__c</fullName>
    <label>Recruiter</label>
  </sharingReasons>
  . . .
</CustomObject>
```

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SharingRecalculation

Represents Apex classes that recalculate the Apex managed sharing for a specific custom object.

For more information, see “Recalculate Apex Managed Sharing” in the Salesforce online help.

Version

Sharing recalculations are available in API version 14.0 and later.
Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>string</td>
<td>Required. The Apex class that recalculates the Apex sharing for a custom object. This class must implement the <code>Database.Batchable</code> interface.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The definition of a sharing recalculation in a custom object:

```xml
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  . . .
  <sharingRecalculations>
    <className>RecruiterRecalculation</className>
  </sharingRecalculations>
  . . .
</CustomObject>
```

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

ValidationRule

Represents a validation rule, which is used to verify that the data a user enters in a record is valid and can be saved. A validation rule contains a formula or expression that evaluates the data in one or more fields and returns a value of `true` or `false`. Validation rules also include an error message that your client application can display to the user when the rule returns a value of `true` due to invalid data.

This type extends the `Metadata` metadata type and inherits its `fullName` field.

As of API version 20.0, validation rules can’t have compound fields. Examples of compound fields include addresses, first and last names, dependent picklists, and dependent lookups.

As of API version 40.0, you can use validation rules with custom metadata types.

Version

Validation rules are available in API version 12.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Indicates whether this validation rule is active, (<code>true</code>), or not active (<code>false</code>).</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
**description** | string | A description of the validation rule.

**errorConditionFormula** | string | Required. The formula defined in the validation rule. If the formula returns a value of `true`, an error message is displayed. See “Define Validation Rules” in the Salesforce online help.

**errorDisplayField** | string | The fully specified name of a field in the application. If a value is supplied, the error message appears next to the specified field. If you do not specify a value or the field isn’t visible on the page layout, the value changes automatically to Top of Page.

**errorMessage** | string | Required. The message that appears if the validation rule fails. The message must be 255 characters or less.

**fullName** | string | The internal name of the object. White spaces and special characters are escaped for validity. The name must:

- Contain characters, letters, or the underscore (_) character
- Must start with a letter
- Can’t end with an underscore
- Can’t contain two consecutive underscore characters.

Inherited from the Metadata component, this field is not defined in the WSDL for this component. It must be specified when creating, updating, or deleting. See `create()` to see an example of this field specified for a call.

### Declarative Metadata Sample Definition

A sample XML definition of a validation rule in a custom object is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
  <deploymentStatus>Deployed</deploymentStatus>
  <fields>
    <fullName>Mommy_Cat__c</fullName>
    <label>Mommy Cat</label>
    <referenceTo>Cat__c</referenceTo>
    <relationshipName>Cats</relationshipName>
    <type>Lookup</type>
  </fields>
  <label>Cat</label>
  <nameField>
    <label>Cat Name</label>
    <type>Text</type>
  </nameField>
  <pluralLabel>Cats</pluralLabel>
  <sharingModel>ReadWrite</sharingModel>
  <validationRules>
    <fullName>CatsRule</fullName>
  </validationRules>
</CustomObject>
```
Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

WebLink

Represents a custom button or link defined in a custom object.

This type extends the Metadata metadata type and inherits its fullName field.

Version

WebLinks are available in API version 12.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>availability</td>
<td>WebLinkAvailability (enumeration of type string)</td>
<td>Required. Indicates whether the button or link is only available online (online, or if it is also available offline (offline).</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the button or link.</td>
</tr>
<tr>
<td>displayType</td>
<td>WebLinkDisplayType (enumeration of type string)</td>
<td>Represents how the button or link is rendered. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• link for a hyperlink</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• button for a button</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• massAction for a button attached to a related list</td>
</tr>
<tr>
<td>encodingKey</td>
<td>Encoding (enumeration of type string)</td>
<td>Required. The default encoding setting is Unicode: UTF-8. Change it if your template requires data in a different format. This is available if your content source is URL. Valid values include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• UTF-8—Unicode (UTF-8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ISO-8859-1—General US &amp; Western Europe (ISO-8859-1, ISO-LATIN-1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Shift_JIS—Japanese (Shift-JIS)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• x-SJIS_0213—Japanese (Shift-JIS_2004)</td>
</tr>
</tbody>
</table>
### WebLink Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|              |                  | - `ks_c_5601-1987`—Korean (ks_c_5601-1987)  
  - `Big5`—Traditional Chinese (Big5)  
  - `GB2312`—Simplified Chinese (GB2312)  
  - `Big5–HKSCS`—Traditional Chinese Hong Kong (Big5–HKSCS)                                                                                     |
| `fullName`   | string           | The name of the custom button or link with white spaces and special characters escaped for validity. The name can only contain characters, letters, and the underscore (_) character, must start with a letter, and cannot end with an underscore or contain two consecutive underscore characters.  
  Inherited from the Metadata component, this field is not defined in the WSDL for this component. It must be specified when creating, updating, or deleting. See `create()` to see an example of this field specified for a call. |
| `hasMenubar`| boolean          | If the `openType` is `newWindow`, this field indicates whether to show the browser menu bar for the window (true) or not (false). Otherwise, leave this field empty. |
| `hasScrollbars` | boolean       | If the `openType` is `newWindow`, this field indicates whether to show the scroll bars for the window (true) or not (false). Otherwise, leave this field empty. |
| `hasToolbar` | boolean          | If the `openType` is `newWindow`, this field indicates whether to show the browser toolbar for the window (true) or not (false). Otherwise, leave this field empty. |
| `height`     | int              | Height in pixels of the window opened by the custom button or link. Required if the `openType` is `newWindow`. Otherwise, leave this field empty. |
| `isResizable`| boolean          | If the `openType` is `newWindow`, this field indicates whether to allow resizing of the window (true) or not (false). Otherwise, leave this field empty. |
| `linkType`   | WebLinkType (enumeration of type string) | Required. Represents whether the content of the button or link is specified by a URL, an sControl, a JavaScript code block, or a Visualforce page.  
  - `url`  
  - `sControl`  
  - `javascript`  
  - `page`  
  - `flow`—Reserved for future use. |
<p>| <code>masterLabel</code>| string           | Master label for this object. This display value is the internal label that is not translated. |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>openType</td>
<td>WebLinkWindowType (enumeration of type string)</td>
<td>Required. When the button or link is clicked, specifies the window style that will be used to display the content. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• newWindow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• sidebar</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• noSidebar</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• replace</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• onClickJavaScript</td>
</tr>
<tr>
<td>page</td>
<td>string</td>
<td>If the value of linkType is page, this field represents the Visualforce page. Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>position</td>
<td>WebLinkPosition (enumeration of type string)</td>
<td>If the value of OpenType is newWindow, this field indicates how the new window should be displayed. Otherwise, don’t specify a value. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fullScreen</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• none</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• topLeft</td>
</tr>
<tr>
<td>protected</td>
<td>boolean</td>
<td>Required. Indicates whether this sub-component is protected (true) or not (false). Protected sub-components can’t be linked to or referenced by components or sub-components created in the installing organization.</td>
</tr>
<tr>
<td>requireRowSelection</td>
<td>boolean</td>
<td>If the openType is massAction, this field indicates whether to require individual row selection to execute the action for this button (true) or not (false). Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>scontrol</td>
<td>string</td>
<td>If the value of linkType is sControl, this field represents the name of the sControl. Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>showsLocation</td>
<td>boolean</td>
<td>If the openType is newWindow, this field indicates whether to show the browser location bar for the window (true) or not (false). Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>showsStatus</td>
<td>boolean</td>
<td>If the openType is newWindow, this field indicates whether or not to show the browser status bar for the window. Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>url</td>
<td>string</td>
<td>If the value of linkType is url, this is the URL value. If the value of linkType is javascript, this is the JavaScript content. If the value is neither of these, leave this field empty. Content must be escaped in a manner consistent with XML parsing rules.</td>
</tr>
<tr>
<td>width</td>
<td>int</td>
<td>Width in pixels of the window opened by the button or link. Required if the openType is newWindow. Otherwise, leave this field empty.</td>
</tr>
</tbody>
</table>
Java Sample

The following Java sample shows sample values for WebLink fields:

```java
public void WebLinkSample(String name) throws Exception {
    WebLink WebLink = new WebLink();
    // name variable represents the full name of the object
    // on which to create the WebLink, for example, customObject__c
    WebLink.setFullName(name + "_.googleButton");
    WebLink.setUrl("http://www.google.com");
    WebLink.setAvailability(WebLinkAvailability.online);
    WebLink.setLinkType(WebLinkType.url);
    WebLink.setEncodingKey(Encoding.fromString("UTF-8"));
    WebLink.setOpenType(WebLinkWindowType.newWindow);
    WebLink.setHeight(600);
    WebLink.setWidth(600);
    WebLink.setShowsLocation(false);
    WebLink.setHasScrollbars(true);
    WebLink.setHasToolbar(false);
    WebLink.setHasMenubar(false);
    WebLink.setShowsStatus(false);
    WebLink.setIsResizable(true);
    WebLink.setPosition(WebLinkPosition.none);
    WebLink.setMasterLabel("google");
    WebLink.setDisplayType(WebLinkDisplayType.link);

    AsyncResult[] asyncResults = metadataConnection.create(new WebLink[]{WebLink});
    // After the create() call completes, we must poll the results of checkStatus()
    //
}
```

Declarative Metadata Sample Definition

The following is the definition of a WebLink in a custom object. For related samples, see HomePageComponent and HomePageLayout.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObject xmlns="http://soap.sforce.com/2006/04/metadata">
    ....
    <WebLinks>
        <fullName>googleButton</fullName>
        <availability>online</availability>
        <displayType>link</displayType>
        <encodingKey>UTF-8</encodingKey>
        <hasMenubar>false</hasMenubar>
        <hasScrollbars>true</hasScrollbars>
        <hasToolbar>false</hasToolbar>
        <height>600</height>
        <isResizable>true</isResizable>
        <linkType>url</linkType>
        <masterLabel>google</masterLabel>
        <openType>newWindow</openType>
        <position>none</position>
        <protected>false</protected>
    </WebLinks>
</CustomObject>
```
Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- HomePageComponent
- HomePageLayout
- CustomPageWebLink

Metadata Field Types

These field types extend the field types described in the SOAP API Developer Guide.

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Objects</th>
<th>What the Field Contains</th>
</tr>
</thead>
<tbody>
<tr>
<td>CustomField</td>
<td>Custom object</td>
<td>Represents a custom field.</td>
</tr>
<tr>
<td></td>
<td>Custom field</td>
<td></td>
</tr>
<tr>
<td>DeleteConstraint</td>
<td>Custom field</td>
<td>A string that represents deletion options for lookup relationships. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SetNull</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Restrict</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cascade</td>
</tr>
<tr>
<td>DeploymentStatus</td>
<td>Custom object</td>
<td>A string which represents the deployment status of a custom object or field. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>Custom field</td>
<td>• InDevelopment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Deployed</td>
</tr>
<tr>
<td>FieldType</td>
<td>Custom field</td>
<td>Indicates the type of a custom field. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AutoNumber</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Lookup</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MasterDetail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MetadataRelationship</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Checkbox</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Currency</td>
</tr>
<tr>
<td>Field Type</td>
<td>Objects</td>
<td>What the Field Contains</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td></td>
<td>Date</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DateTime</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Email</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EncryptedText</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ExternalLookup</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IndirectLookup</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Percent</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Phone</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Picklist</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MultiselectPicklist</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Summary</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Text</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TextArea</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LongTextArea</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Url</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Hierarchy</td>
<td></td>
</tr>
<tr>
<td></td>
<td>File</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Html</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Location (use for geolocation fields)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Time</td>
<td></td>
</tr>
</tbody>
</table>

1 A `Number` custom field is internally represented as a field of type double. Setting the scale of the `Number` field to 0 gives you a double that behaves like an int.

Notation: This page is about Classic Encryption, not Shield Platform Encryption. What’s the difference?

### Gender

Indicates the gender of the noun that represents the object. This is used for languages where words need different treatment depending on their gender. Valid values are:

- Masculine
- Feminine
- Neuter
- AnimateMasculine (Slavic languages—currently Czech, Polish, Russian, Slovak, Slovenian, and Ukrainian)
- ClassI, ClassIII, ClassV, ClassVII, ClassIX, ClassXI, ClassXIV, ClassXV, ClassXVI, ClassXVII, ClassXVIII (African languages—currently Afrikaans, Xhosa, and Zulu)
## Field Type

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Objects</th>
<th>What the Field Contains</th>
</tr>
</thead>
<tbody>
<tr>
<td>What the Field Contains</td>
<td>Note: The following genders are displayed on the Rename Tabs and Labels page in Setup but are stored internally as “Feminine”. When setting them through the Metadata API, use “Feminine”.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Euter (Swedish)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Common (Dutch)</td>
</tr>
<tr>
<td>Picklist (Including Dependent</td>
<td>Custom field</td>
<td>(This field type isn’t used in Metadata API. CustomField includes this field type for Tooling API support). Represents a picklist, a set of labels and values that can be selected from a picklist.</td>
</tr>
<tr>
<td>Picklist)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SharingModel</td>
<td>Custom object</td>
<td>Represents the sharing model for the custom object. Depending on the object, valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Private</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ReadWrite</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ReadWriteTransfer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FullAccess</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ControlledByParent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ControlledByCampaign</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ControlledByLeadOrContact</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, the User object supports Private and Read values. Accounts, opportunities, and custom objects support Private, Read and ReadWrite values. Campaign members support ControlledByCampaign and ControlledByLeadOrContact.</td>
</tr>
<tr>
<td>StartsWith</td>
<td>Custom object</td>
<td>Indicates whether the noun starts with a vowel, consonant, or is a special character. This is used for languages where words need different treatment depending on the first character. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>Custom field</td>
<td>• Consonant</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Vowel</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Special (for nouns starting with z, or s plus consonants)</td>
</tr>
<tr>
<td>TreatBlanksAs</td>
<td>Custom field</td>
<td>Indicates how blanks should be treated. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BlankAsBlank</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BlankAsZero</td>
</tr>
<tr>
<td>ValueSet</td>
<td>Custom field</td>
<td>Represents a set of values that can be selected from a custom picklist field. Defines the valueSet of a the custom picklist field.</td>
</tr>
</tbody>
</table>
**ValueSet**

Represents a set of values that can be selected from a custom picklist field. Defines the valueSet of a the custom picklist field.

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>controllingField</td>
<td>string</td>
<td>The <em>fullName</em> of the controlling field if this is a dependent picklist. A controlling field can be a checkbox or picklist field, but in this case it’s a picklist. The controlling picklist filters the available values in the dependent picklist.</td>
</tr>
<tr>
<td>restricted</td>
<td>boolean</td>
<td>Whether the picklist’s values are limited to only the values defined by a Salesforce admin. Values are true or false.</td>
</tr>
<tr>
<td>valueSetDefinition</td>
<td>ValueSetValuesDefinition</td>
<td>Defines value-specific settings for a custom dependent picklist. Indicates whether the value set of the custom picklist field is sorted alphabetically.</td>
</tr>
<tr>
<td>valueSetName</td>
<td>string</td>
<td>The <em>masterLabel</em> of the global value set to be used for this picklist field.</td>
</tr>
<tr>
<td>valueSettings</td>
<td>ValueSettings</td>
<td>Used for the settings that describe a value in a custom picklist field. The picklist can have its own unique value set, or inherit the values from a global value set.</td>
</tr>
</tbody>
</table>

**ValueSetValuesDefinition**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sorted</td>
<td>boolean</td>
<td>Whether the picklist’s value set is displayed in alphabetical order in the user interface.</td>
</tr>
<tr>
<td>value</td>
<td>CustomValue</td>
<td>Required. The list of values for this local, custom picklist.</td>
</tr>
</tbody>
</table>

**ValueSettings**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>controllingFieldValue</td>
<td>string[]string</td>
<td>Applies only to dependent custom picklists. A list of values in the controlling or parent picklist (that the custom picklist values depend on).</td>
</tr>
<tr>
<td>valueName</td>
<td>string</td>
<td>Defines the values in the custom dependent picklist.</td>
</tr>
</tbody>
</table>

**CustomObjectTranslation**

This metadata type allows you to translate custom objects for a variety of languages.

This type extends the *Metadata* metadata type and inherits its *fullName* field. The ability to translate component labels is part of the Translation Workbench. For more information, see “Enable and Disable the Translation Workbench” in the Salesforce online help.
Declarative Metadata File Suffix and Directory Location

Translations are stored in a file with a format of `customObjectName__c-lang.objectTranslation`, where `customObjectName__c` is the custom object name, and `lang` is the translation language. A sample file name for German translations is `myCustomObject__c-de.objectTranslation`.

Custom object translations are stored in the `objectTranslations` folder in the corresponding package directory.

Version

CustomObjectTranslation components are available in API version 14.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>caseValues</td>
<td>ObjectNameCaseValue[]</td>
<td>Different combinations of the custom object with regard to article, plural, possessive, and case.</td>
</tr>
<tr>
<td>fields</td>
<td>CustomFieldTranslation[]</td>
<td>A list of translations for the custom fields associated with the custom object.</td>
</tr>
<tr>
<td>fieldSets</td>
<td>FieldSetTranslation[]</td>
<td>A list of field set translations. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The name of the custom object and the translation language with a format of <code>customObjectName-lang</code>, where <code>customObjectName</code> is the custom object name, and <code>lang</code> is the translation language.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See <code>createMetadata()</code> to see an example of this field specified for a call.</td>
</tr>
<tr>
<td>gender</td>
<td>Gender</td>
<td>Indicates the gender of the noun that represents the object. This is used for languages where words need different treatment depending on their gender.</td>
</tr>
<tr>
<td>layouts</td>
<td>LayoutTranslation[]</td>
<td>A list of page layout translations.</td>
</tr>
<tr>
<td>nameFieldLabel</td>
<td>string</td>
<td>The label for the name field. Maximum of 765 characters.</td>
</tr>
<tr>
<td>namedFilters</td>
<td>NamedFilterTranslation[]</td>
<td>A list of translations for lookup filter error messages associated with the custom object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field has been removed as of API version 30.0 and is only available in prior versions. The translation metadata associated with a lookup filter is now represented by the <code>lookupFilter</code> field in the CustomFieldTranslation subtype.</td>
</tr>
<tr>
<td>quickActions</td>
<td>QuickActionTranslation[]</td>
<td>A list of translations for actions.</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recordTypes</td>
<td>RecordTypeTranslation[]</td>
<td>A list of record type translations.</td>
</tr>
<tr>
<td>sharingReasons</td>
<td>SharingReasonTranslation[]</td>
<td>A list of sharing reason translations.</td>
</tr>
<tr>
<td>startsWith</td>
<td>StartsWith (enumeration of type string)</td>
<td>Indicates whether the noun starts with a vowel, consonant, or is a special character. This is used for languages where words need different treatment depending on the first character.</td>
</tr>
<tr>
<td>validationRules</td>
<td>ValidationRuleTranslation[]</td>
<td>A list of validation rule translations.</td>
</tr>
<tr>
<td>webLinks</td>
<td>WebLinkTranslation[]</td>
<td>A list of web link translations.</td>
</tr>
<tr>
<td>workflowTasks</td>
<td>WorkflowTaskTranslation[]</td>
<td>A list of workflow task translations.</td>
</tr>
</tbody>
</table>

### CustomFieldTranslation

CustomFieldTranslation contains details for a custom field translation. In API versions 37.0 and earlier standard picklist values could be translated with CustomFieldTranslation. In API version 38.0, use StandardValueSetTranslation instead. For more details, see CustomField.

> **Note:** Not every language supports all the possible values for the fields in CustomFieldTranslation. For language-specific supported values, see the fully supported languages and end-user languages appendices.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>caseValues</td>
<td>ObjectNameCaseValue[]</td>
<td>Different combinations of the custom object with regard to article, plural, possessive, and case. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Translation for the custom field description.</td>
</tr>
<tr>
<td>gender</td>
<td>Gender</td>
<td>Indicates the gender of the noun that represents the object. This is used for languages where words need different treatment depending on their gender. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>help</td>
<td>string</td>
<td>Translation for the text that displays in the field-level help hover text for this field.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Translation for the label. Maximum of 765 characters.</td>
</tr>
<tr>
<td>lookupFilter</td>
<td>LookupFilterTranslation</td>
<td>Represents the translation metadata associated with a lookup filter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> LookupFilter is not supported on the article type object.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the field relative to the custom object; for example, MyField__c.</td>
</tr>
</tbody>
</table>

356
### FieldSetTranslation

FieldSetTranslation contains details for a field set translation. For more details, see FieldSet. Available in API 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Translation for the field set label. Maximum of 765 characters.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The field set name.</td>
</tr>
</tbody>
</table>

### LayoutTranslation

LayoutTranslation contains details for a page layout translation. For more details, see Fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layout</td>
<td>string</td>
<td>Required. The layout name.</td>
</tr>
<tr>
<td>layoutType</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>sections</td>
<td>LayoutSectionTranslation[]</td>
<td>An array of layout section translations.</td>
</tr>
</tbody>
</table>

### LayoutSectionTranslation

LayoutSectionTranslation contains details for a page layout section translation. For more details, see LayoutSection.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Translation for the label. Maximum of 765 characters.</td>
</tr>
<tr>
<td>section</td>
<td>string</td>
<td>Required. The section name.</td>
</tr>
</tbody>
</table>
LookupFilterTranslation

LookupFilterTranslation shows a translation for a lookup filter error message associated with the custom object. Replaces NamedFilterTranslation.

LookupFilterTranslation is available in API version 30.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorMessage</td>
<td>string</td>
<td>The error message that appears if the lookup filter fails.</td>
</tr>
<tr>
<td>informationalMessage</td>
<td>string</td>
<td>The information message displayed on the page. Use to describe things the user might not understand, such as why certain items are excluded in the lookup filter.</td>
</tr>
</tbody>
</table>

NamedFilterTranslation

NamedFilterTranslation has been removed as of API version 30.0 and is only available in previous API versions.

NamedFilterTranslation shows a list of translations for lookup filter error messages associated with the custom object. See NamedFilter for more information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorMessage</td>
<td>string</td>
<td>The error message that appears if the lookup filter fails.</td>
</tr>
<tr>
<td>informationalMessage</td>
<td>string</td>
<td>The information message displayed on the page. Use to describe things the user might not understand, such as why certain items are excluded in the lookup filter.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the lookup filter. If you create this field in the user interface, a name is automatically assigned. If you create this field through Metadata API, you must include the name field.</td>
</tr>
</tbody>
</table>

ObjectNameCaseValue

ObjectNameCaseValue supports multiple cases and definitions of the custom object name to allow usage in various grammatical contexts.

Note: Not every language supports all the possible values for the fields in ObjectNameCaseValue. For language-specific supported values, see the fully supported languages and end-user languages appendices.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>article</td>
<td>Article (enumeration of type string)</td>
<td>English has two types of articles: definite (<code>the</code>) and indefinite (<code>a</code>, <code>an</code>). The usage of these articles depends mainly on whether you are referring to any member of a group, or to a specific member of a group. The valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Definite</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Indefinite</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>None</strong></td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>caseType</td>
<td>CaseType (enumeration of type string)</td>
<td>The case of the custom object name. The valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Ablative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Accusative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Adessive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Allative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Causalfinal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Dative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Delative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Distributive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Elative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Essive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Essiveformal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Genitive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Illative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Inessive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Instrumental</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Lative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Locative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Nominative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Objective</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Partitive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Prepositional</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Subjective</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Sublative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Superessive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Termanative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Translative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Vocative</td>
</tr>
<tr>
<td>plural</td>
<td>boolean</td>
<td>Indicates whether the value field is plural (true) or singular (false).</td>
</tr>
<tr>
<td>possessive</td>
<td>Possessive (enumeration of type string)</td>
<td>The possessive case of a language is a grammatical case used to indicate a relationship of possession. The valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• First</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Second</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Required. The value or label in this grammatical context.</td>
</tr>
</tbody>
</table>
PicklistValueTranslation

PicklistValueTranslation contains details for translation of a picklist value from a local, custom picklist field. For more details, see Picklist (Including Dependent Picklist).

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The picklist value defined on the setup page in the application is your master label. The master label is displayed wherever a translated label is not available.</td>
</tr>
<tr>
<td>translation</td>
<td>string</td>
<td>Required. Translation for the value.</td>
</tr>
</tbody>
</table>

QuickActionTranslation

QuickActionTranslation contains details for an action label in the user interface. For more information, see QuickAction.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Translation for the label. Maximum of 765 characters.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The quick action name.</td>
</tr>
</tbody>
</table>

RecordTypeTranslation

RecordTypeTranslation contains details for a record type name translation. For more details, see RecordType.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Translation for the label. Maximum of 765 characters.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The record type name.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Translation for the record type description. Available in API version 42.0 and later.</td>
</tr>
</tbody>
</table>

SharingReasonTranslation

SharingReasonTranslation contains details for a sharing reason translation. For more details, see SharingReason.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Translation for the sharing reason.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The sharing reason name.</td>
</tr>
</tbody>
</table>

ValidationRuleTranslation

ValidationRuleTranslation contains details for a validation rule translation. For more details, see ValidationRule.
### CustomObjectTranslation

**WebLinkTranslation**

WebLinkTranslation contains details for a web link translation. For more details, see [WebLink](#).

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Translation for the web link label. Maximum of 765 characters.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The web link name.</td>
</tr>
</tbody>
</table>

**WorkflowTaskTranslation**

WorkflowTaskTranslation contains details for a workflow task translation. For more details, see [Workflow](#).

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Translation for the workflow task description.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The workflow task name.</td>
</tr>
<tr>
<td>subject</td>
<td>string</td>
<td>Translation for the workflow task subject.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definitions

This is a sample XML definition of a CustomObjectTranslation for the Description__c object in German, with one custom field, Summary__c. The name and location of the file containing this definition would be `objectTranslations/Description__c-de.objectTranslation`.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObjectTranslation xmlns="http://soap.sforce.com/2006/04/metadata">
    <caseValues>
        <caseType>Nominative</caseType>
        <plural>false</plural>
        <value>Beschreibung</value>
    </caseValues>
    <caseValues>
        <caseType>Nominative</caseType>
        <plural>true</plural>
        <value>Beschreibungen</value>
    </caseValues>
    <caseValues>
        <caseType>Accusative</caseType>
        <plural>false</plural>
    </caseValues>
</CustomObjectTranslation>
```
This is a sample XML definition of a CustomObjectTranslation for the Account object, renaming Account to Client (Kunde) in German. The Account object has one standard field, account_number, and one custom field, Account_Code__c. The name and location of the file containing this definition would be objectTranslations/Account-de.objectTranslation.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomObjectTranslation xmlns="http://soap.sforce.com/2006/04/metadata">
  <caseValues>
    <caseType>Nominative</caseType>
    <plural>false</plural>
    <value>Kunde</value>
  </caseValues>
  <caseValues>
    <caseType>Nominative</caseType>
    <plural>true</plural>
    <value>Kunden</value>
  </caseValues>
  <caseValues>
    <caseType>Accusative</caseType>
    <plural>false</plural>
    <value>Kunden</value>
  </caseValues>
  <caseValues>
    <caseType>Dative</caseType>
    <plural>true</plural>
    <value>Kunden</value>
  </caseValues>
  <fields>
    <label>Zusammenfassung</label>
    <name>Summary_c</name>
  </fields>
  <gender>Feminine</gender>
  <nameFieldLabel>Beschreibungen</nameFieldLabel>
</CustomObjectTranslation>
```
<caseValues>
  <caseType>Accusative</caseType>
  <plural>true</plural>
  <value>Kunden</value>
</caseValues>
<caseValues>
  <caseType>Genitive</caseType>
  <plural>false</plural>
  <value>Kunden</value>
</caseValues>
<caseValues>
  <caseType>Genitive</caseType>
  <plural>true</plural>
  <value>Kunden</value>
</caseValues>
<caseValues>
  <caseType>Dative</caseType>
  <plural>false</plural>
  <value>Kunden</value>
</caseValues>
<caseValues>
  <caseType>Dative</caseType>
  <plural>true</plural>
  <value>Kunden</value>
</caseValues>
<fields>
  <caseValues>
    <caseType>Nominative</caseType>
    <plural>false</plural>
    <value>Kundennummer</value>
  </caseValues>
  <caseValues>
    <caseType>Nominative</caseType>
    <plural>true</plural>
    <value>Kundennummern</value>
  </caseValues>
  <gender>Feminine</gender>
  <name>account_number</name>
</fields>
<fields>
  <label>Kunden-Code</label>
  <name>Account_Code__c</name>
</fields>
<gender>Masculine</gender>
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
  - CustomObject
  - Translations

CustomPageWebLink

Represents a custom link defined in a home page component. This type extends the Metadata metadata type and inherits its fullName field.

All other custom links are stored as a WebLink in a CustomObject.

Declarative Metadata File Suffix and Directory Location

There is one file per custom link definition, stored in the weblinks folder in the corresponding package directory. The file suffix is .weblink.

Version

CustomPageWebLinks are available in API version 13.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>availability</td>
<td>WebLinkAvailability</td>
<td>Required. Indicates whether the link is only available online (online, or if it is also available offline (offline).</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the link.</td>
</tr>
<tr>
<td>displayType</td>
<td>WebLinkDisplayType</td>
<td>Represents how this link is rendered. Valid values:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• link for a hyperlink</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• button for a button</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• massAction for a button attached to a related list</td>
</tr>
<tr>
<td>encodingKey</td>
<td>Encoding (enumeration of type string)</td>
<td>Required. The default encoding setting is Unicode: UTF-8. Change it if your template requires data in a different format. This is available if your content source is URL. Valid values include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• UTF-8—Unicode (UTF-8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ISO-8859-1—General US &amp; Western Europe (ISO-8859-1, ISO-LATIN-1)</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The name used as a unique identifier for API access. The <code>fullName</code> can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.</td>
</tr>
<tr>
<td>hasMenubar</td>
<td>boolean</td>
<td>If the <code>openType</code> is <code>newWindow</code>, this field indicates whether to show the browser menu bar for the window (<code>true</code> or <code>false</code>). Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>hasScrollbars</td>
<td>boolean</td>
<td>If the <code>openType</code> is <code>newWindow</code>, this field indicates whether to show the scroll bars for the window (<code>true</code>) or not (<code>false</code>). Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>hasToolbar</td>
<td>boolean</td>
<td>If the <code>openType</code> is <code>newWindow</code>, this field indicates whether to show the browser toolbar for the window (<code>true</code>) or not (<code>false</code>). Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>height</td>
<td>int</td>
<td>Height in pixels of the window opened by the link. Required if the <code>openType</code> is <code>newWindow</code>. Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>isResizable</td>
<td>boolean</td>
<td>If the <code>openType</code> is <code>newWindow</code>, this field indicates whether to allow resizing of the window (<code>true</code>) or not (<code>false</code>). Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>linkType</td>
<td>WebLinkType (enumeration of type string)</td>
<td>Required. Represents whether the content of the button or link is specified by a URL, an sControl, a JavaScript code block, or a Visualforce page.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>url</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>sControl</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>javascript</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>page</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>flow</code>—Reserved for future use.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>The master label for the link.</td>
</tr>
<tr>
<td>openType</td>
<td>WebLinkWindowType (enumeration of type string)</td>
<td>Required. When the link is clicked, this field specifies the window style used to display the content. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>newWindow</code></td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• sidebar</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• noSidebar</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• replace</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• onClickJavaScript</td>
</tr>
<tr>
<td>page</td>
<td>string</td>
<td>If the value of linkType is page, this field represents the Visualforce</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page. Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>position</td>
<td>WebLinkPosition</td>
<td>If the openType is newWindow, this field indicates how the new window</td>
</tr>
<tr>
<td></td>
<td>(enumeration of</td>
<td>should be displayed. Otherwise, leave this field empty.</td>
</tr>
<tr>
<td></td>
<td>type string)</td>
<td>Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fullScreen</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• none</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• topLeft</td>
</tr>
<tr>
<td>protected</td>
<td>boolean</td>
<td>Required. Indicates whether this component is protected (true) or not (false).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Protected components cannot be linked to or referenced by components</td>
</tr>
<tr>
<td></td>
<td></td>
<td>created in the installing organization.</td>
</tr>
<tr>
<td>requireRowSelection</td>
<td>boolean</td>
<td>If the openType is massAction, this field indicates whether to require</td>
</tr>
<tr>
<td></td>
<td></td>
<td>individual row selection to execute the action for this button (true) or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>not (false). Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>scontrol</td>
<td>string</td>
<td>If the value of linkType is sControl, this field represents the name of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sControl. Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>showsLocation</td>
<td>boolean</td>
<td>If the openType is newWindow, this field indicates whether or not to show</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the browser location bar for the window. Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>showsStatus</td>
<td>boolean</td>
<td>If the openType is newWindow, this field indicates whether or not to show</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the browser status bar for the window. Otherwise, leave this field empty.</td>
</tr>
<tr>
<td>url</td>
<td>string</td>
<td>If the value of linkType is url, this field represents the URL value. If the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>value of linkType is javascript, this field represents the JavaScript</td>
</tr>
<tr>
<td></td>
<td></td>
<td>content. If the value is neither of these, leave this field empty.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Content must be escaped in a manner consistent with XML parsing rules.</td>
</tr>
<tr>
<td>width</td>
<td>int</td>
<td>Width in pixels of the window opened by the link. Required if the openType</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is newWindow. Otherwise, leave this field empty.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is the definition of a Weblink. For related samples, see HomePageComponent and HomePageLayout.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomPageWebLink xmlns="http://soap.sforce.com/2006/04/metadata">
  <availability>online</availability>
  <displayType>button</displayType>
  <encodingKey>UTF-8</encodingKey>
  <hasMenubar>false</hasMenubar>
  <hasScrollbars>true</hasScrollbars>
  <hasToolbar>false</hasToolbar>
  <height>600</height>
  <isResizable>true</isResizable>
  <linkType>url</linkType>
  <masterLabel>detailPageButon</masterLabel>
  <openType>newWindow</openType>
  <position>none</position>
  <protected>false</protected>
  <showsLocation>false</showsLocation>
  <showsStatus>false</showsStatus>
  <url>http://google.com</url>
</CustomPageWebLink>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
  - HomePageComponent
  - HomePageLayout
  - WebLink

CustomPermission

Represents a permission that grants access to a custom feature. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

CustomPermission components have the suffix .customPermission and are stored in the customPermissions folder.

Version

CustomPermission components are available in API version 31.0 and later.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectedApp</td>
<td>string</td>
<td>The name of the connected app that's associated with this permission. Limit: 80 characters.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The custom permission description. Limit: 255 characters.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The custom permission label. Limit: 80 characters.</td>
</tr>
<tr>
<td>requiredPermission</td>
<td>CustomPermissionDependencyRequired[]</td>
<td>Indicates which custom permissions are required by the parent custom permission. This field is available in API version 32.0 and later.</td>
</tr>
</tbody>
</table>

CustomPermissionDependencyRequired

CustomPermissionDependencyRequired determines whether a custom permission is required by the parent custom permission. A required custom permission must be enabled when its parent is enabled.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customPermission</td>
<td>string</td>
<td>Required. The custom permission name.</td>
</tr>
<tr>
<td>dependency</td>
<td>boolean</td>
<td>Required. Indicates whether this custom permission is required by the parent custom permission (true) or not (false).</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a CustomPermission component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomPermission xmlns="http://soap.sforce.com/2006/04/metadata">
  <connectedApp>Acme</connectedApp>
  <description>Read and edit access for Acme accounts.</description>
  <label>Acme Account Full Access</label>
  <requiredPermission>
    <customPermission>Acme_Account_Read</customPermission>
    <dependency>true</dependency>
  </requiredPermission>
</CustomPermission>
```

The following is an example package.xml that references the previous definition, as well as other custom permissions that are associated with a connected app.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    ...
  </types>
</Package>
```
CustomSite

Represents a Salesforce site. Create public websites and applications that are directly integrated with your Salesforce organization, but don’t require users to log in with a username and password.

For more information, see “Salesforce Sites” in the Salesforce online help.

Note: CustomSite does not currently support syndication feeds.

This type extends the Metadata metadata type and inherits its fullName field.

Declarative Metadata File Suffix and Directory Location

Lightning Platform CustomSite components are stored in the sites directory of the corresponding package directory. The file name matches the site name, and the extension is .site.

Version

Lightning Platform CustomSite components are available in API version 14.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Determines whether the site is active.</td>
</tr>
<tr>
<td>allowHomePage</td>
<td>boolean</td>
<td>Required. Determines whether the standard home page is visible to public users. This is a new field in API version 15.0.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>allowStandardAnswersPages</td>
<td>boolean</td>
<td>Determines whether the standard answer pages are visible to public users. This is a new field in API version 19.0.</td>
</tr>
<tr>
<td>allowStandardIdeasPages</td>
<td>boolean</td>
<td>Required. Determines whether the standard Ideas pages are visible to public users. This is a new field in API version 15.0.</td>
</tr>
<tr>
<td>allowStandardLookups</td>
<td>boolean</td>
<td>Required. Determines whether the standard lookup pages are visible to public users. This is a new field in API version 15.0.</td>
</tr>
<tr>
<td>allowStandardPortalPages</td>
<td>boolean</td>
<td>Required. When enabled, authenticated users in this site can access standard Salesforce pages as allowed by their access controls. When disabled, authenticated users in this site can't access standard Salesforce pages, even if their access controls allow it. If your site serves only Visualforce pages, disabling this setting helps add a layer of access protection to your site. This is a new field in API version 39.0.</td>
</tr>
<tr>
<td>allowStandardSearch</td>
<td>boolean</td>
<td>Required. Determines whether the standard search pages are visible to public users. This is a new field in API version 15.0.</td>
</tr>
<tr>
<td>analyticsTrackingCode</td>
<td>string</td>
<td>The tracking code associated with your site. This code can be used by services like Google Analytics to track page request data for your site. This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>authorizationRequiredPage</td>
<td>string</td>
<td>The name of the Visualforce page to be displayed when the guest user tries to access a page for which they are not authorized.</td>
</tr>
<tr>
<td>bandwidthExceededPage</td>
<td>string</td>
<td>The name of the Visualforce page to be displayed when the site has exceeded its bandwidth quota.</td>
</tr>
<tr>
<td>browserXssProtection</td>
<td>boolean</td>
<td>Required. Determines whether protection against reflected cross-site scripting attacks is enabled. If a reflected cross-site scripting attack is detected, the browser shows a blank page with no content. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>changePasswordPage</td>
<td>string</td>
<td>The name of the Visualforce page to be displayed when the portal user attempts to change their password for either the portal or for Chatter Answers, when enabled.</td>
</tr>
<tr>
<td>chatterAnswersForgotPasswordConfirmPage</td>
<td>string</td>
<td>The name of the Visualforce page to be displayed that informs the user that an email has been sent to them with a temporary password. This field is available if</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Chatter Answers is enabled for your organization. This field is available in API version 27.0 and later.</td>
<td>chatterAnswersForgotPasswordPage</td>
<td>string</td>
</tr>
<tr>
<td>The name of the Visualforce page to be displayed when a user clicks the link to retrieve a forgotten password. This field is available if Chatter Answers is enabled for your organization. This field is available in API version 27.0 and later.</td>
<td>chatterAnswersForgotPasswordPage</td>
<td>string</td>
</tr>
<tr>
<td>The name of the Visualforce page to be displayed when the user clicks the help link. This field is available if Chatter Answers is enabled for your organization. This field is available in API version 27.0 and later.</td>
<td>chatterAnswersHelpPage</td>
<td>string</td>
</tr>
<tr>
<td>The name of the Visualforce page to be displayed to allow users to log in to the portal. This field is available if Chatter Answers is enabled for your organization. This field is available in API version 27.0 and later.</td>
<td>chatterAnswersLoginPage</td>
<td>string</td>
</tr>
<tr>
<td>The name of the Visualforce page to be displayed to allow users to register themselves and access the portal. This field is available in API version 27.0 and later.</td>
<td>chatterAnswersRegistrationPage</td>
<td>string</td>
</tr>
<tr>
<td>Required. Sets the clickjack protection level. The options are:</td>
<td>clickjackProtectionLevel</td>
<td>SiteClickjackProtectionLevel (enumeration of type string)</td>
</tr>
<tr>
<td>• AllowAllFraming — Allow framing by any page (no protection)</td>
<td></td>
<td>Required. Sets the clickjack protection level. The options are:</td>
</tr>
<tr>
<td>• External — Allow framing of site or community pages on external domains (good protection)</td>
<td></td>
<td>• SameOriginOnly — Allow framing by the same origin only (recommended)</td>
</tr>
<tr>
<td>• NoFraming — Don’t allow framing by any page (most protection)</td>
<td></td>
<td>This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>Required. Determines whether the browser is prevented from inferring the MIME type from the document content. If enabled, it also prevents the browser from executing some malicious files (JavaScript, Stylesheet) as dynamic content. This field is available in API version 41.0 and later.</td>
<td>contentSniffingProtection</td>
<td>boolean</td>
</tr>
<tr>
<td>Required. Determines whether HTTP requests, including third-party domains, are upgraded to HTTPS. This field is available in API version 41.0 and later.</td>
<td>cspUpgradeInsecureRequests</td>
<td>boolean</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customWebAddresses</td>
<td>SiteWebAddress[]</td>
<td>The root custom URLs associated with the site. Saving or deploying a CustomSite replaces all root custom URLs in the site with the root custom URLs in this list. Custom URLs that use a non-root path prefix are not included in this list and are not affected when saving or deploying a CustomSite. This field is available in API version 21.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The site description.</td>
</tr>
<tr>
<td>enableAuraRequests</td>
<td>boolean</td>
<td>Determines whether guest users can view features available only in Lightning (true). If set to false, Lightning features don't load. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>favoriteIcon</td>
<td>string</td>
<td>The name of the file to be used for the icon that appears in the browser's address field when visiting the site. Sets the favorite icon for the entire site.</td>
</tr>
<tr>
<td>fileNotFoundPage</td>
<td>string</td>
<td>The name of the Visualforce page to be displayed when the guest user tries to access a non-existent page.</td>
</tr>
<tr>
<td>forgotPasswordPage</td>
<td>string</td>
<td>The name of the Visualforce page to be displayed when a user clicks the Forgot Password link on the site's login page. This field is only applicable for Communities sites.</td>
</tr>
<tr>
<td>genericErrorPage</td>
<td>string</td>
<td>The name of the Visualforce page to be displayed for errors not otherwise specified.</td>
</tr>
<tr>
<td>guestProfile</td>
<td>string</td>
<td>Read only. The name of the profile associated with the guest user.</td>
</tr>
<tr>
<td>inMaintenancePage</td>
<td>string</td>
<td>The name of the Visualforce page to be displayed when the site is down for maintenance.</td>
</tr>
<tr>
<td>inactiveIndexPage</td>
<td>string</td>
<td>The name of the Visualforce page set as the inactive site home page.</td>
</tr>
<tr>
<td>indexPage</td>
<td>string</td>
<td>Required. The name of the Visualforce page set as the active site home page.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The name of the site label in the Salesforce user interface.</td>
</tr>
<tr>
<td>portal</td>
<td>string</td>
<td>The name of the portal associated with this site for login access.</td>
</tr>
<tr>
<td>referrerPolicyOriginWhenCrossOrigin</td>
<td>boolean</td>
<td>Required. Determines whether the referrer header shows only Salesforce.com rather than the entire URL when loading a page. This feature eliminates the potential for a referrer header to reveal sensitive information.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>requireHttps</td>
<td>boolean</td>
<td>Required. Determines whether the site requires secure connections (true) or not (false). When false, the site operates normally via insecure connections instead of redirecting to a secure connection.</td>
</tr>
<tr>
<td>requireInsecurePortalAccess</td>
<td>boolean</td>
<td>Required. Determines whether to override your organization’s security settings and exclusively use HTTP when logging in to the associated portal from your site.</td>
</tr>
<tr>
<td>robotsTxtPage</td>
<td>string</td>
<td>The name of the Visualforce page to be displayed for the robots.txt file used by web crawlers.</td>
</tr>
<tr>
<td>serverIsDown</td>
<td>string</td>
<td>The name of the static resource to be displayed from the cache server when Salesforce servers are down. The static resource must be a public zip file 1 MB or smaller and must contain a page named maintenance.html at the root level of the zip file. Other resources in the zip file, such as images or CSS files, can follow any directory structure. This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>siteAdmin</td>
<td>string</td>
<td>The username of the site administrator.</td>
</tr>
<tr>
<td>siteRedirectMappings</td>
<td>SiteRedirectMapping[]</td>
<td>An array of all URL redirect rules set for your site. This field is available in API version 20.0 and later.</td>
</tr>
<tr>
<td>siteTemplate</td>
<td>string</td>
<td>The name of the Visualforce page to be used as the site template.</td>
</tr>
<tr>
<td>siteType</td>
<td>siteType</td>
<td>Required. Identifies whether the site is a Visualforce (Salesforce Sites), Site.com site, or ChatterNetwork (Salesforce Sites). This is a new field in API version 27.0.</td>
</tr>
<tr>
<td>subdomain</td>
<td>string</td>
<td>Required. Read only. The custom subdomain prefix for the site. For example, if your site URL is mycompany.force.com/partners, mycompany.force.com is the subdomain.</td>
</tr>
<tr>
<td>urlPathPrefix</td>
<td>string</td>
<td>The first part of the path on the site’s URL that distinguishes this site from other sites. For example, if your site URL is mycompany.force.com/partners, partners is the urlPathPrefix.</td>
</tr>
</tbody>
</table>
SiteRedirectMapping

SiteRedirectMapping represents a URL redirect rule on your Salesforce site. For more information, see “Salesforce Sites URL Redirects” in Salesforce Help.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| action   | SiteRedirect (enumeration of type string) | Required. The type of the redirect. Available string values are:  
  - Permanent  
  - Temporary |
| isActive | boolean                                 | The status of the redirect: active or inactive.  |
| source   | string                                  | Required. The URL that you want to redirect. It must be a relative URL, but can have any valid extension type, such as .html or .php. |
| target   | string                                  | Required. The new URL you want users to visit. It can be a relative URL or a fully-qualified URL with an http:// or https:// prefix. |

SiteWebAddress

Represents the web address of a Salesforce site.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>certificate</td>
<td>string</td>
<td>Identifies the certificate associated with the custom domain. If the custom domain is set up for Salesforce to serve HTTPS, this field indicates which certificate to use.</td>
</tr>
<tr>
<td>domainName</td>
<td>string</td>
<td>Required. The domain of the website, in the form of <a href="http://www.acme.com">www.acme.com</a>.</td>
</tr>
<tr>
<td>primary</td>
<td>boolean</td>
<td>Required. Indicates whether this is the primary domain (true). If false, this is not the primary domain.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

A sample XML definition of a site is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomSite xmlns="http://soap.sforce.com/2006/04/metadata">
  <active>true</active>
  <allowHomePage>true</allowHomePage>
</CustomSite>
```
**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- Portal

**CustomTab**

Represents a custom tab. Custom tabs let you display custom object data or other web content in Salesforce. When you add a custom tab to an app in Salesforce Classic, it displays as a tab. When you add a custom tab to an app in Lightning Experience, it displays as an item in the app's navigation bar. When a tab displays a custom object, the tab name is the same as the custom object name; for page, s-control, or URL tabs, the name is arbitrary.
For more information, see “Show Custom Object Data for Your Users” in Salesforce Help. This type extends the Metadata metadata type and inherits its fullName field.

**File Suffix and Directory Location**

The file suffix is .tab. There is one file for each tab, stored in the tabs folder in the corresponding package directory.

> Note: Retrieving a component of this metadata type in a project makes the component appear in any Profile and PermissionSet components that are retrieved in the same package.

**Version**

Tabs are available in API version 10.0 and later.

**Fields**

This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionOverrides</td>
<td>ActionOverride[]</td>
<td>A list of the action overrides that are assigned to the tab. Only one override is allowed per formFactor for a given tab. This field is available in API version 37.0 and later.</td>
</tr>
<tr>
<td>auraComponent</td>
<td>string</td>
<td>Indicates whether this tab is for a Lightning component (true) or not (false). If set to true, the name of the tab matches the name of the Lightning component. Only one of these fields should have a value set:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• auraComponent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• customObject</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• flexiPage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• page</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• scontrol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• url</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; Note: auraComponent is part of the Lightning Components (Beta) feature.</td>
</tr>
<tr>
<td>customObject</td>
<td>boolean</td>
<td>Indicates whether this tab is for a custom object (true) or not (false). If set to true, the name of the tab matches the name of the custom object. Only one of these fields should have a value set:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• auraComponent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• customObject</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• flexiPage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• page</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The optional description text for the tab.</td>
</tr>
<tr>
<td>flexiPage</td>
<td>string</td>
<td>The name of the Lightning page to display in this tab. Only one of these</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fields should have a value set:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• auraComponent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• customObject</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• flexiPage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• page</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• scontrol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• url</td>
</tr>
<tr>
<td>frameHeight</td>
<td>int</td>
<td>The height, in pixels of the tab frame. Required for s-control and page</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tab.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The name of the tab. The value of this field depends on the type of tab,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>and the API version.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For custom object tabs, the <code>fullName</code> is the developer-assigned name of</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the custom object (MyCustomObject__c, for example). For custom object tabs,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>this name must be the same as the custom object name, and <code>customObject</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>should be set to <code>true</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For web tabs, the <code>fullName</code> is the developer-assigned name of the tab</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(MyWebTab, for example).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The <code>fullName</code> can contain only underscores and alphanumeric characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>It must be unique, begin with a letter, not include spaces, not end with an</td>
</tr>
<tr>
<td></td>
<td></td>
<td>underscore, and not contain two consecutive underscores. This field is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>inherited from the <code>Metadata</code> component.</td>
</tr>
<tr>
<td>hasSidebar</td>
<td>boolean</td>
<td>Indicates if the tab displays the sidebar panel.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>This is the label of the tab, for web tabs only.</td>
</tr>
<tr>
<td>motif</td>
<td>string</td>
<td>Required. The tab style for the color scheme and icon for the custom tab.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, “Custom70: Handsaw,” is the handsaw icon.</td>
</tr>
<tr>
<td>page</td>
<td>string</td>
<td>The name of the Visualforce page to display in this tab. Only one of these</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fields should have a value set:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• auraComponent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• customObject</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• flexiPage</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>page</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>scontrol</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>url</em></td>
</tr>
</tbody>
</table>

**scontrol** string  The name of the s-control to display in this tab. Only one of these fields should have a value set:
- auraComponent
- customObject
- flexiPage
- page
- scontrol
- url

**splashPageLink** string  The custom link used as the introductory splash page when users click the tab. References a HomePageComponent.

**url** string  The URL for the external web-page to embed in this tab. Only one of these fields should have a value set:
- auraComponent
- customObject
- flexiPage
- page
- scontrol
- url

**urlEncodingKey** Encoding (enumeration of type string)  The default encoding setting is Unicode: UTF-8. Change it if you are passing information to a URL that requires data in a different format. This option is available when the value URL is selected in the tab type.

**Declarative Metadata Sample Definition**

The following is the definition of a tab:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CustomTab xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>Myriad Publishing</description>
  <frameHeight>600</frameHeight>
  <motif>Custom53: Bell</motif>
  <url>http://www.myriadpubs.com</url>
  <urlEncodingKey>UTF-8</urlEncodingKey>
</CustomTab>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- CustomApplication

CustomValue

Represents the definition of a value used in a global value set or local custom picklist. Custom picklist fields can be local and unique, or can inherit their values from a global picklist (called a global value set in API version 38.0). This type extends the Metadata metadata type and inherits its fullName field.

To deactivate a global picklist value, you can invoke an update() call on GlobalPicklist (API version 37.0) or GlobalValueSet (API version 38.0 and later) with the value omitted, or with the value's isActive field set to false. Or, you can invoke an update() call directly on GlobalPicklistValue (API version 37.0) or CustomValue (API version 38.0 and later) with the isActive field set to false.

Note: If picklist values are missing from a component definition, they get deactivated when deployed. Deactivation occurs for picklist values of both standard and custom fields.

CustomValue doesn’t support file-based operations and only supports CRUD-based calls. CustomValue is retrieved or deployed together with a GlobalValueSet or CustomObject component.

File Suffix and Directory Location

CustomValue components have the suffix .customValue. A CustomValue component is returned with either a GlobalValueSet or CustomObject component.

Version

CustomValue components are available in API version 38.0 and later. CustomValue replaces GlobalPicklistValue from API version 37.0.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>string</td>
<td>The color assigned to the picklist value when it’s used in charts on reports and dashboards. The color is in hexadecimal format; for example, #FF6600. If a color is not specified, it’s assigned dynamically upon chart generation.</td>
</tr>
<tr>
<td>default</td>
<td>boolean</td>
<td>Required. Indicates whether this value is the default selection for the global picklist and the custom picklists that share its picklist value set. This field is set to true by default.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A picklist value’s description. It’s useful to include a description for a picklist value so the reason for creating it can be tracked. Limit: 255 characters.</td>
</tr>
</tbody>
</table>
### DescriptionField TypeField Name

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Indicates whether this value is currently active or inactive. The default value is <code>true</code>. Users can select only active values from a picklist. An API retrieve operation for global picklist values returns all active and inactive values in the picklist. (Meanwhile, retrieving the values of a non-global, unrestricted picklist returns only the active values.)</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The value’s display label. If you don’t specify the label when creating a value it defaults to the API name. Available in API version 39.0 and later.</td>
</tr>
</tbody>
</table>

### StandardValue

This metadata type defines a value in a value set for a standard picklist and specifies whether this value is the default value. This type extends the CustomValue metadata type and inherits all its fields.

When you deploy changes to standard picklist fields, picklist values are added as needed.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowEmail</td>
<td>boolean</td>
<td>Indicates whether this value lets users email a quote PDF (<code>true</code>), or not (<code>false</code>). This field is only relevant for the <code>Status</code> field in quotes. This field is available in API version 18.0 and later.</td>
</tr>
<tr>
<td>closed</td>
<td>boolean</td>
<td>Indicates whether this value is associated with a closed status (<code>true</code>), or not (<code>false</code>). This field is only relevant for the standard <code>Status</code> field in cases and tasks. This field is available in API version 16.0 and up to version 36.0. In version 37.0, this field is in GlobalPicklistValue.</td>
</tr>
<tr>
<td>converted</td>
<td>boolean</td>
<td>Indicates whether this value is associated with a converted status (<code>true</code>), or not (<code>false</code>). This field is relevant for only the standard <code>Lead Status</code> field in leads. Your organization can set its own guidelines for determining when a lead is qualified, but typically, you want to convert a lead as soon as it becomes a real opportunity that you want to forecast. For more information, see “Convert Qualified Leads” in the Salesforce online help. This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>cssExposed</td>
<td>boolean</td>
<td>Indicates whether this value is available in your Self-Service Portal (<code>true</code>), or not (<code>false</code>). This field is only relevant for the standard <code>Case Reason</code> field in cases. Self-Service provides an online support channel for your customers - allowing them to resolve their inquiries without contacting a customer service representative. For more information about Self-Service, see “Setting Up Your Self-Service Portal” in the Salesforce online help. Note: Starting with Spring ’12, the Self-Service portal isn’t available for new orgs. Existing orgs continue to have access to the Self-Service portal. This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>forecastCategory</td>
<td>ForecastCategories (enumeration of type string)</td>
<td>Indicates whether this value is associated with a forecast category (true), or not (false). This field is only relevant for the standard Stage field in opportunities.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Omitted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Pipeline</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BestCase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Forecast</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Closed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>highPriority</td>
<td>boolean</td>
<td>Indicates whether this value is a high priority item (true), or not (false). This field is only relevant for the standard Priority field in tasks. For more information about tasks, see “Guidelines for Using Tasks” in the Salesforce online help. This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>probability</td>
<td>int</td>
<td>Indicates whether this value is a probability percentage (true), or not (false). This field is only relevant for the standard Stage field in opportunities. This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>reverseRole</td>
<td>string</td>
<td>A picklist value corresponding to a reverse role name for a partner. If the role is “subcontractor”, then the reverse role might be “general contractor”. Assigning a partner role to an account in Salesforce creates a reverse partner relationship so that both accounts list the other as a partner. This field is only relevant for partner roles. For more information, see “Partner Fields” in the Salesforce online help. This field is available in API version 18.0 and later.</td>
</tr>
<tr>
<td>reviewed</td>
<td>boolean</td>
<td>Indicates whether this value is associated with a reviewed status (true), or not (false). This field is only relevant for the standard Status field in solutions. For more information about opportunities, see “Creating Solutions” in the Salesforce online help. This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>won</td>
<td>boolean</td>
<td>Indicates whether this value is associated with a closed or won status (true), or not (false). This field is only relevant for the standard Stage field in opportunities. This field is available in API version 16.0 and later.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

For an example of CustomValue components within a GlobalValueSet component that’s referenced by a package.xml, see GlobalValueSet.
Dashboard

Represents a dashboard. Dashboards are visual representations of data that allow you to see key metrics and performance at a glance. This type extends the Metadata metadata type and inherits its fullName field. For more information, see “Edit Dashboards in Accessibility Mode in Salesforce Classic” in the Salesforce online help.

Declarative Metadata File Suffix and Directory Location

Dashboards are stored in the dashboards directory of the corresponding package directory. The file name matches the dashboard title and the extension is .dashboard.

Retrieving Dashboards

You can’t use the wildcard (*) symbol with dashboards in package.xml. To retrieve the list of dashboards for populating package.xml with explicit names, call listMetadata() and pass in DashboardFolder as the type. Note that DashboardFolder is not returned as a type in describeMetadata(). Dashboard is returned from describeMetadata() with an associated attribute of inFolder set to true. If that attribute is set to true, you can construct the type by using the component name with the word Folder, such as DashboardFolder.

The following example shows folders in package.xml. The names used in package.xml must be developer names, not dashboard titles.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>MyDBFolder/MyDBName</members>
    <name>Dashboard</name>
  </types>
  <types>
    <members>MyDocumentFolder/MyDocumentName</members>
    <name>Document</name>
  </types>
  <types>
    <members>unfiled$public/MarketingProductInquiryResponse</members>
    <members>unfiled$public/SalesNewCustomerEmail</members>
    <name>EmailTemplate</name>
  </types>
  <types>
    <members>MyReportFolder/MyReportName</members>
    <name>Report</name>
  </types>
  <version>46.0</version>
</Package>
```

Version

Dashboard components are available in API version 14.0 and later.
## Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>backgroundEndColor</code></td>
<td>string</td>
<td>Required. A dashboard can have a gradient color change on its charts. This field defines the second color for the gradient and <code>backgroundStartColor</code> defines the first color. If you prefer your background to be all one color or do not want a gradient color change, select the same color for this field and <code>backgroundStartColor</code>. The color is in hexadecimal format; for example #FF6600.</td>
</tr>
</tbody>
</table>
| `backgroundFadeDirection` | ChartBackgroundColor (enumeration of type string) | Required. The direction of the gradient color change, defined by the `backgroundStartColor` and `backgroundEndColor` fields. The valid values are:  
  - Diagonal  
  - LeftToRight  
  - TopToBottom |
| `backgroundStartColor` | string                      | Required. The starting color for the gradient color change on the dashboard's charts. See `backgroundEndColor` for more information. The color is in hexadecimal format; for example #FF6600. |
| `chartTheme`       | ChartTheme (enumeration of type string) | Determines the default theme for all dashboard charts. Replaces `dashboardChartTheme` for API v42.0 and later.  
  - light—Light-colored theme.  
  - dark—Dark-colored theme.  
  This field is available in API version 42.0 and later. |
| `colorPalette`     | ChartColorPalettes (enumeration of type string) | Determines the default palette for all dashboard charts. Replaces `dashboardColorPalette` for API v42.0 and later.  
  - accessible  
  - bluegrass  
  - colorSafe  
  - Default  
  - dusk  
  - earth  
  - fire  
  - gray  
  - heat  
  - justice  
  - nightfall  
  - pond  
  - sunrise |
### Metadata Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dashboardChartTheme</td>
<td>ChartTheme (enumeration of type string)</td>
<td>Determines the default theme for all dashboard charts. - <strong>light</strong>—Light-colored theme. - <strong>dark</strong>—Dark-colored theme. This field is available to maintain backward compatibility with versions prior to API version 42.0.</td>
</tr>
<tr>
<td>dashboardColorPalette</td>
<td>ChartColorPalettes (enumeration of type string)</td>
<td>Determines the default palette for all dashboard charts. - <strong>accessible</strong> - <strong>bluegrass</strong> - <strong>colorSafe</strong> - <strong>Default</strong> - <strong>dusk</strong> - <strong>earth</strong> - <strong>fire</strong> - <strong>gray</strong> - <strong>heat</strong> - <strong>justice</strong> - <strong>nightfall</strong> - <strong>pond</strong> - <strong>sunrise</strong> - <strong>tropic</strong> - <strong>unity</strong> - <strong>water</strong> - <strong>watermelon</strong> This field is available to maintain backward compatibility with versions prior to API version 42.0.</td>
</tr>
<tr>
<td>dashboardFilters</td>
<td>DashboardFilter[]</td>
<td>The list of filters in a dashboard. This field is available in API version 23.0 and later.</td>
</tr>
<tr>
<td>dashboardGridLayout</td>
<td>DashboardGridLayout</td>
<td>Lists the included DashboardGridComponent objects, specifies the number of dashboard columns, and sets each dashboard row’s height in pixels. This field is available in API version 35.0 and later.</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| dashboardType         | DashboardType (enumeration of type string) | Determines the way visibility settings are set for a dashboard. The valid values are:  
  • SpecifiedUser—All users see data at the access level of one specific running user, specified in the runningUser field, regardless of their own security settings.  
  • LoggedInUser—Each logged-in user sees data according to his or her own access level.  
  • MyTeamUser—Managers can choose to view the dashboard from the point of view of their subordinates in the role hierarchy. This value is available in API version 20.0 and later.  
  This field is available in API version 19.0 and later. |
| description           | string                   | Description for the dashboard. Maximum of 255 characters. |
| folderName            | string                   | Name of the folder that houses the dashboard.  
  This field is available in API version 35.0 and later. |
| fullName              | string                   | Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See `createMetadata()` to see an example of this field specified for a call.  
  This field specifies the folder and dashboard title; for example `folderSales/California`. |
| isGridLayout         | boolean                  | Specifies whether a dashboard uses the Lightning Experience layout (true) or not (false).  
  Lightning Experience allows dashboards with more than three columns with components that span multiple columns and multiple rows in size.  
  This field is available in API version 35.0 and later. |
| dashboardResultRefreshedDate | string             | Required. Date that the dashboard was last refreshed. |
| dashboardResultRunningUser | string          | Required. User currently accessing the dashboard. |
| leftSection           | DashboardComponentSection | Required. The left section or column of the dashboard. |
| middleSection         | DashboardComponentSection | The middle section or column of the dashboard. |
| numSubscriptions      | int                      | Number of subscriptions reported on the dashboard. This field is available in API version 42.0 and later. |
| rightSection          | DashboardComponentSection | Required. The right section or column of the dashboard. |
### Metadata Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>runningUser</td>
<td>string</td>
<td>The username of the user whose role and sharing settings are used to determine the data shown in the dashboard. When you deploy a dashboard and the value in this field is not defined or does not correspond to a valid user, the field is populated with the username of the user performing the deployment. Regardless of their security settings, all users viewing a dashboard see exactly the same data, because dashboards are always run using the security settings of a particular user. <strong>Tip:</strong> To avoid inappropriate exposure of sensitive data, save the dashboard to a folder that is visible only to appropriate users.</td>
</tr>
<tr>
<td>textColor</td>
<td>string</td>
<td>Required. Color of the text on each chart in the dashboard. The color is in hexadecimal format; for example #FF6600.</td>
</tr>
<tr>
<td>title</td>
<td>string</td>
<td>Required. The dashboard title.</td>
</tr>
<tr>
<td>titleColor</td>
<td>string</td>
<td>Required. Color of the titles on each dashboard component. The color is in hexadecimal format; for example #FF6600.</td>
</tr>
<tr>
<td>titleSize</td>
<td>int</td>
<td>Required. Size of characters in title text. For example, a value of 12 indicates 12pt text.</td>
</tr>
</tbody>
</table>

**DashboardFilter**

DashboardFilter represents a filter in a dashboard.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dashboardFilterOptions</td>
<td>DashboardFilterOption[]</td>
<td>The list of items you can select in the Filter Options section of the Add Filter dialog.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The filter label.</td>
</tr>
</tbody>
</table>

**DashboardFilterOption**

DashboardFilterOption represents a filter option in a dashboard.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operator</td>
<td>DashboardFilterOperation</td>
<td>Required. Represents the filter operation for this filter item. Valid values are:</td>
</tr>
</tbody>
</table>
|         | (enumeration of type string)   | • equals  
|         |                                 | • notEqual  
|         |                                 | • lessThan |
### Metadata Types

#### Dashboard

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>values</td>
<td>string[]</td>
<td>Required. One or more values in the Filter Options area of the Add Filter dialog. This field is available in API version 24.0 and later.</td>
</tr>
</tbody>
</table>

**Note:** The “between” operator takes two operands (for example, “between MinimumValue, MaximumValue”). Note also that the minimum value is inclusive, while the maximum value is exclusive. All other dashboard filter operations take a single operand only.

This field is available in API version 24.0 and later.

With API version 23.0, valid values are enumerated in FilterOperation.

#### DashboardGridLayout

Lightning Experience features dashboards with more than three columns and components that span multiple columns and multiple rows in size. DashboardGridLayout lists the included dashboard components, specifies the number of dashboard columns, and sets each dashboard row’s height in pixels.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dashboardGridComponents</td>
<td>DashboardGridComponent[]</td>
<td>List of DashboardGridComponent objects in the dashboard.</td>
</tr>
<tr>
<td>numberOfColumns</td>
<td>int</td>
<td>Required. Total number of columns in the dashboard.</td>
</tr>
<tr>
<td>rowHeight</td>
<td>int</td>
<td>Required. Height of each row in pixels.</td>
</tr>
</tbody>
</table>

#### DashboardGridComponent

Lightning Experience features dashboards with more than three columns and components that span multiple columns and multiple rows in size. DashboardGridComponent specifies location and size of a given dashboard component.
### Metadata Types

**Field** | **Field Type** | **Description**
--- | --- | ---
colSpan | int | Required. The width of the dashboard component in columns. For example, if `colSpan` is 5, then the dashboard component spans five columns.

columnIndex | int | Required. The left-most column that is occupied by the dashboard component.

dashboardComponent | DashboardComponent | Required. The dashboard component that is being sized and placed.

**rowIndex** | int | Required. The top-most row that is occupied by the dashboard component.

**rowSpan** | int | Required. The height of the dashboard component in rows.

---

**DashboardComponent**

A dashboard consists of a group of different components or elements that display data. Each component can use a custom report or a custom s-control as their data source to display corporate metrics or key performance indicators. You can create several dashboard components and display them all in one dashboard aligned in up to three columns.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| chartAxisRange | ChartRangeType (enumeration of type string)    | A manual or automatic axis range for bar or line charts. The valid values are: 
  - auto
  - manual

<table>
<thead>
<tr>
<th>chartAxisRangeMax</th>
<th>double</th>
<th>The maximum axis range to be displayed. This only applies to bar and line charts in which the manual axis range is selected for the <code>chartAxisRange</code> field.</th>
</tr>
</thead>
<tbody>
<tr>
<td>chartAxisRangeMin</td>
<td>double</td>
<td>The minimum axis range to be displayed. This only applies to bar and line charts in which the manual axis range is selected for the <code>chartAxisRange</code> field.</td>
</tr>
</tbody>
</table>

| chartSummary      | ChartSummaryc | Specifies the summary field for the chart data. Required if `isAutoSelectFromReport` is set to false. This field is available in API version 25.0 and later. |

| componentType     | DashboardComponentType (enumeration of type string) | Required. Dashboard component type. The valid values are: 
  - Bar
  - BarGrouped
  - BarStacked
  - BarStacked100 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dashboardFilterColumns</td>
<td>DashboardFilterColumn[]</td>
<td>A list of dashboard filter columns. Each report-based component must have a dashboard filter column that defines the column that the filter applies to. This field is available in API version 23.0 and later.</td>
</tr>
<tr>
<td>dashboardTableColumn</td>
<td>DashboardTableColumn[]</td>
<td>Represents a list of columns on a customized dashboard table component.</td>
</tr>
<tr>
<td>displayUnits</td>
<td>ChartUnits (enumeration of type string)</td>
<td>Chart Units. The valid values are:</td>
</tr>
</tbody>
</table>

- Column
- ColumnGrouped
- ColumnLine
- ColumnLineGrouped
- ColumnLineStacked
- ColumnLineStacked100
- ColumnStacked
- ColumnStacked100
- Donut
- FlexTable
- Funnel
- Gauge
- Line
- lineCumulative
- LineGrouped
- lineGroupedCumulative
- Metric
- Pie
- Scatter
- ScatterGrouped
- Scontrol
- Table

- Auto
- Integer
- Hundreds
- Thousands
- Millions
- Billions
- Trillions
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>drillDownUrl</td>
<td>string</td>
<td>For charts, specifies a URL that users go to when they click the dashboard component. Use this option to send users to another dashboard, report, record detail page, or other system that uses a Web interface. This field overrides the <code>drillEnabled</code> and <code>drillToDetailEnabled</code> fields.</td>
</tr>
<tr>
<td>drillEnabled</td>
<td>boolean</td>
<td>Specifies whether to take users to the full or filtered source report when they click the dashboard component. Set to <code>false</code> to drill to the full source report; set to <code>true</code> to drill to the source report filtered by what they clicked. If set to <code>true</code>, users can click individual groups, axis values, or legend entries. This overrides the <code>drillToDetailEnabled</code> field. This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>drillToDetailEnabled</td>
<td>boolean</td>
<td>When enabled, users are taken to the record detail page when they click a record name, record owner, or feed post in a table or chart. When set to <code>true</code> users can click axis and legend values, chart elements, and table entries. The <code>drillDownUrl</code> and <code>drillEnabled</code> fields override this field. This field is available in API version 20.0 and later.</td>
</tr>
<tr>
<td>enableHover</td>
<td>boolean</td>
<td>Specifies whether to display values, labels, and percentages when hovering over charts. Hover details depend on chart type. Percentages apply to pie, donut, and funnel charts only. This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>expandOthers</td>
<td>boolean</td>
<td>Specifies whether to combine all groups less than or equal to 3% of the total into a single ‘Others’ wedge or segment. This only applies to pie, donut, and funnel charts. Set to <code>true</code> to show all values individually on the chart; set to <code>false</code> to combine small groups into ‘Others.’ This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>flexComponentProperties</td>
<td>DashboardFlexTableComponentProperties</td>
<td>Defines metadata for Lightning Experience table columns and sorting. This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>footer</td>
<td>string</td>
<td>Footer displayed at the bottom of the dashboard component. Maximum of 255 characters.</td>
</tr>
<tr>
<td>gaugeMax</td>
<td>double</td>
<td>The maximum value on a gauge. A gauge is used to see how far you are from reaching a goal. It looks like a speedometer in a car.</td>
</tr>
<tr>
<td>gaugeMin</td>
<td>double</td>
<td>The minimum value on a gauge.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>groupingColumn</td>
<td>string</td>
<td>Specifies the field by which to group data. This data is displayed on the X-axis for vertical column charts and on the Y-axis for horizontal bar charts. This field is available in API version 25.0 and later.</td>
</tr>
<tr>
<td>header</td>
<td>string</td>
<td>Header displayed at the top of the dashboard component. Maximum of 80 characters.</td>
</tr>
<tr>
<td>indicatorBreakpoint1</td>
<td>double</td>
<td>The value that separates the indicatorLowColor from the indicatorMiddleColor on the dashboard.</td>
</tr>
<tr>
<td>indicatorBreakpoint2</td>
<td>double</td>
<td>The value that separates the indicatorMiddleColor from the indicatorHighColor on the dashboard.</td>
</tr>
<tr>
<td>indicatorHighColor</td>
<td>string</td>
<td>The color representing a high number range on the gauge.</td>
</tr>
<tr>
<td>indicatorLowColor</td>
<td>string</td>
<td>The color representing a low number range on the gauge.</td>
</tr>
<tr>
<td>indicatorMiddleColor</td>
<td>string</td>
<td>The color representing a medium number range on the gauge.</td>
</tr>
<tr>
<td>legendPosition</td>
<td>ChartLegendPosition (enumeration of type string)</td>
<td>The location of the legend with respect to the chart. The valid values are: Bottom, OnChart, Right</td>
</tr>
<tr>
<td>maxValuesDisplayed</td>
<td>int</td>
<td>The maximum number of elements to include in the top-level grouping of the horizontal axis of a horizontal chart, vertical axis of a vertical chart, or selected axis of a stacked bar chart. For example, if you want to list only your top five salespeople, create an opportunity report that lists total opportunity amounts by owner and enter 5 in this field.</td>
</tr>
<tr>
<td>metricLabel</td>
<td>string</td>
<td>Descriptive label for the metric. This is relevant if metric is the value of the componentType field.</td>
</tr>
<tr>
<td>page</td>
<td>string</td>
<td>Visualforce page associated with the component.</td>
</tr>
<tr>
<td>pageHeightInPixels</td>
<td>int</td>
<td>Display height of the Visualforce page in pixels.</td>
</tr>
<tr>
<td>report</td>
<td>string</td>
<td>Name of the report associated with the component.</td>
</tr>
<tr>
<td>scontrol</td>
<td>string</td>
<td>S-control associated with component if scontrol is the value of the componentType field. For more information, see “Defining Custom S-Controls” in the Salesforce online help.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>scontrolHeightInPixels</td>
<td>int</td>
<td>Display height of the s-control in pixels.</td>
</tr>
<tr>
<td>showPercentage</td>
<td>boolean</td>
<td>Indicates if percentages are displayed for regions of gauges and wedges and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>segments of pie, donut, and funnel charts (true), or not (false).</td>
</tr>
<tr>
<td>showPicturesOnCharts</td>
<td>boolean</td>
<td>Display Chatter photos for up to 20 records in a horizontal bar chart</td>
</tr>
<tr>
<td></td>
<td></td>
<td>component whose source report is grouped by a user or group name field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If there are more than 20 records with photos, record names are shown instead</td>
</tr>
<tr>
<td></td>
<td></td>
<td>of photos. Set Grouping Display to None to show photos. Set the Drill Down</td>
</tr>
<tr>
<td></td>
<td></td>
<td>to option to Record Detail Page to take users directly to user profile or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>group pages when they click photos. Chatter must be enabled for photos to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>be displayed. Depending on your organization's setup, you may not see</td>
</tr>
<tr>
<td></td>
<td></td>
<td>photos on tables and charts.</td>
</tr>
<tr>
<td>showPicturesOnTables</td>
<td>boolean</td>
<td>Display Chatter photos for up to 20 records in a horizontal bar chart</td>
</tr>
<tr>
<td></td>
<td></td>
<td>component whose source report is grouped by a user or group name field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If there are more than 20 records with photos, record names are shown instead</td>
</tr>
<tr>
<td></td>
<td></td>
<td>of photos. Set Grouping Display to None to show photos. Set the Drill Down</td>
</tr>
<tr>
<td></td>
<td></td>
<td>to option to Record Detail Page to take users directly to user profile or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>group pages when they click photos. Chatter must be enabled for photos to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>be displayed. Depending on your organization's setup, you may not see</td>
</tr>
<tr>
<td></td>
<td></td>
<td>photos on tables and charts.</td>
</tr>
<tr>
<td>showTotal</td>
<td>boolean</td>
<td>Indicates if the total of all wedges is displayed for gauges and donut</td>
</tr>
<tr>
<td></td>
<td></td>
<td>charts (true), or not (false).</td>
</tr>
<tr>
<td>showValues</td>
<td>boolean</td>
<td>Indicates if the values of individual records or groups are displayed for</td>
</tr>
<tr>
<td></td>
<td></td>
<td>charts (true), or not (false).</td>
</tr>
<tr>
<td>sortBy</td>
<td>DashboardComponentFilter (enumeration of type string)</td>
<td>The sort option for the dashboard component.</td>
</tr>
<tr>
<td>title</td>
<td>string</td>
<td>The title of the dashboard component. Maximum of 40 characters.</td>
</tr>
<tr>
<td>useReportChart</td>
<td>boolean</td>
<td>Specifies whether to use the chart defined in the source report on this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dashboard component. The chart settings in the source report determine how</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the chart displays in the dashboard, and any chart settings you define for</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the dashboard are overridden. If you defined a combination chart in the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>source report, use this option to use that combination chart on this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dashboard.</td>
</tr>
</tbody>
</table>
DashboardFilterColumn

DashboardFilterColumn represents a filter column in a dashboard.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>column</td>
<td>string</td>
<td>Required. The report column code for the filter.</td>
</tr>
</tbody>
</table>

DashboardTableColumn

DashboardTableColumn represents a column in a customized table component in a dashboard.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregateType</td>
<td>ReportSummaryType[] (enumeration of type string)</td>
<td>Specifies the aggregation type for the table column.</td>
</tr>
<tr>
<td>column</td>
<td>string</td>
<td>Required. The label of the column to use in the table.</td>
</tr>
<tr>
<td>showTotal</td>
<td>boolean</td>
<td>Displays the totals for each summarizable column in the dashboard table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 19.0 and later.</td>
</tr>
<tr>
<td>sortBy</td>
<td>DashboardComponentFilter (enumeration of type string)</td>
<td>The sort option for the dashboard table component. Sort on just one column per table.</td>
</tr>
</tbody>
</table>

DashboardFlexTableComponentProperties

DashboardFlexTableComponentProperties represents a column in a customized table component in a dashboard.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flexTableColumn</td>
<td>DashboardComponentColumn</td>
<td>Represents a column in a Lightning Experience table component.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>flexTableSortInfo</td>
<td>DashboardComponentSortInfo</td>
<td>Represents sorting column and order in a Lightning Experience table component.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>hideChatterPhotos</td>
<td>boolean</td>
<td>If true, hides any photos from Chatter feeds.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>decimalPrecision</td>
<td>integer</td>
<td>For columns with numeric values, indicates the number of significant digits.</td>
</tr>
</tbody>
</table>

DashboardComponentColumn

DashboardComponentColumn represents a component column in a dashboard. Available in API version 41.0 and later.
## DashboardMetadata Types

### DashboardFilterColumns
DashboardFilterColumns represents a filter column in a dashboard.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>breakPoint1</td>
<td>double</td>
<td>The value that separates the lowRangeColor from the midRangeColor on the dashboard.</td>
</tr>
<tr>
<td>breakPoint2</td>
<td>double</td>
<td>The value that separates the midRangeColor from the highRangeColor on the dashboard.</td>
</tr>
<tr>
<td>breakPointOrder</td>
<td>double</td>
<td>Conditional highlighting can be applied to multiple columns. This field stores the order of conditional highlights.</td>
</tr>
<tr>
<td>highRangeColor</td>
<td>int</td>
<td>The color representing a high number range on the column.</td>
</tr>
<tr>
<td>lowRangeColor</td>
<td>int</td>
<td>The color representing a low number range on the column.</td>
</tr>
<tr>
<td>midRangeColor</td>
<td>int</td>
<td>The color representing a mid number range on the column.</td>
</tr>
<tr>
<td>reportColumn</td>
<td>string</td>
<td>Required. The report column code for the filter.</td>
</tr>
<tr>
<td>showTotal</td>
<td>boolean</td>
<td>If true, the column total is displayed.</td>
</tr>
<tr>
<td>type</td>
<td>DashboardComponentColumnType</td>
<td>Represents the type of Lightning Experience table column:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>- Details</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Aggregates</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Grouping</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 41.0 and later.</td>
</tr>
</tbody>
</table>

### DashboardComponentSortInfo
DashboardComponentSortInfo represents a filter column in a dashboard.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sortColumn</td>
<td>string</td>
<td>Indicates the column on which the table is sorted. This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>sortOrder</td>
<td>string</td>
<td>Indicates whether column sorting is ascending or descending. This field is available in API version 41.0 and later.</td>
</tr>
</tbody>
</table>

### DashboardComponentSection
DashboardComponentSection represents one of the sections or columns in a dashboard.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnSize</td>
<td>DashboardComponentSize</td>
<td>Required. The size of the column in the dashboard:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>- Medium</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Narrow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Wide</td>
</tr>
</tbody>
</table>
**DashboardComponentFilter**

DashboardComponentFilter is an enumeration of type string that lists the sort values for dashboard components. The valid values are:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RowLabelAscending</td>
<td>Sorts in alphabetical order by the label.</td>
</tr>
<tr>
<td>RowLabelDescending</td>
<td>Sorts in reverse alphabetical order by the label.</td>
</tr>
<tr>
<td>RowValueAscending</td>
<td>Sorts lowest to highest by the value.</td>
</tr>
<tr>
<td>RowValueDescending</td>
<td>Sorts highest to lowest by the value.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition — Filtered Dashboard**

A sample XML definition of a filtered dashboard is shown below. Note that this example is supported in API version 24.0 and later. The file name matches the dashboard title and the extension is `.dashboard`.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Dashboard xmlns="http://soap.sforce.com/2006/04/metadata">
    <backgroundEndColor>#FFFFFF</backgroundEndColor>
    <backgroundFadeDirection>Diagonal</backgroundFadeDirection>
    <backgroundStartColor>#FFFFFF</backgroundStartColor>
    <dashboardFilters>
        <dashboardFilterOptions>
            <operator>equals</operator>
            <values>Media</values>
        </dashboardFilterOptions>
        <dashboardFilterOptions>
            <operator>lessThan</operator>
            <values>Working</values>
        </dashboardFilterOptions>
        <dashboardFilterOptions>
            <operator>between</operator>
            <values>ABC</values>
            <values>XYZ</values>
        </dashboardFilterOptions>
        <name>Industry</name>
    </dashboardFilters>
    <dashboardFilters>
        <dashboardFilterOptions>
            <operator>equals</operator>
            <values>Analyst,Partner</values>
        </dashboardFilterOptions>
        <dashboardFilterOptions>
            <operator>startsWith</operator>
            <values>3</values>
        </dashboardFilterOptions>
    </dashboardFilters>
</Dashboard>
```
<values>Integrator</values>
</dashboardFilterOptions>
<name>Account Type</name>
</dashboardFilters>
<dashboardType>SpecifiedUser</dashboardType>
</leftSection>
<columnSize>Medium</columnSize>
<components>
  <chartAxisRange>Auto</chartAxisRange>
  <componentType>Bar</componentType>
  <dashboardFilterColumns>
    <column>INDUSTRY</column>
  </dashboardFilterColumns>
  <dashboardFilterColumns>
    <column>TYPE</column>
  </dashboardFilterColumns>
  <displayUnits>Auto</displayUnits>
  <drillEnabled>false</drillEnabled>
  <drillToDetailEnabled>false</drillToDetailEnabled>
  <enableHover>false</enableHover>
  <expandOthers>false</expandOthers>
  <legendPosition>Bottom</legendPosition>
  <report>unfiled$public/SampleReportofAccounts</report>
  <showPercentage>false</showPercentage>
  <showPicturesOnCharts>false</showPicturesOnCharts>
  <showValues>true</showValues>
  <sortBy>RowLabelAscending</sortBy>
  <useReportChart>false</useReportChart>
</components>
</middleSection>
<columnSize>Medium</columnSize>
<components>
  <chartAxisRange>Auto</chartAxisRange>
  <componentType>Funnel</componentType>
  <dashboardFilterColumns>
    <column>ACCOUNT_INDUSTRY</column>
  </dashboardFilterColumns>
  <dashboardFilterColumns>
    <column>ACCOUNT.TYPE</column>
  </dashboardFilterColumns>
  <displayUnits>Auto</displayUnits>
  <drillEnabled>false</drillEnabled>
  <drillToDetailEnabled>false</drillToDetailEnabled>
  <enableHover>false</enableHover>
  <expandOthers>false</expandOthers>
  <legendPosition>Bottom</legendPosition>
  <report>unfiled$public/SampleReportofCases</report>
  <showPercentage>false</showPercentage>
  <showValues>true</showValues>
  <sortBy>RowLabelAscending</sortBy>
  <useReportChart>false</useReportChart>
</components>
Declarative Metadata Sample Definition — Unfiltered Dashboard

A sample XML definition of a dashboard is shown below. The file name matches the dashboard title and the extension is .dashboard.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Dashboard xmlns="http://soap.sforce.com/2006/04/metadata">
  <backgroundEndColor>#FFFFFF</backgroundEndColor>
  <backgroundFadeDirection>LeftToRight</backgroundFadeDirection>
  <backgroundStartColor>#FFFFFF</backgroundStartColor>
  <description>Dashboard with all possible chart types</description>
  <leftSection>
    <columnSize>Medium</columnSize>
    <components>
      <chartAxisRange>Auto</chartAxisRange>
      <componentType>BarStacked100</componentType>
      <displayUnits>Auto</displayUnits>
      <drillEnabled>true</drillEnabled>
      <enableHover>true</enableHover>
      <report>testFolder/sourceRep</report>
      <sortBy>RowLabelAscending</sortBy>
    </components>
  </leftSection>
</Dashboard>
```
DashboardMetadata Types
Declarative Metadata Sample Definition — Lightning Experience Dashboard with \texttt{isGridLayout} Equals \texttt{true}

A sample XML definition of a Lightning Experience dashboard with \texttt{isGridLayout} equals \texttt{true} is shown below. Note that this example is supported in API version 35.0 and later. The file name matches the dashboard title and the extension is \texttt{.dashboard}.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Dashboard xmlns="http://soap.sforce.com/2006/04/metadata">
  <dashboardType>SpecifiedUser</dashboardType>
  <gridLayout>
    <dashboardGridComponents>
      <colSpan>3</colSpan>
      <columnIndex>0</columnIndex>
      <dashboardComponent>
        <autoselectColumnsFromReport>false</autoselectColumnsFromReport>
        <chartAxisRange>Auto</chartAxisRange>
        <chartSummary>
          <axisBinding>y</axisBinding>
          <column>RowCount</column>
        </chartSummary>
        <componentType>Donut</componentType>
        <drillEnabled>false</drillEnabled>
        <drillToDetailEnabled>false</drillToDetailEnabled>
        <enableHover>false</enableHover>
        <expandOthers>false</expandOthers>
        <groupingColumn>TITLE</groupingColumn>
        <legendPosition>Bottom</legendPosition>
        <report>unfiled$public/lead_rpt</report>
        <showPercentage>false</showPercentage>
        <showTotal>false</showTotal>
        <showValues>true</showValues>
        <sortBy>RowLabelAscending</sortBy>
        <useReportChart>false</useReportChart>
      </dashboardComponent>
      <rowIndex>0</rowIndex>
      <rowSpan>3</rowSpan>
    </dashboardGridComponents>
    <dashboardGridComponents>
      <colSpan>3</colSpan>
      <columnIndex>0</columnIndex>
      <dashboardComponent>
        <autoselectColumnsFromReport>false</autoselectColumnsFromReport>
        <chartAxisRange>Auto</chartAxisRange>
      </dashboardComponent>
    </dashboardGridComponents>
  </gridLayout>
</Dashboard>
```
Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
Folder  
Report

DataCategoryGroup

Represents a data category group.

This type extends the Metadata metadata type and inherits its fullName field.

⚠️ Warning: Using Metadata API to deploy category changes from one organization to another permanently removes categories and record categorizations that are not specified in your XML file. Salesforce recommends that you manually create data categories and record associations in an organization from Setup by entering Data Categories in the Quick Find box, then selecting Data Categories rather than deploying changes from a sandbox to a production organization. For more information, see Usage.

Data category groups are provided to:

- Classify and filter data.
- Share data among users.

Every data category group contains items or data categories that can be organized hierarchically.

The example below shows the Geography data category group and its data categories.

<table>
<thead>
<tr>
<th>Geography</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worldwide</td>
</tr>
<tr>
<td>North America</td>
</tr>
<tr>
<td>United States of America</td>
</tr>
<tr>
<td>Canada</td>
</tr>
<tr>
<td>Mexico</td>
</tr>
<tr>
<td>Europe</td>
</tr>
<tr>
<td>Asia</td>
</tr>
</tbody>
</table>

⚠️ Note: See "Work with Data Categories" in the Salesforce online help for more information on data category groups, data categories, parent and sub categories.

File Suffix and Directory Location

The file suffix is .datacategorygroup. There is one file for each data category group stored in the datacategorygroups folder in the corresponding package directory.
Version
Data category groups are available in API version 18.0 and later.

Fields
This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. The status of the category group. Indicates whether this category group is active, (true), or not active (false).</td>
</tr>
<tr>
<td>dataCategory</td>
<td>DataCategory</td>
<td>Required. The top-level category within the data category group.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the data category group.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The unique name of the data category group. When creating a data category group, the fullName field and the file name (without its suffix) must match. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the Metadata component.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Label that represents the object in Salesforce.</td>
</tr>
<tr>
<td>objectUsage</td>
<td>ObjectUsage</td>
<td>The objects that are associated with the data category group.</td>
</tr>
</tbody>
</table>

DataCategory
Represents an item (or data category) in the data category group. A data category can recursively contain a list of other data categories.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataCategory</td>
<td>DataCategory[]</td>
<td>A recursive list of sub data categories. For example, a list of countries within a continent. You can create up to 100 categories in a data category group and have up to 5 levels in a data category group hierarchy.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Label for the data category throughout the Salesforce user interface.</td>
</tr>
</tbody>
</table>
| name       | string              | Required. The developer name of the data category used as a unique identifier for API access. The name can only contain characters, letters, and the underscore (_) character, must start with a letter, and cannot end with an underscore or contain two consecutive underscore characters.  

⚠️ Important: The value for this field is defined once and cannot be changed later.

⚠️ Warning: If you deploy a category group that already exists in an organization, any category that is not defined in the XML file
is permanently removed from your organization. For more information see Usage.

### ObjectUsage

Represents the objects that can be associated with the data category group. This association allows the object to be classified and filtered using the data categories.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>string[]</td>
<td>A list of the object names that can be associated with the data category group. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• KnowledgeArticleVersion—to associate articles. See &quot;Modify Default Category Group Assignments for Articles&quot; in the Salesforce online help for more information on data category groups association to articles.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Question—to associate questions. You can associate the Question object with at most one category group. See &quot;Assigning Data Categories to Answers&quot; in the Salesforce online help for more information on data category groups association to questions.</td>
</tr>
</tbody>
</table>
|            |            | **Warning:** If you deploy a category group that already exists in an organization, any object association that is not defined in the XML file is permanently removed from your organization. Ensure that your XML file specifies all the records associated with your category group in the organization. For more information see Usage.

### Declarative Metadata Sample Definition

This sample is the definition of the Geography data category group and its data categories:

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<DataCategoryGroup xmlns="http://soap.sforce.com/2006/04/metadata">
  <label>Geography</label>
  <description>Geography structure of service center locations</description>
  <fullName>geo</fullName>

  <dataCategory> <name>WW</name> <label>Worldwide</label> </dataCategory>
  <dataCategory> <name>AMER</name> <label>North America</label> <dataCategory>
    <name>USA</name> <label>United States of America</label>
  </dataCategory>
</DataCategoryGroup>
```
Usage

When you deploy a category group XML file, Metadata API checks whether the category group exists in the target organization. If the category group does not exist, it is created. If the category group already exists, then Metadata API:

- Adds any new category or object defined in the XML file.
- Deletes any category that is not defined in the XML file. Records associated with the deleted categories are re-associated with the parent category.
- Deletes any object association that is not defined in the XML file.
- Moves any category if its hierarchical position differs from the position specified in the XML file.

Note: When a category moves to a new parent category, users that have no visibility on the new parent category lose their visibility to the repositioned category.

Note: For more information about category deletion, category repositioning and its impact on record categorization and visibility see “Delete a Data Category” and “Modify and Arrange Data Categories” in the Salesforce online help.

Using Metadata API to deploy category changes from one organization to another permanently removes categories and record categorizations that are not specified in your XML file. Salesforce recommends that you manually create data categories and record associations in an organization from Setup by entering Data Categories in the Quick Find box, then selecting Data Categories rather than deploying changes from a sandbox to a production organization.
The following example illustrates what happens if you deploy an XML representation of a Geography data category group hierarchy to an organization that already has this data category group defined. Note that the organization contains a US category, while the XML file includes a USA category in the same hierarchical position. The Metadata API deployment process deletes the US category from the organization and moves associations for any records from US to the parent AMER category. It also adds the USA category under AMER. Note that all records that were previously categorized with US are now associated with the AMER category.

The next example illustrates what can happen when you delete or move a category in a data category group and deploy its XML representation from a sandbox to a production organization that already has this data category group defined. Hierarchy 1 shows the initial data category group in the sandbox organization. In hierarchy 2, we add an EU category under EMEA and move FR, SP and UK below EU. In hierarchy 3, we delete FR and associate its records with its new parent, EU. Finally, we deploy the changes from the sandbox to the production organization.
Metadata API has no concept of the order of the changes made to the sandbox organization. It just deploys the changes from one organization to another. During the deployment, it first notices the deletion of the FR category and removes it from the production organization. Consequently, it moves associations for any records from FR to its parent on the production organization, EMEA. Metadata API then adds the EU category and moves SP and UK below it. Although the category group hierarchy looks the same in both organizations, record categorization in production is different from the sandbox organization. The records that were originally associated with FR in hierarchy 1 are associated with EU in the sandbox organization, but are associated with EMEA in the production organization.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

DelegateGroup

Represents a group of users who have the same administrative privileges. These groups are different from public groups used for sharing. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

DelegateGroup components have the suffix .delegateGroup and are stored in the delegateGroups folder. The file prefix must match the developer name of the delegate group. For example, a delegate group with a developer name of MyDelegateGroup would have a file name of MyDelegateGroup.delegateGroup.

Version

DelegateGroup components are available in API version 36.0 and later.

Special Access Rules

Only users with the “View Setup and Configuration” permission can be delegated administrators.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customObjects</td>
<td>string[]</td>
<td>The custom objects associated with the group. Delegated administrators can customize nearly every aspect of each of those custom objects, including creating a custom tab. However, they cannot create or modify relationships on the objects or set organization-wide sharing defaults. Delegated administrators must have access to custom objects to access the merge fields on those objects from formulas.</td>
</tr>
<tr>
<td>groups</td>
<td>string[]</td>
<td>The groups with users assigned by delegated administrators.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The delegated group's non-API name.</td>
</tr>
<tr>
<td>loginAccess</td>
<td>boolean</td>
<td>Required. Allows users in this group to log in as users in the role hierarchy that they administer (true) or not (false). Depending on your organization settings, individual users must grant login access to allow their administrators to log in as them.</td>
</tr>
<tr>
<td>permissionSets</td>
<td>string[]</td>
<td>The permission sets assignable to users in specified roles and all subordinate roles by delegated administrators.</td>
</tr>
<tr>
<td>profiles</td>
<td>string[]</td>
<td>The profiles assignable to users by delegated administrators.</td>
</tr>
<tr>
<td>roles</td>
<td>string[]</td>
<td>The roles and subordinates for which delegated administrators of the group can create and edit users.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a DelegateGroup component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<DelegateGroup xmlns="http://soap.sforce.com/2006/04/metadata"
    <label>MyDelegateGroup</label>
    <loginAccess>true</loginAccess>
    <name>MyDelegateGroup</name>
    <profiles>Chatter Free User</profiles>
    <profiles>Chatter Moderator User</profiles>
    <profiles>Marketing User</profiles>
    <permissionSets>My Permset</permissionSets>
    <roles>LesserBossMan</roles>
</DelegateGroup>
```

The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>*</members>
        <name>DelegateGroup</name>
    </types>
    <version>46.0</version>
</Package>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Document

Represents a Document. All documents must be in a document folder, such as sampleFolder/TestDocument.

This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.

Currently, users are not able to export document metadata to a local file system using the Force.com IDE.

Retrieving Documents

You can't use the wildcard (*) symbol with documents in package.xml. To retrieve the list of documents for populating package.xml with explicit names, call listMetadata() and pass in DocumentFolder as the type. Note that DocumentFolder is not returned as a type in describeMetadata(). Document is returned from describeMetadata() with an associated attribute of inFolder set to true. If that attribute is set to true, you can construct the type by using the component name with the word Folder, such as DocumentFolder.

The following example shows folders in package.xml:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>MyDBFolder/MyDBName</members>
    <name>Dashboard</name>
  </types>
  <types>
    <members>MyDocumentFolder/MyDocumentName</members>
    <name>Document</name>
  </types>
  <types>
    <members>unfiled$public/MarketingProductInquiryResponse</members>
    <members>unfiled$public/SalesNewCustomerEmail</members>
    <name>EmailTemplate</name>
  </types>
  <types>
    <members>MyReportFolder/MyReportName</members>
    <name>Report</name>
  </types>
  <version>46.0</version>
</Package>
```

For each document an accompanying metadata file named DocumentFilename-meta.xml is created in the document folder. For example, for a document TestDocument.png in the sampleFolder folder, there's a TestDocument.png-meta.xml in the documents/sampleFolder of the package.

Version

Documents are available in API version 10.0 and later.
In API version 17.0 and later, you can delete a folder containing documents moved to the Recycle Bin. When you delete the folder, any related documents in the Recycle Bin are permanently deleted.

In API version 18.0 and later, documents do not need an extension.

**Fields**

This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
<td>base64</td>
<td>Content of the document. Base 64-encoded binary data. Prior to making an API call, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to binary. This conversion is usually handled for you by a SOAP client. This field is inherited from the MetadataWithContent component.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the document. Enter a description to distinguish this document from others.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The name of the document, including the folder name. In version 17.0 and earlier, the fullName included the document extension. In version 18.0 and later, the fullName does not include the file extension. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. If this field contained characters before version 14.0 that are no longer allowed, the characters were stripped out of this field, and the previous value of the field was saved in the name field. This field is inherited from the Metadata component.</td>
</tr>
<tr>
<td>internalUseOnly</td>
<td>boolean</td>
<td>Required. Indicates whether the document is confidential (true) or not (false). This field and public are mutually exclusive; you cannot set both to true.</td>
</tr>
<tr>
<td>keywords</td>
<td>string</td>
<td>Contains one or more words that describe the document. A check for matches to words in this field is performed when doing a search.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The list of characters allowed in the fullName field has been reduced for versions 14.0 and later. This field contains the value contained in the fullName field before version 14.0. This field is only populated if the value of the fullName field contained characters that are no longer accepted in that field.</td>
</tr>
<tr>
<td>public</td>
<td>boolean</td>
<td>Required. Indicates whether the document is an image available for HTML email templates and does not require a Salesforce username and password to view in an email (true) or not (false). If the images will be used as a custom app logo or custom tab icon, both of which require a Salesforce username and password to view, set this field to false. This field and internalUseOnly are mutually exclusive; you cannot set both to true.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is the definition of a document:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Document xmlns="http://soap.sforce.com/2006/04/metadata">
  <internalUseOnly>false</internalUseOnly>
  <name>Q2 Campaign Analysis</name>
  <public>false</public>
  <description>Analyze Q2 campaign effectiveness</description>
</Document>
```

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- Folder

DuplicateRule

Represents a rule that specifies how duplicate records in an object are detected. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

DuplicateRule components have the .duplicateRule suffix and are stored in the duplicateRules/ directory. The name of the component file is based on the name of the object associated with the rule. For example, the component file name duplicateRules/Account.Standard_Account_Duplicate_Rule.duplicateRule describes a duplicate rule component associated with the Account object.

Version

DuplicateRule components are available in API version 46.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionOnInsert</td>
<td>DupeActionType (enumeration)</td>
<td>Required. Determines what the duplicate rule does when users or the DuplicateRule API try to insert a record that is a duplicate. Valid values are: Allow For users, if operationsOnInsert is set to alert, the UI displays the value of alertText in a dialog. The dialog prompts</td>
</tr>
</tbody>
</table>
users to continue or cancel. If the user chooses to continue, the insertion proceeds. If the user chooses to cancel, the record isn’t inserted.

The DuplicateRule API returns an error code and a message. To complete the insertion, the code must set the `allowSave` field in `DuplicateRuleHeader` to `true` and reissue the request.

If `operationsOnInsert` isn’t set to `alert`, the UI inserts the record without issuing an alert. The API inserts the record and doesn’t return an error code.

**Block**

For users, the UI displays an error message and prevents them from inserting the new record. The DuplicateRule API returns an error and doesn’t insert the record.

---

**actionOnUpdate**

**DupeActionType (enumeration of type string)**

Required. Determines what the duplicate rule does when users or the DuplicateRule API try to update a record, and the result is a duplicate. Valid values are:

- **Allow**
  - For users, if `operationsOnUpdate` is set to `alert`, the UI displays the value of `alertText` in a dialog. The dialog prompts users to continue or cancel. If the user chooses to continue, the update proceeds. If the user chooses to cancel, the record isn’t updated.
  - The DuplicateRule API returns a message. To complete the update, the code must set the `allowSave` field in `DuplicateRuleHeader` to `true` and reissue the request.
  - If `operationsOnUpdate` isn’t set to `alert`, the UI updates the record without issuing an alert. The API updates the record and doesn’t return an error code.

- **Block**
  - For users, the UI displays an error message and prevents them from continuing. The DuplicateRule API returns an error.

---

**alertText**

**string**

Text that’s sent when the duplicate rule is triggered. The text is only sent if `isActive` is `true`. In the UI, the text displays as a message. The DuplicateRule API returns the message in its response.

You can set a value for `alertText` only when you have `actionOnInsert` or `actionOnUpdate` (or both) set to `Allow`. Otherwise, you receive a validation error when you add or update this component.

---

**description**

**string**

Required. Text that describes the duplicate rule. The value is customer-supplied, but is not visible in the UI.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>duplicateRuleFilter</td>
<td>DuplicateRuleFilter</td>
<td>Required. Criteria that define how to find records to consider when looking for duplicates. For example, use <code>duplicateRuleFilter</code> to exclude records from the match when looking for duplicates.</td>
</tr>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Required. If <code>true</code>, the <code>DuplicateRule</code> detects duplicate records. Otherwise, the rule has no effect.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Master label for this <code>DuplicateRule</code>. This value is the internal label for the rule.</td>
</tr>
<tr>
<td>operationsOnInsert</td>
<td>string[]</td>
<td>Required. Controls the action to take when <code>actionOnInsert</code> is set to <code>Allow</code> and the duplicate rule is triggered. Either one or both of these values can be set in the array:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>alert</strong> If set, the action specified in <code>actionOnInsert</code> occurs; otherwise, the insert proceeds.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>report</strong> If set, the insert operation is added to the report of duplicates.</td>
</tr>
<tr>
<td>operationsOnUpdate</td>
<td>string[]</td>
<td>Required. Controls the action to take when <code>actionOnUpdate</code> is set to <code>Allow</code> and the duplicate rule is triggered. Either one or both of these values can be set in the array:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>alert</strong> If set, the action specified in <code>actionOnUpdate</code> occurs; otherwise, the update proceeds.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>report</strong> If set, the update operation is added to the report of duplicates.</td>
</tr>
<tr>
<td>securityOption</td>
<td>DupeSecurityOptionType (enumeration of type string)</td>
<td>Required. Determines how record sharing rules affect duplicate management. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>EnforceSharingRules</strong> Sharing rules affect duplicate management. If a duplicate rule is triggered because an insert or update duplicates an existing record, but the running user doesn’t have sharing access to that record, the insert or update proceeds. The sharing rule doesn’t prevent the user from creating or updating the record because the record is hidden from the user. No message is issued.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>BypassSharingRules</strong> Sharing rules don’t affect duplicate management. If a duplicate rule is triggered because an insert or update duplicates an existing record, sharing rules are ignored, but other access restrictions apply.</td>
</tr>
<tr>
<td>sortOrder</td>
<td>int</td>
<td>Required. Determines the order in which duplicate rules are applied.</td>
</tr>
</tbody>
</table>
DuplicateRuleMatchRule

Describes the MatchingRule associated with the DuplicateRule. The MatchingRule identifies duplicate records.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>matchRuleSObjectType</td>
<td>string</td>
<td>Required. The name of the target object of the matching rule. For example, if you define a duplicate rule for Contact records, and you want to match with Lead records, the value of matchRuleSObjectType is Lead.</td>
</tr>
<tr>
<td>matchingRule</td>
<td>string</td>
<td>Required. Value that corresponds to the value of developerName in the MatchingRule for this duplicate rule.</td>
</tr>
<tr>
<td>objectMapping</td>
<td>ObjectMapping</td>
<td>Required. Foreign key to an ObjectMapping that maps fields from the duplicate rule’s object to fields in the target object specified by matchRuleSObjectType.</td>
</tr>
</tbody>
</table>

DuplicateRuleFilter

Specifies filter criteria for a DuplicateRule. Salesforce only applies the DuplicateRule if the record matches the criteria.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Required. A string of boolean operators that establishes the filter logic for the filter items specified in duplicateRuleFilterItems.</td>
</tr>
<tr>
<td>duplicateRuleFilterItems</td>
<td>DuplicateRuleFilterItem[]</td>
<td>Required. A list of DuplicateRuleFilterItem components.</td>
</tr>
</tbody>
</table>

DuplicateRuleFilterItem

This type extends the FilterItem type and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sortOrder</td>
<td>int</td>
<td>Required. The order of this item in the duplicate rule filter.</td>
</tr>
<tr>
<td>table</td>
<td>string</td>
<td>Required. The object that has the field specified in the field field of DuplicateRuleFilterItem. See the documentation for FilterItem for the definition of field.</td>
</tr>
</tbody>
</table>

ObjectMapping

Represents a map of fields in the input object of the DuplicateRule to fields in the output object of DuplicateRule. The input object is the object associated with the DuplicateRule. The output object can be the same object or a different object with similar fields.

For example, you can have a DuplicateRule that looks for duplicates between the Contact object and the Lead object. In this case, the input object is Contact, and the output object is Lead.
**Field Name** | **Field Type** | **Description**
--- | --- | ---
inputObject | string | Required. The input object for the duplicate rule. The DuplicateRule is associated with this object. For example, if you define a duplicate rule for Contact records, and you want to match with Lead records, the value of inputObject is Contact.

mappingFields | ObjectMappingField[] | Required. The mapping of source object fields to target object fields for the duplicate rule.

outputObject | string | Required. The output object for the duplicate rule. This value is the same as the value of the matchRuleSObjectType field in DuplicateRuleMatchRule. Any duplicate rules that this object has are ignored when the DuplicateRule API uses the ObjectMapping.

---

### ObjectMappingField

A field name in the input object of the DuplicateRule, and the corresponding field name in the output object.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputField</td>
<td>string</td>
<td>Required. Field in the object specified by the inputObject field in ObjectMapping. This field is mapped to the field in outputField, which is assumed to be a field in the object specified by the outputObject field in ObjectMapping.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>outputField</td>
<td>string</td>
<td>Required. Field in the object specified by the outputObject field in ObjectMapping. The field is mapped to the field name in inputField, which is assumed to be a field in the object specified by the inputObject in ObjectMapping.</td>
</tr>
</tbody>
</table>

---

### Declarative Metadata Sample Definition

The following is an example of a DuplicateRule component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<DuplicateRule xmlns="http://soap.sforce.com/2006/04/metadata"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <actionOnInsert>Allow</actionOnInsert>
    <actionOnUpdate>Allow</actionOnUpdate>
    <alertText>You are creating a duplicate record. Use an existing record instead.</alertText>
    <description>Detects a contact that duplicates a Lead</description>
    <duplicateRuleFilter>
        <booleanFilter xsi:nil="true"/>
        <duplicateRuleFilterItems>
            <field>Username</field>
            <operation>equals</operation>
            <value>user@example.com</value>
            <sortOrder>1</sortOrder>
        </duplicateRuleFilterItems>
    </duplicateRuleFilter>
</DuplicateRule>
```
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>ContactToLeadDuplicate</members>
    <name>DuplicateRule</name>
  </types>
  <version>38.0</version>
</Package>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### EclairGeoData

Represents an Analytics custom map chart. Custom maps are user-defined maps that are uploaded to Analytics and are used just as standard maps are. Custom maps are accessed in Analytics from the list of maps available with the map chart type.
File Suffix and Directory Location

EclairGeoData components have the suffix `geodata` and are stored in the `eclair` folder.

Version

EclairGeoData components are available in API version 39.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maps</td>
<td>EclairMap[]</td>
<td>A list of EclairMap objects. Each EclairMap object specifies the bounding box (if any) and the map name that appears in the user interface.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Master label for this object. This display value is the internal label that is not translated.</td>
</tr>
</tbody>
</table>

EclairMap

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boundingBoxBottom</td>
<td>double</td>
<td>When bounding-box coordinates are used, this contains the bottom coordinate.</td>
</tr>
<tr>
<td>boundingBoxLeft</td>
<td>double</td>
<td>When bounding-box coordinates are used, this contains the left side coordinate.</td>
</tr>
<tr>
<td>boundingBoxRight</td>
<td>double</td>
<td>When bounding-box coordinates are used, this contains the right side coordinate.</td>
</tr>
<tr>
<td>boundingBoxTop</td>
<td>double</td>
<td>When bounding-box coordinates are used, this contains the top coordinate.</td>
</tr>
<tr>
<td>mapLabel</td>
<td>string</td>
<td>Required. The user-interface name of the map. This name appears in the maps list for the map chart in Analytics.</td>
</tr>
<tr>
<td>mapName</td>
<td>string</td>
<td>Required. Label for this object. This display value is the internal label that is not translated.</td>
</tr>
<tr>
<td>projection</td>
<td>string</td>
<td>Required. The type of map projection used to create the map. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Equirectangular</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Mercator</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AlbersUSA</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of an EclairGeoData component:

```xml
```

418
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>EclairGeoData</name>
  </types>
  <version>39.0</version>
</Package>
```

## Wildcard Support in the Manifest File

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### EmailServicesFunction

Represents an email service. This type extends the Metadata metadata type and inherits its `fullName` field.

#### File Suffix and Directory Location

EmailServicesFunction components have the suffix `.xml` and are stored in the `emailservices` folder.
**Version**

EmailServicesFunction components are available in API version 42.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apexClass</td>
<td>string</td>
<td>Required. The name of the Apex class that the email service uses to process inbound messages.</td>
</tr>
</tbody>
</table>
| attachmentOption    | EmailServicesAttOptions (enumeration of type string) | Required. Indicates the types of attachments the email service accepts. One of the following values:  
  - None — The email service accepts the message but discards any attachment.  
  - NoContent — The attachment metadata (filename, MIME type, and so on) is provided to the Apex class, but the body is set to null.  
  - TextOnly — The email service only accepts the following types of attachments:  
    - Attachments with a Multipurpose Internet Mail Extension (MIME) type of text.  
    - Attachments with a MIME type of application/octet-stream and a file name that ends with either a .vcf or .vcs extension. These are saved as text/x-vcard and text/calendar MIME types, respectively.  
  - BinaryOnly — The email service only accepts binary attachments, such as image, audio, application, and video files.  
  - All — The email service accepts any type of attachment. |
| authenticationFailureAction | EmailServicesErrorAction (enumeration of type string) | Required. Indicates what the email service does with messages that fail or do not support any of the authentication protocols if the isAuthenticationRequired field is true.  
One of the following values:  
  - UseSystemDefault — The system default is used.  
  - Bounce — The email service returns the message to the sender with a notification that explains why the message was rejected.  
  - Discard — The email service deletes the message without notifying the sender.  
  - Requeue — The email service queues the message for processing in the next 24 hours. If the message is not processed within 24 hours, the email service returns the message to the sender with a notification that explains why the message was rejected. |
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| authorizationFailureAction | EmailServicesErrorAction (enumeration of type string) | Required. Indicates what the email service does with messages received from senders who are not listed in the authorizedSenders field on either the email service or email service address.  
One of the following values:  
  - **UseSystemDefault** — The system default is used.  
  - **Bounce** — The email service returns the message to the sender with a notification that explains why the message was rejected.  
  - **Discard** — The email service deletes the message without notifying the sender.  
  - **Requeue** — The email service queues the message for processing in the next 24 hours. If the message is not processed within 24 hours, the email service returns the message to the sender with a notification that explains why the message was rejected. |
| authorizedSenders       | string                                   | Configures the email service to only accept messages from the email addresses or domains listed in this field. If the email service receives a message from an unlisted email address or domain, the email service performs the action specified in the authorizationFailureAction field. Leave this field blank if you want the email service to receive email from any email address. |
| emailServicesAddresses   | EmailServicesAddress[]                    | A list of EmailServiceAddress records.                                                                                                                                                                     |
| errorRoutingAddress      | email                                    | The destination email address for error notification email messages when isErrorRoutingEnabled is true.                                                                                                     |
| functionInactiveAction   | EmailServicesErrorAction (enumeration of type string) | Required. Indicates what the email service does with messages it receives when the email service itself is inactive.  
One of the following values:  
  - **UseSystemDefault** — The system default is used.  
  - **Bounce** — The email service returns the message to the sender with a notification that explains why the message was rejected.  
  - **Discard** — The email service deletes the message without notifying the sender.  
  - **Requeue** — The email service queues the message for processing in the next 24 hours. If the message is not processed within 24 hours, the email service returns the message to the sender with a notification that explains why the message was rejected. |
| functionName             | string                                   | Required. The name of the email service in the API. This name can contain only underscores and alphanumeric characters and must be unique in your org. The value in this 64-character field must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. |
In managed packages, this field prevents naming conflicts on package installations. This field is automatically generated, but you can supply your own value if you create the record using the API. With this field, a developer can change the object’s name in a managed package and the changes are reflected in a subscriber’s organization.

**Note:** When creating large sets of data, always specify a unique `functionName` for each record. If no `functionName` is specified, performance may slow while Salesforce generates one for each record.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Indicates whether this object is active (true) or not (false).</td>
</tr>
<tr>
<td>isAuthenticationRequired</td>
<td>boolean</td>
<td>Configures the email service to verify the legitimacy of the sending server before processing a message.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The email service uses the SPF, SenderId, and DomainKeys protocols to verify the sender’s legitimacy: If the sending server passes at least one of these protocols and does not fail any, the email service accepts the email. If the server fails a protocol or does not support any of the protocols, the email service performs the action specified in the <code>authenticationFailureAction</code> field.</td>
</tr>
<tr>
<td>isErrorRoutingEnabled</td>
<td>boolean</td>
<td>When incoming email messages can’t be processed, indicates whether error notification email messages are routed to a chosen address or to the senders.</td>
</tr>
<tr>
<td>isTextAttachmentsAsBinary</td>
<td>boolean</td>
<td>If true, text attachments are supplied to the Apex code as a <code>Messaging.BinaryAttachment</code> instead of as a <code>Messaging.TextAttachment</code>. This means that the body is supplied as an Apex Blob instead of as an Apex String.</td>
</tr>
<tr>
<td>isTlsRequired</td>
<td>boolean</td>
<td>Not currently in use.</td>
</tr>
</tbody>
</table>
| overLimitAction             | EmailServicesErrorAction (enumeration of type string) | Required. Indicates what the email service does with messages if the total number of messages processed by all email services combined has reached the daily limit for your organization. One of the following values:  
  - UseSystemDefault—The system default is used.  
  - Bounce—The email service returns the message to the sender with a notification that explains why the message was rejected.  
  - Discard—The email service deletes the message without notifying the sender.  
  - Requeue—The email service queues the message for processing in the next 24 hours. If the message is not processed within 24 hours, the email service returns the message to the sender with a notification that explains why the message was rejected.  
The system calculates the limit by multiplying the number of user licenses by 1,000. |
EmailServicesAddress

Each email service has one or more email addresses to which users can send messages for processing. An email service only processes messages it receives at one of its addresses.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authorizedSenders</td>
<td>string</td>
<td>Configures the email service address to only accept messages from the email addresses or domains listed in this field. If the email service address receives a message from an unlisted email address or domain, the email service performs the action specified in the authorizationFailureAction field of its associated email service. Leave this field blank if you want the email service address to receive email from any email address.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. The name of the object in the API. This name can contain only underscores and alphanumeric characters and must be unique in your org. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This 25-character field must be unique among other EmailServicesAddress records under the same EmailServiceFunction parent. In managed packages, this field prevents naming conflicts on package installations. This field is automatically generated, but you can supply your own value if you create the record using the API. With this field, a developer can change the object’s name in a managed package and the changes are reflected in a subscriber’s organization.</td>
</tr>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Indicates whether this object is active (true) or not (false).</td>
</tr>
<tr>
<td>localPart</td>
<td>string</td>
<td>Required. The local-part of the email service address, which is the string that comes before the @ symbol. For the local-part of a Salesforce email address, all alphanumeric characters are valid, plus the following special characters: ! # $ % &amp; ' * / = ? ^ _ + - ` {</td>
</tr>
<tr>
<td>runAsUser</td>
<td>string</td>
<td>Required. The username of the user whose permissions the email service assumes when processing messages sent to this address.</td>
</tr>
</tbody>
</table>

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
EmailTemplate

Represents an email template.

This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.

Note: Packaging isn't supported for Lightning email templates.

File Suffix and Directory Location

The file suffix is .email for the template file. The accompanying metadata file is named EmailTemplateName-meta.xml. EmailTemplate components are stored in the email folder in the corresponding package directory. For example, for an email template named SampleTemplate in the sampleFolder folder, there's a SampleTemplate-meta.xml in the email/sampleFolder of the package.

Retrieving Email Templates

You can't use the wildcard (*) symbol with email templates in package.xml. To retrieve the list of email templates for populating package.xml with explicit names, call listMetadata() and pass in EmailTemplate as the type.

The following example shows folders in package.xml:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>MyDBFolder/MyDBName</members>
    <name>Dashboard</name>
  </types>
  <types>
    <members>MyDocumentFolder/MyDocumentName</members>
    <name>Document</name>
  </types>
  <types>
    <members>unfiled$public/MarketingProductInquiryResponse</members>
    <members>unfiled$public/SalesNewCustomerEmail</members>
    <name>EmailTemplate</name>
  </types>
  <types>
    <members>MyReportFolder/MyReportName</members>
    <name>Report</name>
  </types>
</Package>
```

Version

Email templates are available in API version 12.0 and later.
Fields

This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>double</td>
<td>The API version if this is a Visualforce email template. Every Visualforce email template has an API version specified at creation. This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>attachedDocuments</td>
<td>string[]</td>
<td>A list of references to documents in your organization. These documents are included as attachments in the email template. Each document is referenced by its path, for example MyFolder/MyDocument.txt.</td>
</tr>
<tr>
<td>attachments</td>
<td>Attachment[]</td>
<td>A list of attachments for the email template.</td>
</tr>
<tr>
<td>available</td>
<td>boolean</td>
<td>Required. Indicates whether this template is offered to users when sending an email (true) or not (false).</td>
</tr>
</tbody>
</table>
| content          | base64Binary     | Content of the email template. Base 64-encoded binary data. Prior to making an API call, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to binary. This conversion is usually handled for you by a SOAP client. This field contains:  
  • Binary content of the email body if type is set to text  
  • HTML email content if type is set to html  
  • HTML body if type is set to custom  
  • Visualforce body if type is set to visualforce  
This field is inherited from the MetadataWithContent component. |
| description      | string           | The email template description. This can be useful to describe the reason for creating the template.                                                                                                          |
| encodingKey      | Encoding (enumeration of type string) | Required for Classic email templates. The default encoding setting is Unicode: UTF-8. Change it if your template requires data in a different format. Valid values include:  
  • UTF-8—Unicode (UTF-8)  
  • ISO-8859-1—General US & Western Europe (ISO-8859-1, ISO-LATIN-1)  
  • Shift_JIS—Japanese (Shift-JIS)  
  • x-SJIS_0213—Japanese (Shift-JIS_2004)  
  • ks_c_5601-1987—Korean (ks_c_5601-1987)  
  • Big5—Traditional Chinese (Big5)  
  • GB2312—Simplified Chinese (GB2312)  
  • Big5-HKSCS—Traditional Chinese Hong Kong (Big5-HKSCS) |
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fullName</td>
<td>string</td>
<td>The email template developer name used as a unique identifier for API access. The <code>fullName</code> can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. If this field contained characters before version 14.0 that are no longer allowed, the characters were stripped out of this field, and the previous value of the field was saved in the <code>name</code> field. This field is inherited from the Metadata component. Note: Lightning email templates don’t use this field. Instead, the encoding values are taken directly from the user’s encoding settings.</td>
</tr>
<tr>
<td>letterhead</td>
<td>string</td>
<td>The letterhead name associated with this email template. Only available when <code>type</code> is set to <code>html</code>.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Email template name. The list of characters allowed in the <code>fullName</code> field has been reduced for versions 14.0 and later. This field contains the value contained in the <code>fullName</code> field before version 14.0.</td>
</tr>
<tr>
<td>packageVersions</td>
<td>PackageVersion[]</td>
<td>The list of package versions for any managed packages containing components that are referenced by this email template. This field is only relevant for Visualforce email templates. For more information about managed packages, see the Lightning Platform Quick Reference for Developing Packages. For more information about package versions, see “About Package Versions” in the Salesforce online help. This field is available in API version 16.0 and later.</td>
</tr>
<tr>
<td>relatedEntityType</td>
<td>Object Name (enumeration of type string)</td>
<td>Reserved for future use with Lightning Experience.</td>
</tr>
<tr>
<td>style</td>
<td>EmailTemplateStyle (enumeration of type string)</td>
<td>Required. The style of the template. This field is only available when <code>type</code> is set to <code>html</code>. Valid style values include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• none</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• freeForm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• formalLetter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• promotionRight</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• promotionLeft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• newsletter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• products</td>
</tr>
<tr>
<td>subject</td>
<td>string</td>
<td>The email subject.</td>
</tr>
<tr>
<td>textOnly</td>
<td>string</td>
<td>The text of the email body if <code>type</code> is set to <code>html</code> or <code>custom</code>.</td>
</tr>
</tbody>
</table>
**Field Name** | **Field Type** | **Description**
--- | --- | ---
type | EmailTemplateType (enumeration of type string) | Required. The email template type. The valid values are:
- **text** - all users can create or change text email templates.
- **html** - administrators and users with the “Edit HTML Templates” permission can create HTML email templates based on a letterhead.
- **custom** - administrators and users with the “Edit HTML Templates” permission can create custom HTML email templates without using a letterhead. You must either know HTML or obtain the HTML code to insert in your email template.
- **visualforce** - administrators and users with the “Customize Application” permission can create email templates using Visualforce.

UiType | EmailTemplateUiType (enumeration of type string) | Indicates the user interface where this template is usable. Valid values are:
- **Aloha** (Salesforce Classic)
- **SFX** (Lightning Experience)
- **SFX_Sample** (Lightning Experience Sample)

If UiType is SFX, the type must be custom.

**Note:** Packaging is supported for Salesforce Classic email templates only.

---

**Example:**

```xml
<EmailTemplate>
  <available>true</available>
  <description>Notification that user has been added to a community.</description>
  <encodingKey>UTF-8</encodingKey>
  <name>Communities: New Member Welcome Email</name>
  <style>none</style>
  <subject>Welcome to {!Community_Name}</subject>
  <type>custom</type>
  <uiType>Aloha</uiType>
</EmailTemplate>
```

**Attachment**

Attachment represents an email attachment.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
<td>base64Binary</td>
<td>Required. The attachment content. Base 64-encoded binary data. Prior to making an API call, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

A sample XML definition of an email template is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<EmailTemplate xmlns="http://soap.sforce.com/2006/04/metadata">
  <available>true</available>
  <description>Sample Email Template</description>
  <encodingKey>ISO-8859-1</encodingKey>
  <name>Sample Email Template</name>
  <style>none</style>
  <subject>Sample email subject</subject>
  <textOnly>Your case has been resolved.</textOnly>
  <type>custom</type>
</EmailTemplate>
```

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
Letterhead

EmbeddedServiceBranding

Represents the branding for each Embedded Service deployment. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

EmbeddedServiceBranding components are stored in the developer_name.EmbeddedServiceBranding file in the EmbeddedServiceBranding folder.

Version

EmbeddedServiceBranding is available in API version 39.0 and later.
### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contrastInvertedColor</td>
<td>string</td>
<td>Accent branding color used in the embedded component, displayed as a hexadecimal value. Changes made to this field in the API aren’t reflected in the embedded component.</td>
</tr>
<tr>
<td>contrastPrimaryColor</td>
<td>string</td>
<td>Accent branding color used in the embedded component, displayed as a hexadecimal value.</td>
</tr>
<tr>
<td>embeddedServiceConfig</td>
<td>string</td>
<td>Required. The Embedded Service configuration that this branding applies to.</td>
</tr>
<tr>
<td>font</td>
<td>string</td>
<td>Font used in the text of the embedded component.</td>
</tr>
<tr>
<td>height</td>
<td>int</td>
<td>Height of the embedded component. Available in API version 43.0 and later.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The name of the Embedded Service configuration node.</td>
</tr>
<tr>
<td>navBarColor</td>
<td>string</td>
<td>Color used for the navigation bar in the embedded component, displayed as a hexadecimal value.</td>
</tr>
<tr>
<td>primaryColor</td>
<td>string</td>
<td>Primary branding color used in the embedded component, displayed as a hexadecimal value.</td>
</tr>
<tr>
<td>secondaryColor</td>
<td>string</td>
<td>Secondary branding color used in the embedded component, displayed as a hexadecimal value.</td>
</tr>
<tr>
<td>width</td>
<td>int</td>
<td>Width of the embedded component. Available in API version 43.0 and later.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of an EmbeddedServiceBranding file.

```xml
< EmbeddedServiceBranding xmlns="http://soap.sforce.com/2006/04/metadata">
  < contrastInvertedColor > #ffffff </ contrastInvertedColor >
  < contrastPrimaryColor > #333333 </ contrastPrimaryColor >
  < embeddedServiceConfig > EswConfig001 </ embeddedServiceConfig >
  < font > Salesforce Sans </ font >
  < height > 498 </ height >
  < masterLabel > EmbeddedServiceBranding_Parent04IRM000000002a_16033cd2c16 </ masterLabel >
  < navBarColor > #222222 </ navBarColor >
  < primaryColor > #222222 </ primaryColor >
  < secondaryColor > #005290 </ secondaryColor >
  < width > 320 </ width >
</ EmbeddedServiceBranding >
```
Wildcard Support in the Manifest File

This metadata type doesn't support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

EmbeddedServiceConfig

Represents a setup node for creating an Embedded Service for Web deployment. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

EmbeddedServiceConfig components are stored in the developer_name.EmbeddedServiceConfig file in the EmbeddedServiceConfig folder.

Version

EmbeddedServiceConfig is available in API version 37.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>areGuestUsersAllowed</td>
<td>boolean</td>
<td>Specifies whether a user must be logged in to access an embedded component. Available in API version 45.0 and later.</td>
</tr>
</tbody>
</table>
| authMethod                  | EmbeddedServiceAuthMethod | Type of login method selected for this Embedded Service deployment. One of the following values: 
• CommunitiesLogin—Customers log in using Communities. 
• CustomLogin—Customers log in using your own custom authentication. Available in API version 43.0 and later. |
<p>| customMinimizedComponent    | string     | The custom Lightning component that’s used in this Embedded Service deployment in its minimized state. Available in API version 43.0 and later. |
| embeddedServiceCustomComponents | EmbeddedServiceCustomComponent | The custom components used in this Embedded Service deployment. Available in API version 44.0 and later. |
| embeddedServiceCustomLabels | EmbeddedServiceCustomLabel | The custom labels used in this Embedded Service deployment. Available in API version 44.0 and later. |
| embeddedServiceFlowConfig   | EmbeddedServiceFlowConfig | Represents a setup node for creating an embedded flow. Available in API version 45.0 and later. |
| embeddedServiceFlows        | EmbeddedServiceFlow | All of the flows used by this Embedded Service deployment. Available in API version 45.0 and later. |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The name of the Embedded Service configuration node. Available in API version 37.0 and later.</td>
</tr>
<tr>
<td>shouldHideAuthDialog</td>
<td>boolean</td>
<td>Specifies whether the prompt that the customer login again during a flow is hidden (true) or not (false). When it’s hidden, the customer is taken directly to your login page. This field is set to false by default. Available in API version 43.0 and later.</td>
</tr>
<tr>
<td>site</td>
<td>string</td>
<td>Required. The name of the Salesforce Community or site connected to this Embedded Service deployment. Available in API version 37.0 and later.</td>
</tr>
</tbody>
</table>

**EmbeddedServiceCustomComponent**

Returns a custom component that’s associated with an EmbeddedServiceConfig setup.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customComponent</td>
<td>string</td>
<td>The name of the custom component.</td>
</tr>
<tr>
<td>customComponentType</td>
<td>EmbeddedServiceCustomComponent (enumeration of type string)</td>
<td>The type of custom component. One of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LA_Prechat (component for pre-chat in Embedded Chat)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LA_Minimized (component for the minimized chat window)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LA_PlainTextChatMessage (component for the text area in Embedded Chat)</td>
</tr>
</tbody>
</table>

**EmbeddedServiceCustomLabel**

Returns a custom label that’s associated with an EmbeddedServiceConfig setup.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customLabel</td>
<td>string</td>
<td>The customized label that appears in the embedded component.</td>
</tr>
<tr>
<td>feature</td>
<td>EmbeddedServiceFeature (enumeration of type string)</td>
<td>The feature that this embedded component uses. One of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Base</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FieldService</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Flows</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LiveAgent</td>
</tr>
</tbody>
</table>

431
### Field Name | Field Type | Description
--- | --- | ---
labelKey | EmbeddedServiceLabelKey (enumeration of type string) | The type of label for this embedded component. The value corresponds to the label within a label group (substate of chat state or page type).

## EmbeddedServiceFlow

Returs an embedded flow that’s associated with an EmbeddedServiceConfig setup.

### Field Name | Field Type | Description
--- | --- | ---
flow | string | The developer name of the flow.
flowType | EmbeddedServiceFlowType (enumeration of type string) | The type of flow. One of the following values:
- FS_Flow
- FS_NewAppointment
- FS_ModifyAppointment
- FS_CancelAppointment
isAuthenticationRequired | boolean | Indicates whether users are required to log in to access the Embedded Service component. The value cannot be true for the FS_Flow value and must be true for all other values.

## EmbeddedServiceFlowConfig

Returns the EmbeddedServiceFlowConfig type.

### Field Name | Field Type | Description
--- | --- | ---
enabled | boolean | Indicates whether the embedded flow is enabled.

## Declarative Metadata Sample Definition

The following is an example of an EmbeddedServiceConfig file.

```
<?xml version="1.0" encoding="UTF-8"?>
<EmbeddedServiceConfig xmlns="http://soap.sforce.com/2006/04/metadata">
  <authMethod>CommunitiesLogin</authMethod>
</EmbeddedServiceConfig>
```

432
Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

EmbeddedServiceFieldService

Represents a setup node for creating an embedded Appointment Management deployment. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

EmbeddedServiceFieldService components are stored in a developer_name.EmbeddedServiceFieldService file in the EmbeddedServiceFieldService folder.

Version

EmbeddedServiceFieldService is available in API version 43.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appointmentBookingFlowName</td>
<td>string</td>
<td>Name of the appointment booking flow for this embedded Appointment Management (beta) deployment.</td>
</tr>
<tr>
<td>cancelApptBookingFlowName</td>
<td>string</td>
<td>Name of the appointment cancellation flow for this embedded Appointment Management (beta) deployment.</td>
</tr>
<tr>
<td>embeddedServiceConfig</td>
<td>string</td>
<td>Required. The name of the Embedded Service configuration node.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether this embedded Appointment Management deployment is enabled (true).</td>
</tr>
<tr>
<td>fieldServiceConfirmCardImg</td>
<td>string</td>
<td>URL of the image used for the confirmation card in embedded Appointment Management (beta).</td>
</tr>
<tr>
<td>fieldServiceHomeImg</td>
<td>string</td>
<td>URL of the image used for the home screen in embedded Appointment Management (beta).</td>
</tr>
<tr>
<td>fieldServiceLogoImg</td>
<td>string</td>
<td>URL of the logo used for the home screen in embedded Appointment Management (beta).</td>
</tr>
</tbody>
</table>
**Field Name** | **Field Type** | **Description**
--- | --- | ---
masterLabel | string | Required. Name of the embedded Appointment Management (beta) deployment.
modifyApptBookingFlowName | string | Name of the appointment modification flow for this embedded Appointment Management (beta) deployment.
shouldShowExistingAppointment | boolean | Specifies whether to display a button on the home screen for customers to access their existing appointments (true) or not (false). This field is false by default.
shouldShowNewAppointment | boolean | Specifies whether to display a button on the home screen for customers to create a new appointment (true) or not (false). This field is false by default.

### Declarative Metadata Sample Definition

The following is an example of an EmbeddedServiceFieldService file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<EmbeddedServiceFieldService xmlns="http://soap.sforce.com/2006/04/metadata">
  <appointmentBookingFlowName>ESW_FS_BookAppt_Main_Flow</appointmentBookingFlowName>
  <cancelApptBookingFlowName>ESW_FS_CancelAppt_Flow</cancelApptBookingFlowName>
  <embeddedServiceConfig>EswFS</embeddedServiceConfig>
  <enabled>true</enabled>
  <fieldServiceConfirmCardImg>https://google.com/AppointmentConfirmationImg.png</fieldServiceConfirmCardImg>
  <fieldServiceHomeImg>https://google.com/HeroImg.png</fieldServiceHomeImg>
  <fieldServiceLogoImg>https://google.com/logo.png</fieldServiceLogoImg>
  <masterLabel>EmbeddedServiceFieldService_Parent04IRM000000007p2AA_162d4270834</masterLabel>
  <modifyApptBookingFlowName>ESW_FS_ModifyAppt_Main_Flow</modifyApptBookingFlowName>
  <shouldShowExistingAppointment>true</shouldShowExistingAppointment>
  <shouldShowNewAppointment>true</shouldShowNewAppointment>
</EmbeddedServiceFieldService>
```

### Usage

Note: Any changes you make to the image fields override what you’ve entered in Setup. We recommend setting your image URLs in Setup.

**EmbeddedServiceFlowConfig**

Represents a setup node for creating an embedded flow. This type extends the Metadata metadata type and inherits its fullName field.
File Suffix and Directory Location


Version

EmbeddedServiceFlowConfig is available in API version 45.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Indicates whether the embedded flow is enabled (<code>true</code>) or not (<code>false</code>). Defaults to <code>false</code>.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of an EmbeddedServiceFlowConfig file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<EmbeddedServiceFlowConfig xmlns="http://soap.sforce.com/2006/04/metadata">
  <enabled>true</enabled>
</EmbeddedServiceFlowConfig>
```

EmbeddedServiceLiveAgent

Represents a setup node for creating an embedded chat deployment. This type extends the `Metadata` metadata type and inherits its `fullName` field.

File Suffix and Directory Location

EmbeddedServiceLiveAgent components are stored in the `developer_name.EmbeddedServiceLiveAgent` file in the `EmbeddedServiceLiveAgent` folder.

Version

EmbeddedServiceLiveAgent is available in API version 38.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>avatarImg</td>
<td>string</td>
<td>Avatar image for this embedded chat deployment.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>customPrechatComponent</td>
<td>string</td>
<td>The custom Lightning Component that’s used for the pre-chat page in this embedded chat deployment.</td>
</tr>
<tr>
<td>embeddedServiceConfig</td>
<td>string</td>
<td>Required. The name of the embedded service configuration node.</td>
</tr>
<tr>
<td>embeddedServiceQuickActions</td>
<td>EmbeddedServiceQuickAction</td>
<td>The quick action used by the pre-chat form.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether this embedded chat deployment is enabled (true).</td>
</tr>
</tbody>
</table>
| fontSize                 | EmbeddedServiceFontSize (enumeration of type string) | Required. The font size for the text in the embedded chat window. One of the following values:  
  • Small  
  • Medium  
  • Large |
| headerBackgroundImg      | string                     | Header background image for this embedded chat window.                       |
| isOfflineCaseEnabled     | boolean                    | Indicates whether offline support is enabled for this embedded chat deployment. Available in API version 43.0 and later. |
| isQueuePositionEnabled   | boolean                    | Indicates whether queue position (displaying the chat visitor’s place in line while they wait for an agent) is enabled for this embedded chat deployment. Available in API version 43.0 and later. |
| liveAgentChatUrl         | string                     | The rest endpoint for chats.                                               |
| liveAgentContentUrl      | string                     | The rest endpoint for cChat content.                                       |
| liveChatButton           | string                     | Required. Reference to a chat button created in Chat setup.                |
| liveChatDeployment       | string                     | Required. Reference to a deployment created in Chat setup.                 |
| masterLabel              | string                     | Required. Name of the embedded chat deployment.                            |
| offlineCaseBackgroundImg | string                     | Offline support case form background image for this embedded chat window. Available in API version 43.0 and later. |
| prechatBackgroundImg     | string                     | Pre-chat background image for this embedded chat window.                   |
| prechatEnabled           | string                     | Required. Indicates whether the embedded chat pre-chat form is enabled for this deployment. |
| prechatJson              | string                     | JSON object of all the fields of the selected pre-chat form in Chat setup. |
| scenario                 | EmbeddedServiceScenario (enumeration of type string) | Required. The scenario for the embedded chat window that determines which objects to relate to the chat. One of the following values:  
  • Sales  
  • Service  
  • Basic |
| smallCompanyLogoImg      | string                     | Company logo image for this embedded chat window.                          |
### EmbeddedServiceQuickAction

Returns a quick action that’s associated with an EmbeddedServiceLiveAgent setup. The quick action include the pre-chat form fields that the embedded chat window displays and shows the order in which he fields are displayed.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>embeddedServiceLiveAgent</td>
<td>string</td>
<td>Reference to the embedded chat deployment.</td>
</tr>
<tr>
<td>order</td>
<td>int</td>
<td>Order in which this quick action appears in the embedded chat pre-chat form.</td>
</tr>
<tr>
<td>quickActionDefinition</td>
<td>string</td>
<td>Reference to a quick action.</td>
</tr>
<tr>
<td>quickActionType</td>
<td>EmbeddedServiceQuickType (enumeration of type string)</td>
<td>Quick action type. One of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Prechat–Pre-chat</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• OfflineCase–Offline support (Cases)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Available in API version 43.0 and later.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of an EmbeddedServiceLiveAgent file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<EmbeddedServiceLiveAgent xmlns="http://soap.sforce.com/2006/04/metadata">
  <avatarImg>https://google.com/avatar.png</avatarImg>
  <customPrechatComponent>auraCustomPrechat</customPrechatComponent>
  <embeddedServiceConfig>EswConfig001</embeddedServiceConfig>
  <embeddedServiceQuickActions>
    <embeddedServiceLiveAgent>EmbeddedServiceLiveAgent_Parent04Ixx0000000001EAA_15ec5bd2971</embeddedServiceLiveAgent>
    <order>1</order>
    <quickActionDefinition>Snapins_Contact_QuickAction_08hRM0000000001h</quickActionDefinition>
  </embeddedServiceQuickActions>
  <embeddedServiceQuickActions>
    <embeddedServiceLiveAgent>EmbeddedServiceLiveAgent_Parent04Ixx0000000001EAA_15ec5bd2971</embeddedServiceLiveAgent>
    <order>1</order>
    <quickActionType>OfflineCase</quickActionType>
    <quickActionDefinition>Snapins_Case_OfflineCaseQuickAction_08hRM0000000001h</quickActionDefinition>
  </embeddedServiceQuickActions>
</EmbeddedServiceLiveAgent>
```
Usage

EmbeddedServiceLiveAgent represents a Chat configuration that is added to your web page. The EmbeddedServiceLiveAgent record contains a unique combination of a chat button and the Chat deployment that the administrator selects during setup.

To create an EmbeddedServiceLiveAgent record:

1. Create a Chat Deployment record.
2. Create a Chat Button record.
3. Create an EmbeddedServiceConfig record.
4. Set the fields for the Chat Deployment record, Chat Button record, and EmbeddedServiceConfig record as references on the EmbeddedServiceLiveAgent record.

Note: Any changes you make to the image fields override what you’ve entered in Setup. We recommend setting your image URLs in Setup.

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
EntitlementProcess

Represents the settings for an entitlement process. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

Entitlement process values are stored in files in the entitlementProcesses directory. Each file has the name of a process and the suffix .entitlementProcess. Each file contains one entitlement process or, if entitlement versioning is enabled, one version of an entitlement process.

The name of the file is the name of the entitlement process with the version appended to the end, if applicable (for example, an entitlement process named “gold_support” might have the file name “gold_support_v2.entitlementProcess”). This file name corresponds to the slaProcess.NameNorm field exposed through the SOAP API. This file name is distinct from the name field, which represents what displays in the user interface and, if versioning is enabled, might be shared among multiple versions of the same entitlement process. The slaProcess.NameNorm field contains the lowercase version of the name field shown in the user interface.

Version

Entitlement processes are available in API version 27.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Indicates whether the entitlement process is active (true) or not (false).</td>
</tr>
<tr>
<td>businessHours</td>
<td>string</td>
<td>The business hours that apply to the entitlement process.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the entitlement process.</td>
</tr>
<tr>
<td>entryStartDateField</td>
<td>string</td>
<td>For milestone processes on which a case enters the process based on a custom date/time field on the case, specifies which date and time are used. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SlaStartDate (entitlement process start date)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CreatedDate (date case was opened)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ClosedDate (date case was closed)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LastModifiedDate (date case was last modified)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• StopStartDate (date case was stopped)</td>
</tr>
<tr>
<td>exitCriteriaBooleanFilter</td>
<td>string</td>
<td>For milestone processes on which a case exits the process when custom criteria are met, and for which filter logic is added, specifies that logic.</td>
</tr>
<tr>
<td>exitCriteriaFilterItems</td>
<td>FilterItem[]</td>
<td>For milestone processes on which a case exits the process when custom criteria are met, specifies those criteria.</td>
</tr>
</tbody>
</table>
### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>exitCriteriaFormula</td>
<td>string</td>
<td>For milestone processes on which a case exits the process when a custom formula evaluates to true, specifies that formula.</td>
</tr>
<tr>
<td>isVersionDefault</td>
<td>boolean</td>
<td>Indicates whether the entitlement process is the default version (true) or not (false). This field is available in API version 28.0 and later.</td>
</tr>
<tr>
<td>milestones</td>
<td>EntitlementProcessMilestoneItem[]</td>
<td>Represents a milestone on the entitlement process.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the entitlement process as it displays in the user interface.</td>
</tr>
<tr>
<td>SObjectType</td>
<td>string</td>
<td>Indicates the type of record that the entitlement process can run on.</td>
</tr>
<tr>
<td>versionMaster</td>
<td>string</td>
<td>Identifies the sequence of versions to which this entitlement process belongs. This field’s contents can be any value as long as it is identical among all versions of the entitlement process. This field is available in API version 28.0 and later.</td>
</tr>
<tr>
<td>versionNotes</td>
<td>string</td>
<td>The description of the entitlement process version. This field is available in API version 28.0 and later.</td>
</tr>
<tr>
<td>versionNumber</td>
<td>int</td>
<td>The version number of the entitlement process. Must be 1 or greater. This field is available in API version 28.0 and later.</td>
</tr>
</tbody>
</table>

#### EntitlementProcessMilestoneItem

Represents a milestone item on an entitlement process.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>businessHours</td>
<td>string</td>
<td>The business hours that apply to the milestone. This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>criteriaBooleanFilter</td>
<td>string</td>
<td>For milestones that apply only when criteria are met and for which filter logic is added, specifies that logic.</td>
</tr>
<tr>
<td>milestoneCriteriaFilterItems</td>
<td>FilterItem[]</td>
<td>For milestones that apply only when criteria are met, specifies those criteria.</td>
</tr>
</tbody>
</table>
Metadata Types

EntitlementProcessMetadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milestoneCriteriaFormula</td>
<td>string</td>
<td>For milestones that apply only when a formula evaluates to true, specifies that formula.</td>
</tr>
<tr>
<td>milestoneName</td>
<td>string</td>
<td>The name of the milestone.</td>
</tr>
<tr>
<td>minutesCustomClass</td>
<td>string</td>
<td>The name of the Apex class that is used to calculate the trigger time. This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>minutesToComplete</td>
<td>int</td>
<td>The number of minutes from when the case enters the entitlement process that the milestone occurs.</td>
</tr>
<tr>
<td>successActions</td>
<td>WorkflowActionReference[]</td>
<td>The actions triggered when the milestone is completed.</td>
</tr>
<tr>
<td>timeTriggers</td>
<td>EntitlementProcessMilestoneTimeTrigger[]</td>
<td>The time triggers on an entitlement process milestone.</td>
</tr>
<tr>
<td>useCriteriaStartTime</td>
<td>boolean</td>
<td>When the milestone starts: when the milestone criteria are met (true) or when the case enters the entitlement process (false).</td>
</tr>
</tbody>
</table>

EntitlementProcessMilestoneTimeTrigger

Represents the time trigger on an entitlement process milestone.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actions</td>
<td>WorkflowActionReference[]</td>
<td>The actions to take when the time trigger is reached, if, at that time, the milestone is not completed.</td>
</tr>
<tr>
<td>timeLength</td>
<td>int</td>
<td>The length of time between the time trigger activation and the milestone target completion date. This may be a negative or positive value. Negative values indicate that the target completion date has not yet arrived and correspond to warning time triggers. Positive values indicate that the target completion date has passed and correspond to violation time triggers.</td>
</tr>
</tbody>
</table>
| workflowTimeTriggerUnit     | MilestoneTimeUnits (enumeration of type string) | Specifies the type of unit used to determine when a workflow should be triggered. Valid values are:
  • Minutes
  • Hours
  • Days  |
Declarative Metadata Sample Definition

This is a sample entitlement process.

<?xml version="1.0" encoding="UTF-8"?>
<EntitlementProcess xmlns="http://soap.sforce.com/2006/04/metadata">
  <active>true</active>
  <description>eperson</description>
  <entryStartDateField>SlaStartDate</entryStartDateField>
  <exitCriteriaBooleanFilter>1 OR 2</exitCriteriaBooleanFilter>
  <exitCriteriaFilterItems>
    <field>Case.IsClosed</field>
    <operation>equals</operation>
    <value>true</value>
  </exitCriteriaFilterItems>
  <exitCriteriaFilterItems>
    <field>Case.Description</field>
    <operation>startsWith</operation>
    <value>foo</value>
  </exitCriteriaFilterItems>
  <milestones>
    <milestoneName>m1</milestoneName>
    <minutesToComplete>1</minutesToComplete>
    <successActions>
      <name>emailBob</name>
      <type>Alert</type>
    </successActions>
    <timeTriggers>
      <actions>
        <name>emailAlice</name>
        <type>Alert</type>
      </actions>
      <actions>
        <name>setEscalateToTrue</name>
        <type>FieldUpdate</type>
      </actions>
      <timeLength>1</timeLength>
      <workflowTimeTriggerUnit>Minutes</workflowTimeTriggerUnit>
    </timeTriggers>
    <timeTriggers>
      <actions>
        <name>setStopToTrue</name>
        <type>FieldUpdate</type>
      </actions>
      <timeLength>2</timeLength>
      <workflowTimeTriggerUnit>Minutes</workflowTimeTriggerUnit>
    </timeTriggers>
    <useCriteriaStartTime>false</useCriteriaStartTime>
  </milestones>
  <milestones>
    <milestoneCriteriaFilterItems>
      <field>Case.Priority</field>
      <operation>equals</operation>
      <value>High</value>
    </milestoneCriteriaFilterItems>
  </milestones>
</EntitlementProcess>
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

EntitlementTemplate

Represents an entitlement template. Entitlement templates are predefined terms of customer support that you can quickly add to products. For example, you can create entitlement templates for Web or phone support so that users can easily add entitlements to products offered to customers.

EntitlementTemplate extends the Metadata metadata type and inherits its fullName field.

Declarative Metadata File Suffix and Directory Location

EntitlementTemplate components are stored in the entitlementTemplates directory of the corresponding package directory. The file name matches the unique name of the entitlement template, and the extension is .entitlementTemplate.

Version

Lightning Platform EntitlementTemplate components are available in API version 18.0 and higher.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>businessHours</td>
<td>string</td>
<td>The entitlement’s supported business hours.</td>
</tr>
<tr>
<td>casesPerEntitlement</td>
<td>int</td>
<td>Lets you limit the number of cases the entitlement supports.</td>
</tr>
<tr>
<td>entitlementProcess</td>
<td>string</td>
<td>The entitlement process associated with the entitlement.</td>
</tr>
<tr>
<td>isPerIncident</td>
<td>boolean</td>
<td>true if entitlements created from this template service a limited number of cases; false otherwise.</td>
</tr>
<tr>
<td>term</td>
<td>int</td>
<td>The number of days the entitlement is in effect.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>The type of entitlement, such as Web or phone support.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

A sample XML definition of an entitlement template is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<EntitlementTemplate xmlns="http://soap.sforce.com/2006/04/metadata">
  <businessHours>AlternateBusinessHours</businessHours>
  <casesPerEntitlement>12</casesPerEntitlement>
  <entitlementProcess>Process1</entitlementProcess>
  <isPerIncident>true</isPerIncident>
  <term>33</term>
  <type>Phone Support</type>
</EntitlementTemplate>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

EscalationRules

Represents case escalation rules to escalate cases automatically if they are not resolved within a certain period of time. You can access rules metadata for all applicable objects, for a specific object, or for a specific rule on a specific object.

The `package.xml` syntax for accessing all escalation rules for all objects is:

```xml
<types>
  <members>*</members>
  <name>EscalationRules</name>
</types>
```

All rules for a specific object uses a similar syntax without the wildcard. For example, all escalation rules for the Case object would use this syntax:

```xml
<types>
  <members>Case</members>
  <name>EscalationRules</name>
</types>
```

You can also access specific escalation rules for an object. The following example only accesses the "samplerule" and "newrule" escalation rules on the Case object. Notice that for this example the type name syntax is `EscalationRule` and not `EscalationRules`.

```xml
<types>
  <members>Case.samplerule</members>
  <members>Case.newrule</members>
  <name>EscalationRule</name>
</types>
```

File Suffix and Directory Location

EscalationRules for an object have the suffix `.escalationRules` and are stored in the `escalationRules` folder. For example, all Case escalation rules are stored in the `Case.escalationRules` file.
Version

EscalationRules components are available in API version 27.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>escalationRule</td>
<td>EscalationRule[]</td>
<td>Represents one escalation rule and specifies whether it is active or not. Escalation rules are processed in the order they appear in the EscalationRules container.</td>
</tr>
</tbody>
</table>

EscalationRule

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Indicates whether the escalation rule is active (true) or not (false).</td>
</tr>
<tr>
<td>fullname</td>
<td>string</td>
<td>Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See createMetadata() to see an example of this field specified for a call. This value cannot be null.</td>
</tr>
<tr>
<td>ruleEntry</td>
<td>RuleEntry[]</td>
<td>Contains the definitions of the rule entries in the escalation rule.</td>
</tr>
</tbody>
</table>

RuleEntry

Represents the fields used by the rule.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Advanced filter conditions that were specified for the rule.</td>
</tr>
<tr>
<td>businessHours</td>
<td>string</td>
<td>The hours at which escalation actions are performed. Specify only if businessHoursSource is set to Static.</td>
</tr>
</tbody>
</table>
| businessHoursSource | BusinessHoursSourceType (enumeration of type string) | Valid values are:  
  • None  
  • Case  
  • Static |
| criteriaItems   | FilterItem         | The items in the list that define the assignment criteria.                                                                                 |
### Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>disableEscalationWhenModified</td>
<td>boolean</td>
<td>Indicates whether the escalation is disabled when the record is modified true or not (false).</td>
</tr>
<tr>
<td>escalationAction</td>
<td>EscalationAction[]</td>
<td>The actions to perform when the escalation criteria are met.</td>
</tr>
<tr>
<td>escalationStartTime</td>
<td>EscalationStartTimeType</td>
<td>Indicates the start time for the escalation. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• CaseCreation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CaseLastModified</td>
</tr>
<tr>
<td>formula</td>
<td>string</td>
<td>The validation formula.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: Specify either formula or criteriaItems, but not both fields.</td>
</tr>
</tbody>
</table>

### EscalationAction

Describes the action to take for an escalation rule.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignedTo</td>
<td>string</td>
<td>The name of the user or queue the item is assigned to.</td>
</tr>
<tr>
<td>assignedToTemplate</td>
<td>string</td>
<td>Specifies the template to use for the email that is automatically sent to the new owner specified by the escalation rule. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>assignedToType</td>
<td>AssignToLookupValueType</td>
<td>Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• User</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Queue</td>
</tr>
<tr>
<td>minutesToEscalation</td>
<td>int</td>
<td>The number of minutes until the escalation occurs.</td>
</tr>
<tr>
<td>notifyCaseOwner</td>
<td>boolean</td>
<td>Indicates that the owner of the case is notified when the case is escalated true or not (false).</td>
</tr>
<tr>
<td>notifyEmail</td>
<td>string</td>
<td>Specifies the email address of the user to notify.</td>
</tr>
<tr>
<td>notifyTo</td>
<td>string</td>
<td>Specifies the user to notify.</td>
</tr>
<tr>
<td>notifyToTemplate</td>
<td>string</td>
<td>Specifies the template to user for the notification email.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example EscalationRules component:

```xml
<EscalationRules xmlns="http://soap.sforce.com/2006/04/metadata">
  <escalationRule>
    <fullName>samplerule</fullName>
    <active>false</active>
    <ruleEntry>
      <businessHours>test</businessHours>
      <businessHoursSource>Static</businessHoursSource>
      <criteriaItems>
        <field>Case.Description</field>
        <operation>equals</operation>
        <value>test</value>
      </criteriaItems>
      <escalationAction>
        <assignedTo>someuser@org.com</assignedTo>
        <assignedToTemplate>emailtemplatename</assignedToTemplate>
        <assignedToType>User</assignedToType>
        <minutesToEscalation>1440</minutesToEscalation>
        <notifyCaseOwner>false</notifyCaseOwner>
      </escalationAction>
      <escalationStartTime>CaseLastModified</escalationStartTime>
    </ruleEntry>
  </escalationRule>
</EscalationRules>
```

EventDelivery

Represents how an event instance maps to a target payload. Removed in API version 46.0. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

Event delivery components have the suffix file path .delivery, and are stored in the eventDeliveries folder.

Version

Event delivery components are available in API versions 41.0 to 45.0.

Limits

Your org can have a maximum of 2500 EventDelivery object instances.
### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventParameters</td>
<td>EventParameterMap[]</td>
<td>An array of parameters to deliver in addition to the published event’s data.</td>
</tr>
<tr>
<td>eventSubscription</td>
<td>string</td>
<td>Required. The ID of the subscription to deliver the data to.</td>
</tr>
<tr>
<td>referenceData</td>
<td>string</td>
<td>User-defined non-unique identifier.</td>
</tr>
<tr>
<td>type</td>
<td>EventDeliveryType</td>
<td>Required. Determines what action occurs when the event is delivered to the listeners on behalf of the subscribers. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• StartFlow—When the event occurs, it’s delivered to a flow of type CustomEvent. Those flows are built through Process Builder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ResumeFlow—Reserved for future use.</td>
</tr>
</tbody>
</table>

### EventParameterMap

Parameters to deliver in addition to the published event’s data.

If `type` is `StartFlow`, you must include a parameter where `parameterName` is `FlowVersionName` and `parameterValue` is the name of the flow that you want to start. The flow name must include its version number. For example, `myFlow-3`.

Each event delivery can have up to 10 parameters.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameterName</td>
<td>string</td>
<td>The parameter name.</td>
</tr>
<tr>
<td>parameterValue</td>
<td>string</td>
<td>The parameter value.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of an event delivery file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<EventDelivery xmlns="http://soap.sforce.com/2006/04/metadata">
  <eventParameters>
    <parameterName>FlowVersionName</parameterName>
    <parameterValue>My_Event_Based_Process-1</parameterValue>
  </eventParameters>
  <eventSubscription>MySubscription</eventSubscription>
  <referenceData>My_Event_Based_Process_1</referenceData>
  <type>StartFlow</type>
</EventDelivery>
```

The following is an example `package.xml` that deploys or retrieves all the available event delivery metadata in your org.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
```

448
**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**EventSubscription**

Represents a subscription to an event type. Removed in API version 46.0. This type extends the `Metadata` metadata type and inherits its `fullName` field.

**File Suffix and Directory Location**

EventSubscription components have the suffix file path `.subscription`, and are stored in the `eventSubscriptions` folder.

**Version**

Event subscription components are available in API versions 41.0 to 45.0.

**Limits**

Your org can have a maximum of:

- 4,000 total event subscriptions
- 2,000 active event subscriptions

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>If the subscription isn’t active, it never receives any events.</td>
</tr>
<tr>
<td>eventParameters</td>
<td><code>EventParameterMap[]</code></td>
<td>An array of parameters that must be true for published events.</td>
</tr>
<tr>
<td>eventType</td>
<td>string</td>
<td>Required. The name of the platform event.</td>
</tr>
<tr>
<td>referenceData</td>
<td>string</td>
<td>Required. If the subscriber is a flow of type CustomEvent, <code>referenceData</code> is <code>flowName_versionNumber</code>. For example, <code>Printer_Management_2</code>.</td>
</tr>
</tbody>
</table>

**EventParameterMap**

An array of parameters that must be true for published events. For example, subscribe to Vendor Response events only if `Status__c` is `Shipped`. 

---

449
Each event subscription can have up to 10 parameters.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameterName</td>
<td>string</td>
<td>Required. The published event’s field name.</td>
</tr>
<tr>
<td>parameterValue</td>
<td>string</td>
<td>The value that must be true.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example of an active event subscription.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<EventSubscription xmlns="http://soap.sforce.com/2006/04/metadata">
  <active>true</active>
  <eventType>Printer_Status__e</eventType>
  <referenceData>Printer_Management</referenceData>
</EventSubscription>
```

The following is an example of an inactive event subscription that sets event parameters.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<EventSubscription xmlns="http://soap.sforce.com/2006/04/metadata">
  <name>MySubscription</name>
  <active>false</active>
  <eventParameters>
    <parameterName>Ink_Status__c</parameterName>
    <parameterValue>low</parameterValue>
  </eventParameters>
  <eventParameters>
    <parameterName>Serial_Number__c</parameterName>
    <parameterValue>00123456789</parameterValue>
  </eventParameters>
  <eventType>Printer_Status__e</eventType>
  <referenceData>My_Event_Based_Process_1</referenceData>
</EventSubscription>
```

The following is an example `package.xml` that deploys or retrieves all the available event subscription metadata in your org.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>EventSubscription</name>
  </types>
  <version>41.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
ExperienceBundle (Developer Preview)

Represents a text-based code structure of Community settings and site components, such as pages, branding sets, and themes. Developers can quickly update and deploy one or more Lightning communities *programmatically* using their preferred development tools. This type extends the `Metadata` metadata type and inherits its `fullName` field.

ExperienceBundle is available as a developer preview. ExperienceBundle isn’t generally available unless or until Salesforce announces its general availability in documentation or in press releases or public statements. All commands, parameters, and other features are subject to change or deprecation at any time, with or without notice. Don’t implement functionality developed with these commands or tools.

File Suffix and Directory Location

ExperienceBundle components have the suffix `.json` and are stored in the `experiences` folder when retrieved. Each community in your org has its own folder. Each of these folders contains other folders for the supported properties on page 452.

The ExperienceBundle can contain one or more community definitions under the `experiences` folder. Each community definition has resource folders for `brandingSets`, `config`, `routes`, `themes`, and `views` each with additional, related configuration information in JSON files. Here’s an example community definition.

Version

ExperienceBundle components are available in API version 46.0 and later.

*Note:* This release contains a Developer Preview version of ExperienceBundle Metadata API that is production quality but has known limitations. To provide feedback and suggestions, go to IdeaExchange.

Special Access Rules

While in Developer Preview, ExperienceBundle is available only in Developer Edition orgs. To access ExperienceBundle, select “Enable ExperienceBundle Metadata API” in the Community Management Settings section under Settings > Community Settings.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>experienceResources</td>
<td>ExperienceResources</td>
<td>The list of resources in this ExperienceBundle. Each resource represents an artifact of a community such as brandingSets, config, routes, themes, and views.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Represents the name of the ExperienceBundle.</td>
</tr>
<tr>
<td>type</td>
<td>SiteType</td>
<td>Required. Identifies the kind of site. Only Lightning communities are supported, using the value ChatterNetworkPicasso.</td>
</tr>
</tbody>
</table>

ExperienceResources

Represents a list of communities in the bundle.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>experienceResource</td>
<td>ExperienceResource</td>
<td>The list of resources in this ExperienceBundle. Each resource represents a property for the community, such as brandingSets, config, routes, themes, and views.</td>
</tr>
</tbody>
</table>

ExperienceResource

Represents specific community information included in the ExperienceBundle.

Each type has a folder in the structure. Each folder contains one or more files providing information about that type and the site. Each corresponds to a specific folder and file in the ExperienceBundle.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileName</td>
<td>string</td>
<td>Required. Name of resource file.</td>
</tr>
<tr>
<td>format</td>
<td>string</td>
<td>Required. Only JSON is allowed.</td>
</tr>
<tr>
<td>source</td>
<td>base64</td>
<td>The JSON content of each file.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Required. The type of the resource. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• brandingSets</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• config</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• routes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• themes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• views</td>
</tr>
</tbody>
</table>

Folders and Bundled Definitions

Each ExperienceBundle includes five folders and associated data that is contained in JSON files.
brandingSets Folder

This folder contains one JSON file per brandingset, named \textit{brandingSets\_name}.json. Each file has the same structure and properties.

\texttt{<brandingSets\_name>.json}

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| definitionName    | string | Required. Represents the name for the branding set that is used in grouping branding sets under a theme. Defined as \textit{theme:branding\_theme}. For example, if the community theme is Stella, the \textit{definitionName} would be \textit{stella:branding-stella}. In addition, there are several standard templates that have unique naming:  
  \begin{itemize}
  \item Customer Account Portal uses \textit{cpt:branding-cpt}
  \item Customer Service uses \textit{service:branding-service}
  \item Help Center uses \textit{helpCenter:branding-helpCenter}
  \item Partner Central uses \textit{prm:branding-prm}
  \item Build Your Own uses \textit{starter:branding-starter}
  \end{itemize}
  
  \textbf{Note:} The combination of \textit{definitionName} + \textit{label} must be unique in your org. |
| id                | UUID   | Represents the component’s GUID.                                                                                                             |
| label             | string | Represents the name of the branding set.                                                                                                     |
|                   |        | \textbf{Note:} The combination of \textit{definitionName} + \textit{label} must be unique in your org.                                      |
| type              | string | Represents the component type. The only supported value is \textit{brandingSet}.                                                           |
| values            | map    | Required. Represents a map of branding values that can be applied to a community.                                                            |

```json
{
  "values" : {
```
The `config` folder contains several JSON files.

- `sitename.json`
- `languages.json`
- `page_name.json`

**Note:** One for each single-page application in the site: `loginAppPage.json` and `mainAppPage.json`
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>forgotPasswordRouteId</td>
<td>UUID</td>
<td>Represents the ID of the route to use when a user forgets their password.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Not supported if the active template for the community doesn’t support login (such as Help Center).</td>
</tr>
<tr>
<td>isAvailableToGuests</td>
<td>boolean</td>
<td>Indicates whether public users have access to the site (true) or not (false). The default value is false.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Not supported if the active template for the community doesn’t support login (such as Help Center).</td>
</tr>
<tr>
<td>isFilteredComponentsView</td>
<td>boolean</td>
<td>Indicates whether the list of components is filtered based on the current page type (true) or not (false). Some components require specific parameters from the page and don’t work unless you manually configure them. The default value is false.</td>
</tr>
<tr>
<td>isProgressiveRenderingEnabled</td>
<td>boolean</td>
<td>Indicates whether the display order of page components is prioritized (true) or not (false). The default value is false.</td>
</tr>
<tr>
<td>loginAppPageId</td>
<td>UUID</td>
<td>Represents the ID of the login page.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Not supported if the active template for the community doesn’t support login (such as Help Center).</td>
</tr>
<tr>
<td>mainAppPageId</td>
<td>UUID</td>
<td>Required. Represents the ID of the main page.</td>
</tr>
<tr>
<td>preferredDomainId</td>
<td>string</td>
<td>Represents the domain to use for indexing a community’s pages. Improves search engine results.</td>
</tr>
<tr>
<td>selfRegistrationRouteId</td>
<td>UUID</td>
<td>Represents the ID of the login route to use for self-registration.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Not supported if the active template for the community doesn’t support login (such as Help Center).</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Represents the component type. The only supported value is <code>site</code>.</td>
</tr>
</tbody>
</table>

```json
{
  "isAvailableToGuests" : false,
  "isFilteredComponentsView" : false,
  "mainAppPageId" : "df9907cb-6e68-4ca1-8bb2-51173ca5374e",
  "loginAppPageId" : "58e9939a-84b2-498d-bbc5-7a89d89087fa",
  "selfRegistrationRouteId" : "ad5c8bf1-297f-4ad3-b47c-0e35d85f10ef",
  "forgotPasswordRouteId" : "e3139f6f-44d8-4eec-be9d-3609ce063039",
  "isProgressiveRenderingEnabled" : false,
  "preferredDomainId" : "none",
  "type" : "site"
}
```

languages.json File Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultCode</td>
<td>string</td>
<td>Required. Represents the base language code plus the country code where used.</td>
</tr>
<tr>
<td>defaultLabel</td>
<td>string</td>
<td>Required. Defines the display label for the language.</td>
</tr>
<tr>
<td>id</td>
<td>UUID</td>
<td>Represents the component’s GUID.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Represents the component type. The only supported value is <code>languageContainer</code>.</td>
</tr>
</tbody>
</table>

There is one section per supported language as a container in languages.json

language container

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| countryCode    | string  | Represents the country code of the selected language. This string can be empty. It applies only when the selected language has variations depending on the country, like Arabic (Algeria) and Arabic (Bahrain). In this case, use countryCode to distinguish between them. For example:
```
{
  "languageCode" : "ar",
  "CountryCode" : "DZ",
  "Label" : "Arabic (Algeria) (DZ)",
},
```

| "Code" : "ar",
| "CountryCode" : "BH", |
The page file represents single-page applications in the community. One file per page, named `page_name.json`.

**Note:** Each Lightning community is actually a single-page application, which is a web app that loads a single HTML page. Single-page applications use multiple views on page 462 to update the page dynamically as the user interacts with it.
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmsSettings</td>
<td>map</td>
<td>Settings for the CMS Connect header and footer. Allowed values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• headerName</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• headerUrl</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• headerPersonalization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• footerName</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• footerUrl</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• footerPersonalization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Both source and target org must have the CMSConnect and CMSPersonalization org perms enabled for settings to be retrieved.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>🔄 Note: Not supported if the active template for the community doesn't support login (such as Help Center).</td>
</tr>
<tr>
<td>currentThemeId</td>
<td>UUID</td>
<td>Required. Represents the UUID of the site's current theme. This field is available only for the site's main page.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>🔄 Note: Not supported if the active template for the community doesn't support login (such as Help Center).</td>
</tr>
<tr>
<td>headMarkup</td>
<td>string</td>
<td>Required. Allows the addition of custom markup to the community’s main page. Similar to using Community Builder &gt; Setting &gt; Advanced &gt; Head Markup. See Salesforce Help for markup guidance.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>🔄 Note: Not supported if the active template for the community doesn't support login (such as Help Center).</td>
</tr>
<tr>
<td>id</td>
<td>UUID</td>
<td>Required. Represents the component’s GUID.</td>
</tr>
<tr>
<td>isRelaxedCSPLevel</td>
<td>boolean</td>
<td>Controls the ability to run scripts and script access to third-party hosts. The default is false.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>🔄 Note: Not supported if the active template for the community doesn't support login (such as Help Center).</td>
</tr>
</tbody>
</table>
Metadata Types

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Represents the name of the page.</td>
</tr>
<tr>
<td>templateName</td>
<td>string</td>
<td>Required. Represents the unique developer name of the template page. Allowed values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Help Center</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Customer Account Portal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Community Template</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Customer Service</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Build Your Own</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Partner Central</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Custom template (name which was used during the template creation)</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Required. Represents the component type. The only supported value is appPage.</td>
</tr>
</tbody>
</table>

```
{
    "headMarkup" : null,
    "isRelaxedCSPLevel" : false,
    "templateName" : "CPT Community Template",
    "cmsSettings" : { },
    "currentThemeId" : "ff52089c-6ad9-4dd9-b5b5-251d4a117ce3",
    "label" : "main",
    "id" : "df9907cb-6e68-4ca1-8bb2-51173ca5374e",
    "type" : "appPage"
}
```

routes Folder

The routes folder contains one JSON file per page, named `<page_name>.json`. `<page_name>.json`

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appPageId</td>
<td>UUID</td>
<td>Required. Represents the Single Page Application (SPA) page for the route. It points to either main.json or login.json.</td>
</tr>
<tr>
<td>id</td>
<td>UUID</td>
<td>Required. Represents the component GUID. Inherited from the component.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Represents the name of the route. Inherited from the component.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>objectApiName</td>
<td>string</td>
<td>Required. The name of the custom object API. (Not available for standard objects.)</td>
</tr>
<tr>
<td>pageAccess</td>
<td>string</td>
<td>Required. Identifies the status of a route as public or private. When set to the default value UseParent, the status of the site determines the status of the route. Not editable from the user interface for routes that are always private. Valid values are UseParent, Public, and RequireLogin.</td>
</tr>
<tr>
<td>routeType</td>
<td>string</td>
<td>Required. Identifies the type of route. Value is unique among all routes that share the same RoutePage.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Required. Represents the component type. The only supported value is route.</td>
</tr>
<tr>
<td>urlPrefix</td>
<td>string</td>
<td>Required. Represents the base URL for the route.</td>
</tr>
</tbody>
</table>

```
{
  "urlPrefix" : "",
  "appPageId" : "b5fe94e2-071f-47b2-b76d-427a624cb407",
  "routeType" : "home",
  "pageAccess" : "UseParent",
  "label" : "Home",
  "id" : "c7263124-7bc4-4147-a39a-25fe7e305b98",
  "type" : "route"
}
```

**themes Folder**

The themes folder contains one JSON file per theme named `theme_name.json`.

**theme_name.json**

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>activeBrandingSetId</td>
<td>UUID</td>
<td>The id of the branding set currently in use. The branding set's definitionName must match the theme's brandingSetReference.</td>
</tr>
<tr>
<td>customCSS</td>
<td>string</td>
<td>Custom CSS for pages created from Lightning community template.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. The unique developer name of the theme. Most themes derive their names.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. Represents the component’s GUID.</td>
</tr>
<tr>
<td>id</td>
<td>UUID</td>
<td>Required. Represents the component’s GUID.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Represents the name of the theme.</td>
</tr>
<tr>
<td>layouts</td>
<td>map</td>
<td>Required. Maps ThemeLayoutType to UUID, and contains the definition of the ThemeLayout. Login and Inner theme layouts are always required.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Required. Represents the component type. The only supported value is theme.</td>
</tr>
</tbody>
</table>

```
{
  "developerName" : "cpt",
  "layouts" : {
    "Login" : "12162c3e-06ac-43a9-adc7-db36ae5140b0",
    "Inner" : "c09d58be-0622-4fc4-806a-ed34174929f9"
  },
  "customCSS" : "",
  "activeBrandingSetId" : "283407c3-5938-4a6b-b97f-621cda6968c8",
  "label" : "Customer Account Portal",
  "id" : "ff52089c-6ad9-4dd9-b5b5-251d4a117ce3",
  "type" : "theme",
  "views" : [ {
    "componentName" : "salesforceIdentity:loginBody2",
    "label" : "Login",
    "id" : "12162c3e-06ac-43a9-adc7-db36ae5140b0",
    "type" : "view",
    "regions" : [ {
      "regionName" : "header",
      "id" : "f8354922-1ff2-495d-9d89-0a51943af2b0",
      "type" : "region",
      "components" : [ ]
    } ]
  } ]
}
```
Note: Views can be children of a theme. These children are structured the same as views on page 462 in the views folder.

views Folder

The `views` folder contains several JSON files that each define a view. Each Lightning community is built from single-page applications, which are web apps that load a single HTML page. Single-page applications consist of multiple views that update the page dynamically as the user interacts with it.

A view is made up of regions that contain other regions or components in the rendered page for the user. Within the `views` folder there is one file per view, named `view_name.json`.

Note: Single-page applications in your community are defined in the page files of the `config` folder on page 454.

`view_name.json`

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appPageId</td>
<td>UUID</td>
<td>Required. Single Page Application (SPA) page ID of the view. It points to either main.json or login.json.</td>
</tr>
<tr>
<td>componentName</td>
<td>string</td>
<td>Required. The FQN of the layout component. The component must implement <code>forceCommunity:layout</code> or, for theme layouts, <code>forceCommunity:themeLayout</code></td>
</tr>
<tr>
<td>id</td>
<td>UUID</td>
<td>Required. Represents the GUID of the component.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The name that appears in Community Builder &gt; Settings &gt; Theme &gt; Configure.</td>
</tr>
<tr>
<td>themeLayoutType</td>
<td>string</td>
<td>Theme layout type of the view (exposed only for views).</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Required. Represents the type of the component. The only supported value is view.</td>
</tr>
<tr>
<td>viewType</td>
<td>string</td>
<td>Required. Matches <code>routeType</code> for the route.</td>
</tr>
</tbody>
</table>

There are one or more regions as a container in each `<view_name>.json`.

`region container`

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>UUID</td>
<td>Required. Represents the component GUID.</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>regionLabel</td>
<td>string</td>
<td>Specifies region labels for tabs.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
<td>This property is present only for tab regions that are children of a component.</td>
</tr>
<tr>
<td>regionName</td>
<td>string</td>
<td>Required. Matches the design attribute in the design file of the layout component.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Required. Represents the component type. The only supported value is region.</td>
</tr>
</tbody>
</table>

There are one or more components as a container in the region section of each `<view_name>.json` component container

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentAttributes</td>
<td>HashMap</td>
<td>Required. The design attribute values of the component.</td>
</tr>
<tr>
<td>componentName</td>
<td>string</td>
<td>Required. The FQN of the component. Only components that can be used in the component panel in Community Builder can be used in this field.</td>
</tr>
<tr>
<td>id</td>
<td>UUID</td>
<td>Required. Represents the component GUID.</td>
</tr>
<tr>
<td>renderPriority</td>
<td>enums.priority</td>
<td>Sets priority value for progressive rendering of the component. Possible Values: HIGHEST, HIGH, NEUTRAL</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
<td>Only evaluated if the community has progressive rendering turned on in Community Builder &gt; Settings &gt; Advanced.</td>
</tr>
<tr>
<td>renditionMap</td>
<td>HashMap</td>
<td>Map of different rendition keys to UUIDs of RenditionComponents.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Required. Represents the component type. The only supported value is component.</td>
</tr>
</tbody>
</table>

Each component can have a rendition container in each `<view_name>.json` rendition container

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>UUID</td>
<td>Required. Represents the component GUID.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>renditionValue</td>
<td>map</td>
<td>Map of different variations of a component, such as different languages of text.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Required. Represents the component type. The only supported value is renditionComponent.</td>
</tr>
</tbody>
</table>

```json
{
    "themeLayoutType": "Inner",
    "viewType": "account-management",
    "appPageId": "df9907cb-6e68-4ca1-8bb2-51173ca5374e",
    "componentName": "siteforce:sldsOneColLayout",
    "label": "Account Management",
    "id": "9ca8fa47-8e87-4915-a6f7-c2d8d37f3076",
    "type": "view",
    "regions": [
        {
            "regionName": "content",
            "id": "969ada98-7d72-4e45-8a10-7db51fae247c",
            "type": "region",
            "components": [
                {
                    "componentName": "forceCommunity:tabset",
                    "componentAttributes": {
                        "tabsetConfig": {
                            "UUID": "4711850e-ffdc-4375-a45e-f716bcdbbb1c",
                            "activeTab": "tab1",
                            "useOverflowMenu": false,
                            "tabs": [
                                {
                                    "UUID": "f520-4010-861c-57b930a3daab",
                                    "attributes": {
                                        "מכשיר": "יתרונות"}
                                }
                            ]
                        }
                    }
                }
            ],
            "renderPriority": "NEUTRAL",
            "renditionMap": {},
            "id": "4711850e-ffdc-4375-a45e-f716bcdbbb1c",
            "type": "component",
            "renditions": [
                {
                    "renditionValue": {
                        "LumenInstanceAttributes": {
                            "richTextValue": "<p>new text</p>"
                        }
                    },
                    "id": "9d8878df-f520-4010-861c-57b930a3daab",
                    "type": "renditionComponent"
                }
            ]
        }
    ]
}
```
Declarative Metadata Sample Definition

Here's an example of an ExperienceBundle declaration. For individual folder and file examples for the bundled code, see brandingSets on page 453, config on page 454, routes on page 459, themes on page 460, and views on page 462.

```xml
<xsd:complexType name="ExperienceBundle">
  <xsd:complexContent>
    <xsd:extension base="tns:Metadata">
      <xsd:sequence>
        <xsd:element name="experienceResources" minOccurs="0" type="tns:ExperienceResources"/>
        <xsd:element name="label" type="xsd:string"/>
        <xsd:element name="type" type="tns:SiteType"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="ExperienceResources">
  <xsd:sequence>
    <xsd:element name="experienceResource" minOccurs="0" maxOccurs="unbounded" type="tns:ExperienceResource"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ExperienceResource">
  <xsd:sequence>
    <xsd:element name="fileName" type="xsd:string"/>
    <xsd:element name="format" type="xsd:string"/>
    <xsd:element name="source" minOccurs="0" type="xsd:base64Binary"/>
    <xsd:element name="type" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
```

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- ExperienceBundleSettings (Developer Preview)
- Developer Guide: ExperienceBundle for Lightning Communities (Developer Preview)

ExternalDataSource

Represents the metadata associated with an external data source. Create external data sources to manage connection details for integration with data and content that are stored outside your Salesforce org.

This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

ExternalDataSource components are stored in the dataSources directory of the corresponding package directory. ExternalDataSource components have the suffix .dataSource, and the prefix is the name of the external data source.
### Version

ExternalDataSource components are available in API version 28.0 and later.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authProvider</td>
<td>string</td>
<td>The authentication provider represented by the AuthProvider component.</td>
</tr>
<tr>
<td>certificate</td>
<td>string</td>
<td>If you specify a certificate, your Salesforce org supplies it when establishing each two-way SSL connection with the external system. The certificate is used for digital signatures, which verify that requests are coming from your Salesforce org.</td>
</tr>
<tr>
<td>customConfiguration</td>
<td>string</td>
<td>A string of configuration parameters that are specific to the external data source's type.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• customConfiguration for Salesforce Connect—Cross-Org Adapter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• customConfiguration for Salesforce Connect—OData 2.0 or 4.0 Adapter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• customConfiguration for Salesforce Connect—Custom Adapter</td>
</tr>
<tr>
<td>customHttpHeader</td>
<td>CustomHttpHeader[]</td>
<td>Represents custom HTTP headers used with OData 2.0 or OData 4.0 connectors. Available in API version 43.0 or later.</td>
</tr>
<tr>
<td>endpoint</td>
<td>string</td>
<td>The URL of the external system, or if that URL is defined in a named credential, the named credential URL. Corresponds to URL in the user interface.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A named credential URL contains the scheme <code>callout:</code> , the name of the named credential, and an optional path. For example: <code>callout:My_Named_Credential/some_path</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can append a query string to a named credential URL. Use a question mark (?) as the separator between the named credential URL and the query string. For example: <code>callout:My_Named_Credential/some_path?format=json</code>.</td>
</tr>
<tr>
<td>isWritable</td>
<td>boolean</td>
<td>Lets the Lightning platform and users in this org create, update, and delete records for external objects associated with the external data source. The external object data is stored outside the org. By default, external objects are read only. Corresponds to Writable External Objects in the user interface.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Available in API version 35.0 and later. However, with the cross-org adapter for Salesforce Connect, you can set this field to <code>true</code> only in API version 39.0 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>A user-friendly name for the external data source. The label is displayed in the Salesforce user interface, such as in list views. Examples include Acme Team Marketing Site, or Acme SharePoint.</td>
</tr>
<tr>
<td>oauthRefreshToken</td>
<td>string</td>
<td>The OAuth refresh token. Used to obtain a new access token for an end user when a token expires.</td>
</tr>
<tr>
<td>oauthScope</td>
<td>string</td>
<td>Specifies the scope of permissions to request for the access token. Corresponds to Scope in the user interface.</td>
</tr>
<tr>
<td>oauthToken</td>
<td>string</td>
<td>The access token issued by the external system.</td>
</tr>
<tr>
<td>password</td>
<td>string</td>
<td>The password to be used by your org to access the external system. Ensure that the credentials you use have adequate privileges to access the external system, perform searches, return data, and return information about the external system’s metadata.</td>
</tr>
<tr>
<td>principalType</td>
<td>External PrincipalType</td>
<td>Determines whether you’re using one set or multiple sets of credentials to access the external system. Corresponds to Identity Type in the user interface. The valid values are: Anonymous, PerUser, NamedUser.</td>
</tr>
<tr>
<td>protocol</td>
<td>Authentication Protocol</td>
<td>The authentication protocol that’s required to access the external system. The valid values are: NoAuthentication, Oauth, Password. For cloud-based Files Connect external systems, select Oauth 2.0. For on-premises systems, select Password Authentication. For Simple URL data sources, select No Authentication.</td>
</tr>
<tr>
<td>repository</td>
<td>string</td>
<td>Used for SharePoint Online. If metadata is not accessible, use this field to create tables and default table fields.</td>
</tr>
<tr>
<td>type</td>
<td>ExternalDataSourceSourceType</td>
<td>For Salesforce Connect, specifies the adapter that connects to the external system. The valid values are: OData—OData 2.0 adapter, OData4—OData 4.0 adapter, SfdcOrg—cross-org adapter, ApexClassId—DataSource.Provider class that defines the custom adapter created via the Apex Connector Framework.</td>
</tr>
</tbody>
</table>
For Files Connect, specifies the data source type. The valid values are:

- **ContentHubSharepoint**—SharePoint 2010 or 2013
- **ContentHubSharepointOffice365**—SharePoint Online
- **ContentHubSharepointOneDrive**—OneDrive for Business
- **ContentHubGDrive**—Google Drive

If Chatter is enabled, you can also specify **SimpleURL** to access data hosted on a web server that doesn’t require authentication.

- **outgoingemail**—A data source used for sending an email through a quick action.

The **Identity** and **Wrapper** types are reserved for future use.

For the federated search external data source type, the valid value is **OpenSearch**.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>string</td>
<td>The username to be used by your org to access the external system. Ensure that the credentials you use have adequate privileges to access the external system, perform searches, return data, and return information about the external system’s metadata.</td>
</tr>
<tr>
<td>version</td>
<td>string</td>
<td>Reserved for future use.</td>
</tr>
</tbody>
</table>

**CustomHttpHeader**

Represents a custom HTTP header used with OData 2.0 or OData 4.0 connectors. Available in API version 43.0 or later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the custom HTTP header.</td>
</tr>
<tr>
<td>headerFieldName</td>
<td>string</td>
<td>Required. Name of the custom HTTP header field.</td>
</tr>
<tr>
<td>headerFieldValue</td>
<td>string</td>
<td>Required. Value of the custom HTTP header field derived from a formula.</td>
</tr>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Specifies whether the custom HTTP header field is active (true) or inactive (false).</td>
</tr>
</tbody>
</table>

**customConfiguration** for Salesforce Connect—Cross-Org Adapter

The following sample JSON-encoded configuration string defines parameters that apply when the external data source’s type is set to SfdcOrg.

```json
{"apiVersion":"32.0","environment":"CUSTOM","searchEnabled":"true","timeout":"120"}
```
The parameters correspond to these fields in the user interface:

- **apiVersion**—API Version
- **environment**—Connect to
- **searchEnabled**—Enable Search
- **timeout**—Connection Timeout

### customConfiguration for Salesforce Connect—OData 2.0 or 4.0 Adapter

The following JSON-encoded configuration string defines parameters that apply when the external data source's `type` is set to OData or OData4.

```
{"inlineCountEnabled":"true","csrfTokenName":"X-CSRF-Token",
"requestCompression":"false","pagination":"CLIENT",
"noIdMapping":"false","format":"ATOM",
"searchFunc":null,"compatibility":"DEFAULT",
"csrfTokenEnabled":"true","timeout":"120",
"searchEnabled":"true"}
```

The parameters correspond to these fields in the user interface.

- **compatibility**—Special Compatibility
- **csrfTokenEnabled**—CSRF Protection
- **csrfTokenName**—Anti-CSRF Token Name
- **format**—Format
- **inlineCountEnabled**—Request Row Counts
- **noIdMapping**—High Data Volume
- **pagination**—Server Driven Pagination
- **requestCompression**—Compress Requests
- **searchEnabled**—Enable Search
- **searchFunc**—Custom Query Option for Salesforce Search
- **timeout**—Connection Timeout

### customConfiguration for Salesforce Connect—Custom Adapter

The following sample JSON-encoded configuration string defines the parameter that applies when the external data source's `type` is set to the ID of a `DataSource.Provider` class.

```
{"noIdMapping":"false"}
```

The `noIdMapping` parameter corresponds to the `High Data Volume` field in the user interface.

### Declarative Metadata Sample Definition

The following is the definition of an external data source for Salesforce Connect—OData 2.0 or 4.0 adapter.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ExternalDataSource xmlns="http://soap.sforce.com/2006/04/metadata">
  <authProvider>FacebookAuth</authProvider>
</ExternalDataSource>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

ExternalServiceRegistration

Represents the External Service configuration for an org. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

ExternalServiceRegistration components have the suffix .externalServiceRegistration and are stored in the externalServiceRegistrations folder.

Version

ExternalServiceRegistration components are available in API version 39.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Required. The external service description defined when the service is created.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The service name as it appears on the External Services wizard.</td>
</tr>
<tr>
<td>namedCredential</td>
<td>string</td>
<td>Required. The reference of the named credential to be used for the service.</td>
</tr>
<tr>
<td>schema</td>
<td>string</td>
<td>Required. The content of the JSON schema in the Interagent format. Nillable.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example of an ExternalServiceRegistration component that references an external credit service.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ExternalServicesRegistration xmlns="http://soap.sforce.com/2016/11/metadata">
  <label>creditService</label>
  <namedCredential>AcmeCreditService</namedCredential>
  <schema>/schema</schema>
  <schemaType>InteragentHyperSchema</schemaType>
  <schemaUrl>http://interagent.github.io/interagent-hyper-schema</schemaUrl>
  <status>complete</status>
</ExternalServicesRegistration>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

FeatureParameterBoolean

Represents a boolean feature parameter in a packaging org that has access to the Feature Management App (FMA). Feature parameters let you drive app behavior and track activation metrics in subscriber orgs that install your package. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

FeatureParameterBoolean components have the suffix .featureParameterBoolean. The components are stored in the featureParameters folder, which contains components for all the feature parameter metadata types.

Version

FeatureParameterBoolean components are available in API version 41.0 and later.
Special Access Rules

Available only in packaging orgs that have access to the Feature Management App (FMA). For details, see Manage Features in the ISVforce Guide.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataFlowDirection</td>
<td>FeatureParameterDataFlowDirection</td>
<td>After a package containing the components is installed, indicates whether the feature parameter’s value is editable in your License Management Org (LMO) and read-only in your customer’s org or the other way around.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>The feature parameter name that appears in the user interface.</td>
</tr>
<tr>
<td>value</td>
<td>boolean</td>
<td>The default value for this feature parameter. You can reference this value in your code, just like you reference other values in a subscriber’s org.</td>
</tr>
</tbody>
</table>

FeatureParameterDataFlowDirection

Represents the direction of the data flow between your License Management Org (LMO) and the customer’s org.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FeatureParameterDataFlowDirection</td>
<td>string</td>
<td>After a package containing the components is installed, indicates whether the feature parameter’s value is editable in your License Management Org (LMO) and read-only in your customer’s org or the other way around. Valid values are: • LmoToSubscriber • SubscriberToLmo</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a FeatureParameterBoolean component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<FeatureParameterBoolean xmlns="http://soap.sforce.com/2006/04/metadata">
  <dataflowDirection>SubscriberToLmo</dataflowDirection>
  <masterLabel>Budget Tracking Enabled</masterLabel>
  <value>false</value>
</FeatureParameterBoolean>
```
The following is an example `package.xml` that references the previous definition (and the definitions for the other feature parameter types).

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>FeatureParameterBoolean</name>
  </types>
  <types>
    <members>*</members>
    <name>FeatureParameterDate</name>
  </types>
  <types>
    <members>*</members>
    <name>FeatureParameterInteger</name>
  </types>
  <version>41.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**FeatureParameterDate**

Represents a date feature parameter in a packaging org that has access to the Feature Management App (FMA). Feature parameters let you drive app behavior and track activation metrics in subscriber orgs that install your package. This type extends the `Metadata` metadata type and inherits its `fullName` field.

**File Suffix and Directory Location**

`FeatureParameterDate` components have the suffix `.featureParameterDate`. The components are stored in the `featureParameters` folder, which contains components for all the feature parameter metadata types.

**Version**

`FeatureParameterDate` components are available in API version 41.0 and later.

**Special Access Rules**

Available only in packaging orgs that have access to the Feature Management App (FMA). For details, see Manage Features in the ISVforce Guide.
## Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataFlowDirection</td>
<td>FeatureParameterDataFlowDirection</td>
<td>After a package containing the components is installed, indicates whether the feature parameter’s value is editable in your License Management Org (LMO) and read-only in your customer’s org or the other way around.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>The feature parameter name that appears in the user interface.</td>
</tr>
<tr>
<td>value</td>
<td>date</td>
<td>The default value for this feature parameter. You can reference this value in your code, just like you reference other values in a subscriber’s org.</td>
</tr>
</tbody>
</table>

### FeatureParameterDataFlowDirection

Represents the direction of the data flow between your License Management Org (LMO) and the customer’s org.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| FeatureParameterDataFlowDirection | string | After a package containing the components is installed, indicates whether the feature parameter’s value is editable in your License Management Org (LMO) and read-only in your customer’s org or the other way around. Valid values are:  
- LmoToSubscriber  
- SubscriberToLmo |

### Declarative Metadata Sample Definition

The following is an example of a FeatureParameterDate component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<FeatureParameterDate xmlns="http://soap.sforce.com/2006/04/metadata">
  <dataflowDirection>SubscriberToLmo</dataflowDirection>
  <masterLabel>Activation Date</masterLabel>
  <value>2017-10-23</value>
</FeatureParameterDate>
```

The following is an example package.xml that references the previous definition (and the definitions for the other feature parameter types).

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>FeatureParameterBoolean</name>
  </types>
</Package>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

FeatureParameterInteger

Represents an integer feature parameter in a packaging org that has access to the Feature Management App (FMA). Feature parameters let you drive app behavior and track activation metrics in subscriber orgs that install your package. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

FeatureParameterInteger components have the suffix .featureParameterInteger. The components are stored in the featureParameters folder, which contains components for all the feature parameter metadata types.

Version

FeatureParameterInteger components are available in API version 41.0 and later.

Special Access Rules

Available only in packaging orgs that have access to the Feature Management App (FMA). For details, see Manage Features in the ISVforce Guide.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataFlowDirection</td>
<td>FeatureParameterDataFlowDirection</td>
<td>After a package containing the components is installed, indicates whether the feature parameter’s value is editable in your License Management Org (LMO) and read-only in your customer’s org or the other way around.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>The feature parameter name that appears in the user interface.</td>
</tr>
</tbody>
</table>
The default value for this feature parameter. You can reference this value in your code, just like you reference other values in a subscriber’s org.

**FeatureParameterDataFlowDirection**

Represents the direction of the data flow between your License Management Org (LMO) and the customer’s org.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| FeatureParameterDataFlowDirection | string | After a package containing the components is installed, indicates whether the feature parameter’s value is editable in your License Management Org (LMO) and read-only in your customer’s org or the other way around. Valid values are:  
• LmoToSubscriber  
• SubscriberToLmo |

**Declarative Metadata Sample Definition**

The following is an example of a FeatureParameterInteger component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<FeatureParameterInteger xmlns="http://soap.sforce.com/2006/04/metadata">
  <dataflowDirection>SubscriberToLmo</dataflowDirection>
  <masterLabel>Current Project Count</masterLabel>
  <value>42</value>
</FeatureParameterInteger>
```

The following is an example `package.xml` that references the previous definition (and the definitions for the other feature parameter types).

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>FeatureParameterBoolean</name>
  </types>
  <types>
    <members>*</members>
    <name>FeatureParameterDate</name>
  </types>
  <types>
    <members>*</members>
    <name>FeatureParameterInteger</name>
  </types>
  <version>41.0</version>
</Package>
```
**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**FlexiPage**

Represents the metadata associated with a Lightning page. A Lightning page represents a customizable screen made up of regions containing Lightning components.

Note: A Lightning page region can contain up to 25 components.

This type extends the Metadata metadata type and inherits its `fullName` field.

Note: These pages are known as FlexiPages in the API, but are referred to as Lightning pages in the rest of the Salesforce documentation and UI.

Lightning pages are used in several places.

- In the Salesforce app, a Lightning page is the home page for an app that appears in the navigation menu.
- In Lightning Experience, Lightning pages can be used:
  - To customize the layout of record pages, the Salesforce Home page, and the Email Application pane in the Outlook and Gmail integrations.
  - As the home page for an app.
  - As the utility bar for a Lightning app.

For more information on Lightning pages, see Salesforce Help.

**File Suffix and Directory Location**

FlexiPage components have the suffix `.flexipage` and are stored in the `flexipages` folder.

**Version**

FlexiPage components are available in API version 29.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The optional description text of the Lightning page.</td>
</tr>
<tr>
<td>flexiPageRegions</td>
<td>FlexiPageRegion[]</td>
<td>The list of regions of a page.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The label for the Lightning page, which displays in Setup.</td>
</tr>
<tr>
<td>pageTemplate</td>
<td>string</td>
<td>Deprecated. Use this field in API versions 33.0 to 38.0 only. In later versions, use <code>template</code>. Required. The template associated with the Lightning page.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>parentFlexiPage</td>
<td>string</td>
<td>The name of the Lightning page that this page inherits behavior from.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 37.0 or later.</td>
</tr>
<tr>
<td>platformActionList</td>
<td>PlatformActionList</td>
<td>The list of all actions, and their order, that display on a Lightning app page. In the Salesforce mobile app, the actions appear in the action bar.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 34.0 and later.</td>
</tr>
<tr>
<td>quickActionList</td>
<td>QuickActionList</td>
<td>The list of quick actions associated with the Lightning page.</td>
</tr>
<tr>
<td>sobjectType</td>
<td>string</td>
<td>The object the Lightning page is associated with. For Lightning pages of type AppPage or HomePage, this field is null. After the value of this field is set, it can’t be changed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 37.0 or later.</td>
</tr>
<tr>
<td>template</td>
<td>FlexiPageTemplateInstance</td>
<td>Required. The template associated with the Lightning page.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 39.0 and later.</td>
</tr>
<tr>
<td>type</td>
<td>FlexiPageType (enumeration of type string)</td>
<td>Required. The type of a page. In API versions 32.0 through 36.0, this field can only have a value of AppPage. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AppPage—A Lightning page that is used as the home page for a custom app.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CommAppPage—A Lightning page that is used to represent a custom page, as created in the Community Builder, in Communities. This value is available in API version 37.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CommFlowPage—A Lightning page used to override a flow page, as created in the Community Builder, in Communities. This value is available in API version 45.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CommForgotPasswordPage—A Lightning page that’s used to override a forgot-password page, as created in Community Builder, in Communities. This value is available in API version 39.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CommGlobalSearchResultPage—A Lightning page used to override the global search result page, as created in Community Builder, in Communities. This value is available in API version 41.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CommFlowPage—An out-of-the-box flow page, as created in Community Builder, in Communities. This value is available in API version 45.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CommGlobalSearchResultPage—A Lightning page used to override the global search result page, as created in Community Builder, in Communities. This value is available in API version 41.0 and later.</td>
</tr>
</tbody>
</table>
FlexiPage Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommLoginPage</td>
<td>Field Type</td>
<td>A Lightning page that’s used to override the login page, as created in Community Builder, in Communities. This value is available in API version 39.0 and later.</td>
</tr>
<tr>
<td>CommObjectPage</td>
<td>Field Type</td>
<td>A Lightning page used to override an object page, as created in Community Builder, in Communities. This value is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>CommQuickActionCreatePage</td>
<td>Field Type</td>
<td>A Lightning page used to override the create record page, as created in Community Builder, in Communities. This value is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>CommRecordPage</td>
<td>Field Type</td>
<td>A Lightning page used to override a record page, as created in the Community Builder, in Communities. This value is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>CommRelatedListPage</td>
<td>Field Type</td>
<td>A Lightning page used to override a related list page, as created in the Community Builder, in Communities. This value is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>CommSearchResultPage</td>
<td>Field Type</td>
<td>A Lightning page used to override the search result page, as created in Community Builder, in Communities. This value is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>CommSelfRegisterPage</td>
<td>Field Type</td>
<td>A Lightning page used to override the self-registration page, as created in Community Builder, in Communities. This value is available in API version 39.0 and later.</td>
</tr>
<tr>
<td>CommThemeLayoutPage</td>
<td>Field Type</td>
<td>A Lightning page used to override a theme layout page, as created in the Community Builder, in Communities. This value is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>EmbeddedServicePage</td>
<td>Field Type</td>
<td>This value is available in API version 45.0 and later.</td>
</tr>
<tr>
<td>HomePage</td>
<td>Field Type</td>
<td>A Lightning page that is used to override the Home page in Lightning Experience. This value is available in API version 37.0 and later.</td>
</tr>
<tr>
<td>MailAppAppPage</td>
<td>Field Type</td>
<td>An email application pane used to override the default layout in the Outlook and Gmail integrations. This value is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>RecordPage</td>
<td>Field Type</td>
<td>A Lightning page used to override an object record page in Lightning Experience. This value is available in API version 37.0 and later.</td>
</tr>
</tbody>
</table>
### Metadata Types

#### FlexiPageRegion

FlexiPageRegion represents the properties of a region of a page. A region can contain a record list component or a recent items component that can be scoped to a set of entities.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appendable</td>
<td>RegionFlagStatus</td>
<td>This field is available in Communities in API 45.0 or later, but is reserved for future use for all other areas. Valid values are: disbedid, enabled. This field is assessed in combination with replaceable and prependable. If all the properties are set to enabled, the region is unlocked. If all the properties are set to disabled, the region is locked. If none of the properties are specified OR any of these three properties are missing, the region is unlocked. This field is available in API version 35.0 or later.</td>
</tr>
<tr>
<td>componentInstances</td>
<td>ComponentInstance[]</td>
<td>Properties and name of the component instance.</td>
</tr>
<tr>
<td>mode</td>
<td>FlexiPageRegionMode</td>
<td>This field is reserved for future use. Valid values are: Append, Prepend, Replace. This field is available in API version 35.0 or later.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name of the FlexiPage region.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| prependable | RegionFlagStatus (enumeration of type string) | This field is available in Communities in API 45.0 or later, but is reserved for future use for all other areas. Valid values are:  
- *disabled*  
- *enabled*  
This field is assessed in combination with appendable and replaceable.  
- If all the properties are set to *enabled*, the region is unlocked  
- If all the properties are set to *disabled*, the region is locked  
- If none of the properties are specified OR any of these three properties are missing, the region is unlocked. This field is available in API version 35.0 or later. |
| replaceable | RegionFlagStatus (enumeration of type string) | This field is available in Communities in API 45.0 or later, but is reserved for future use for all other areas. Valid values are:  
- *disabled*  
- *enabled*  
This field is assessed in combination with appendable and prependable.  
- If all the properties are set to *enabled*, the region is unlocked  
- If all the properties are set to *disabled*, the region is locked  
- If none of the properties are specified OR any of these three properties are missing, the region is unlocked. This field is available in API version 35.0 or later. |
| type | FlexiPageRegionType (enumeration of type string) | Required. The type of FlexiPage region. Valid values are:  
- *Background*—Represents a region for background utility items, which aren’t visible in the UI. Supported for utility bars only.  
- *Facet*  
- *Region*  
This field is available in API version 35.0 or later. |
ComponentInstance

Instance of a component in a page, such as a filter list.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentInstanceProperties</td>
<td>ComponentInstanceProperty[]</td>
<td>The value of a single property in a component instance.</td>
</tr>
<tr>
<td>componentName</td>
<td>string</td>
<td>Required. The name of a single instance of a component.</td>
</tr>
<tr>
<td>visibilityRule</td>
<td>UiFormulaRule</td>
<td>A set of one or more filters that define the conditions under which the component displays on the page. If the rule evaluates to true, the component displays on the page. If false, it doesn't display. If this field is null, the component displays by default. This field is available in API version 41.0 and later.</td>
</tr>
</tbody>
</table>

ComponentInstanceProperty

Value of a single property in a component instance.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the property, unique within the component instance. For Lightning components, this value is the <a href="">aura:attribute</a> as defined in the .cmp file.</td>
</tr>
</tbody>
</table>
| type       | ComponentInstancePropertyType    | If this field value is null, then the ComponentInstanceProperty values apply to the Lightning component. If this field value is decorator, then the ComponentInstanceProperty values apply to the component decorator for the Lightning component. The component decorator is a wrapper around a Lightning component. The decorator can apply more capabilities to the component when it renders on a specific page in Lightning Experience. For example, you can configure a component decorator around a component on the Lightning Experience utility bar to set the component's height or width when opened. The UtilityBar is the only page type that supports component decorators. Valid values are:  
• decorator  
This field is available in API version 38.0 or later. |
| value      | string                          | Reference or value of the property. When defining a Related List component, to use a parent record set the parentFieldApiName value to |
When you give a standard label to a tab in a Tabs component—such as Activity, Collaborate, or Details—and when the name field is set to title, the value field uses a system-defined value instead of the label. Here are some examples of the system-defined values:

- Standard.Tab.activity
- Standard.Tab.collaborate
- Standard.Tab.detail
- Standard.Tab.feed
- Standard.Tab.preview
- Standard.Tab.relatedLists

For example, let’s say you have a Lightning page that contains a tab with the standard label “Activity”. If you query the definition that page, you see the system-defined name of the tab, not the label, in value.

```xml
<componentInstances>
  <componentInstanceProperties>
    <name>title</name>
    <value>Standard.Tab.activity</value>
  </componentInstanceProperties>
  <componentName>flexipage:tab</componentName>
</componentInstances>
```

**UiFormulaRule**

A set of one or more filters that define the conditions under which a component displays on a Lightning page. For example, you could construct a filter that causes a rich text component on an opportunity page to display only when the Amount is greater than $1,000,000.

Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Specifies advanced filter conditions such as 1 AND 2.</td>
</tr>
<tr>
<td>criteria</td>
<td>UiFormulaCriterion[]</td>
<td>List of one or more filters that, when evaluated, determine component visibility.</td>
</tr>
</tbody>
</table>

**FlexiPageTemplateInstance**

FlexiPageTemplateInstance represents an instance of a Lightning page template.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of a single instance of a template.</td>
</tr>
</tbody>
</table>
**Field Name** | **Field Type** | **Description**
--- | --- | ---
properties | ComponentInstanceProperty[] | The value of a single property in a template instance. Valid only for CommThemeLayoutPage. Contains a name and value pair for each theme layout property associated with the page template. In Community Builder, the theme layout and its properties appear in the Theme area.

---

### PlatformActionList

PlatformActionList represents the list of actions, and their order, that display on a Lightning app page. Available in API version 34.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionListContext</td>
<td>PlatformActionListContext</td>
<td>Required. The context of the action list. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>- ActionDefinition—Reserved for future use.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Assistant</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- BannerPhoto</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Chatter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Dockable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- FeedElement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Flexipage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Global</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ListView</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ListViewDefinition</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ListViewRecord</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Lookup</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- MruList</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- MruRow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ObjectHomeChart</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Photo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Record</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- RecordEdit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>platformActionListItems</th>
<th>PlatformActionListItem[]</th>
<th>The actions in the PlatformActionList.</th>
</tr>
</thead>
<tbody>
<tr>
<td>relatedSourceEntity</td>
<td>string</td>
<td>When the ActionListContext is RelatedList or RelatedListRecord, this field represents the API name of the related list to which the action belongs.</td>
</tr>
</tbody>
</table>
PlatformActionListItem

PlatformActionListItem represents an action in the PlatformActionList. Available in API version 34.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>string</td>
<td>Required. The API name for the action in the list.</td>
</tr>
<tr>
<td>actionType</td>
<td>PlatformActionType (enumeration of type string)</td>
<td>Required. The type of action. Valid values: ActionLink—An indicator on a feed element that targets an API, a web page, or a file, represented by a button in the Salesforce Chatter feed UI. CustomButton—When clicked, opens a URL or a Visualforce page in a window or executes JavaScript. InvocableAction ProductivityAction—Productivity actions are predefined and attached to a limited set of objects. Productivity actions include Send Email, Call, Map, View Website, and Read News. Except for the Call action, you can’t edit or delete productivity actions. QuickAction—A global or object-specific action. StandardButton—A predefined Salesforce button such as New, Edit, and Delete.</td>
</tr>
<tr>
<td>sortOrder</td>
<td>int</td>
<td>Required. The placement of the action in the list.</td>
</tr>
<tr>
<td>subtype</td>
<td>string</td>
<td>The subtype of the action. For quick actions, the subtype is QuickActionType. For custom buttons, the subtype is WebLinkTypeEnum. For action links, subtypes are Api, ApiAsync, Download, and Ui. Standard buttons and productivity actions have no subtype.</td>
</tr>
</tbody>
</table>

UiFormulaCriterion

A single filter that when evaluated, helps define component visibility on a Lightning page. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>leftValue</td>
<td>string</td>
<td>Required. The field upon which the filter is based. For example, AMOUNT.</td>
</tr>
<tr>
<td>operator</td>
<td>string</td>
<td>Required. Defines the operator used to filter the data. Valid values: CONTAINS EQUAL NE—not equal GT—greater than GE—greater than or equal</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
rightValue | string | The value by which you want to evaluate the component’s visibility. For example, 1000000.

You can use these expressions in the `leftValue` field when setting filters for component visibility.

- `{!$Client.FormFactor}`—Use this expression to control component visibility based on the device the page is being rendered on. Valid values are Small (phone), Medium (tablet), and Large (Lightning Experience desktop). Supported for app pages only.
- `{!$Permission.CustomPermission.permissionName}`—Use this expression to control component visibility based on the custom permissions of the user viewing the Lightning page. Supported for app, Home, and record pages only.
- `{!$Permission.StandardPermission.permissionName}`—Use this expression to control component visibility based on the standard permissions of the user viewing the Lightning page. Supported for app, Home, and record pages only.
- `{!Record.field}`—Supported for record pages only.
- `{!$User.field}`—Supported for app, Home, and record pages only.

For example, to display a component only when it renders on a phone, add this filter: `{!$Client.FormFactor} EQUAL "SMALL". Or, to display a component only to the System Administrator, use `{!$User.Profile.Name} EQUAL "System Administrator".

Expressions in component visibility rules can span no more than five fields. For example, `{!Record.Account.Owner.Manager.Manager.Manager.LastName}` has six spans and therefore isn’t supported.

### Declarative Metadata Sample Definition

Here’s a sample XML FlexiPage component definition for a custom opportunity record page. It includes a tab set and a rich text component with visibility rules assigned to it.

Note: As a community page, three initial regions in the definition show the header region as locked, the content region as unlocked, and the footer region as unlocked.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<FlexiPage xmlns="http://soap.sforce.com/2006/04/metadata">
  <componentInstances>
    <componentInstanceProperties>
      <name>collapsed</name>
      <value>false</value>
    </componentInstanceProperties>
    <componentInstanceProperties>
      <name>numVisibleActions</name>
      <value>3</value>
    </componentInstanceProperties>
    <componentInstanceProperties>
      <componentName>force:highlightsPanel</componentName>
    </componentInstanceProperties>
  </componentInstances>
  <flexiPageRegions>
    <appendable>disabled</appendable>
    <name>header</name>
```

486
<name>title</name>
  <value>Standard.Tab.detail</value>
</componentInstanceProperties>
<componentName>flexipage:tab</componentName>
</componentInstances>

<componentInstances>
  <componentInstanceProperties>
    <name>body</name>
    <value>facet-4bd6f038a-4568-4c4e-b220-b5ec47f656e9</value>
  </componentInstanceProperties>
  <componentInstanceProperties>
    <name>title</name>
    <value>Standard.Tab.activity</value>
  </componentInstanceProperties>
  <componentName>flexipage:tab</componentName>
</componentInstances>

<componentInstances>
  <componentInstanceProperties>
    <name>body</name>
    <value>facet-8d5ecfff-4590-49c0-a09e-3df84179f642</value>
  </componentInstanceProperties>
  <componentInstanceProperties>
    <name>title</name>
    <value>Recent Items</value>
  </componentInstanceProperties>
  <componentName>flexipage:tab</componentName>
</componentInstances>

<componentInstances>
  <componentInstanceProperties>
    <name>active</name>
    <value>true</value>
  </componentInstanceProperties>
  <componentInstanceProperties>
    <name>body</name>
    <value>facet-551d8770-5093-4222-927e-9138061ebbbf</value>
  </componentInstanceProperties>
  <componentInstanceProperties>
    <name>title</name>
    <value>Standard.Tab.relatedLists</value>
  </componentInstanceProperties>
  <componentName>flexipage:tab</componentName>
</componentInstances>

<name>facet-f433d4c6-94e6-4de4-90fc-83501abf3536</name>
<type>Facet</type>
</flexiPageRegions>

<flexiPageRegions>
  <componentInstances>
    <componentInstanceProperties>
      <name>tabs</name>
      <value>facet-f433d4c6-94e6-4de4-90fc-83501abf3536</value>
    </componentInstanceProperties>
    <componentName>flexipage:tabset</componentName>
  </componentInstances>
  <name>main</name>
A million dollar opportunity closed! Oh yeah!
And, here's the sample package.xml file that references the FlexiPage component definition:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <fullName>New Opportunity Page Package</fullName>
  <description>Contains an opportunity page with a rich text component that has rules assigned</description>
  <types>
    <members>New_Opportunity_Page</members>
    <name>FlexiPage</name>
  </types>
  <version>41.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**Flow**

Represents the metadata associated with a flow. With Flow, you can create an application that navigates users through a series of screens to query and update records in the database. You can also execute logic and provide branching capability based on user input to build dynamic applications.

For information about the corresponding UI-based flow building tool, see “Flow Builder” in Salesforce Help.

When using Metadata API to work with flows, consider that:

- You can’t use Metadata API to access a flow installed from a managed package, unless the flow is a template.
- Spaces in flow file names can cause errors at deployment. Heading and trailing spaces are allowed, but are trimmed during deployment.
- When using Metadata API, you can deploy changes to active flows in specific situations.
  - The org isn’t a production org (for example, a scratch org or sandbox). The active flow is the latest version.
  - The org is a production org that enabled the Deploy processes and flows as active preference. The active flow is the latest version.

After you deploy changes to an active flow, the flow’s detail page displays a new flow version that’s active. The new version includes your changes. For example, version 3 of myflow is active and the latest version. After you change and deploy version 3, the myflow’s detail page displays version 4 as active.

- You can delete a flow version, as long as it:
  - Isn’t active
  - Doesn’t have any paused interviews

If the flow version has paused interviews, wait for those interviews to resume and finish, or delete them.

**Warning:** Don’t edit the metadata of retrieved Process Builder processes (Flow components whose processType is Workflow or InvocableProcess.) If you deploy process metadata that you’ve edited, you might not be able to open the process in the target org.
Declarative Metadata File Suffix and Directory Location

Flows are stored in the `Flow` directory of the corresponding package directory. The file name matches the flow's unique full name, and the extension is `.flow`.

Version

The flow Metadata API is available in API version 24.0 and later.

Flow

This metadata type represents a valid definition of a flow. This type extends the `Metadata` metadata type and inherits its `fullName` field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionCalls</td>
<td><code>FlowActionCall[]</code></td>
<td>An array of nodes that define calls to actions. This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>apexPluginCalls</td>
<td><code>FlowApexPluginCall[]</code></td>
<td>An array of nodes that define calls to Apex plug-ins.</td>
</tr>
<tr>
<td>assignments</td>
<td><code>FlowAssignment[]</code></td>
<td>An array of assignment nodes.</td>
</tr>
<tr>
<td>choices</td>
<td><code>FlowChoice[]</code></td>
<td>An array of static choice options.</td>
</tr>
<tr>
<td>constants</td>
<td><code>FlowConstant[]</code></td>
<td>An array of constants.</td>
</tr>
<tr>
<td>decisions</td>
<td><code>FlowDecision[]</code></td>
<td>An array of decision nodes.</td>
</tr>
<tr>
<td>description</td>
<td><code>string</code></td>
<td>Description of the flow.</td>
</tr>
<tr>
<td>dynamicChoiceSets</td>
<td><code>FlowDynamicChoiceSet[]</code></td>
<td>An array that constructs a set of choice options based on a database lookup.</td>
</tr>
<tr>
<td>formulas</td>
<td><code>FlowFormula[]</code></td>
<td>An array of formulas.</td>
</tr>
<tr>
<td>fullName</td>
<td><code>string</code></td>
<td>Required; inherited from the Metadata component. Name of the file in Metadata API. A unique name for the flow that contains only underscores and alphanumeric characters. The name must be unique across the org, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. To deploy or retrieve a version, you can specify the version number. For example, <code>sampleFlow-3</code> specifies version 3 of the flow whose unique name is sampleFlow. If you don't specify a version number, the flow is the latest version. In API version 43.0 and earlier, this field included the version number. In API version 44 and later, this field no longer includes the version number.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>interviewLabel</td>
<td>string</td>
<td>Label for the interview. This label helps users and administrators differentiate interviews from the same flow. In the user interface, this label appears in thePaused Flow Interviews component on the user’s Home tab and in the Paused and Waiting Interviews list on the flow management page.</td>
</tr>
<tr>
<td>isTemplate</td>
<td>boolean</td>
<td>Indicates whether the process or flow is a template. When installed from managed packages, processes and flows can’t be viewed or cloned by subscribers because of intellectual property (IP) protection. But when those processes and flows are templates, subscribers can open them in a builder, clone them, and customize the clones. Available in API version 45.0 and later. Default: false</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Label for the flow.</td>
</tr>
<tr>
<td>loops</td>
<td>FlowLoop[]</td>
<td>An array of nodes for iterating through collections. This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>processMetadataValues</td>
<td>FlowMetadataValue[]</td>
<td>Metadata values for the flow. This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>processType</td>
<td>FlowProcessType (enumeration of type string)</td>
<td>The type of the flow, as determined by the active version (or latest version, if there’s no active version). Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Appointments—A flow for Lightning Scheduler. This value is available in API version 44.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AutoLaunchedFlow—A flow that doesn’t require user interaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ContactRequestFlow—A flow that lets customers request that customer support get back to them. This flow is used to create contact request records. This value is available in API version 45.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CustomEvent—A process that is invoked when it receives a platform event message. In the UI, it’s an event process. This value is available in API version 41.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FieldServiceMobile—A flow for the Field Service Lighting mobile app. This value is available in API version 39.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FieldServiceWeb—A flow for embedded Appointment Booking. Its UI label is Field Service Embedded Flow. This value is available in API version 41.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Flow—A flow that requires user interaction because it contains one or more screens or local actions, choices, or dynamic choices. In the UI and Salesforce Help, it’s a screen flow. Screen flows can be launched from the UI, such as with a flow action, Lightning page, or web tab.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>recordCreates</td>
<td>FlowRecordCreate[]</td>
<td>An array of nodes for creating records in the database.</td>
</tr>
<tr>
<td>recordDeletes</td>
<td>FlowRecordDelete[]</td>
<td>An array of nodes for deleting records in the database.</td>
</tr>
<tr>
<td>recordLookups</td>
<td>FlowRecordLookup[]</td>
<td>An array of nodes for looking up records in the database.</td>
</tr>
<tr>
<td>recordUpdates</td>
<td>FlowRecordUpdate[]</td>
<td>An array of nodes for updating records in the database.</td>
</tr>
<tr>
<td>screens</td>
<td>FlowScreen[]</td>
<td>An array of screen nodes.</td>
</tr>
<tr>
<td>stages</td>
<td>FlowStage[]</td>
<td>An array of stages that you can use throughout the flow. This field is available in API version 42.0 and later.</td>
</tr>
<tr>
<td>startElementReference</td>
<td>string</td>
<td>Specifies which node or element is the starting point in the flow.</td>
</tr>
<tr>
<td>status</td>
<td>FlowVersionStatus (enumeration of type string)</td>
<td>The activation status of the flow. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>• Active</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Draft—in the UI, this status appears as Inactive.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Obsolete—in the UI, this status appears as Inactive.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• InvalidDraft—in the UI, this status appears as Draft.</td>
<td></td>
</tr>
</tbody>
</table>

Across flow versions, you can change the type only from `Flow` to `AutolaunchedFlow` or vice versa.

These values are reserved for future use.

- ActionCadenceFlow
- ActionPlan
- CheckoutFlow
- DigitalForm
- JourneyBuilderIntegration
- LoginFlow
- OrchestrationFlow
- TransactionSecurityFlow
- UserProvisioningFlow

This field is available in API version 31.0 and later.
FlowActionCall

Defines a call to an action from the flow. It extends FlowNode.

Available in API version 31.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>string</td>
<td>Required. Name for the action. Must be unique across actions with the same actionType.</td>
</tr>
<tr>
<td>actionType</td>
<td>InvocableActionType (enumeration of type string)</td>
<td>Required. The action type. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• activateSessionPermSet—activates a session-based permission set for the running user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• addMessageToQuipChat—adds a Quip message to an existing chat room. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• addMessageToQuipDocument—adds a Quip message to an existing Quip document, spreadsheet, or slide. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• addQuipDocumentToFolder—adds an existing Quip document, spreadsheet, or slide to an existing folder. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• addUsersToQuipDocument—adds users, identified by their email addresses, to an existing Quip document, spreadsheet, or slide. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• addUsersToQuipChat—adds users, identified by their email addresses, to an existing Quip chat room. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• attachQuipDocumentToRecord—attaches a Quip document, spreadsheet, or slide to a Salesforce record. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• apex—invokes an Apex method that has the @invocableMethod annotation</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• archiveKnowledgeArticles</td>
<td>archiveKnowledgeArticles</td>
<td>archives a list of published Knowledge articles. Available in API version 45.0 and later.</td>
</tr>
<tr>
<td>• assignKnowledgeArticles</td>
<td>assignKnowledgeArticles</td>
<td>mass assigns Knowledge articles from article list views. Available in API version 44.0 and later.</td>
</tr>
<tr>
<td>• chatterPost</td>
<td>chatterPost</td>
<td>posts to Chatter</td>
</tr>
<tr>
<td>• choosePricebook</td>
<td>choosePricebook</td>
<td>selects a price book</td>
</tr>
<tr>
<td>• contactRequestAction</td>
<td>contactRequestAction</td>
<td>creates a contact request record. Available in API version 45.0 and later.</td>
</tr>
<tr>
<td>• component</td>
<td>component</td>
<td>invokes the Lightning component that implements the lightning:availableForFlowActions interface and that is referenced by actionName. Available in API version 43.0 and later.</td>
</tr>
<tr>
<td>• contentWorkspaceEnableFolders</td>
<td>contentWorkspaceEnableFolders</td>
<td>enables folders in a library</td>
</tr>
<tr>
<td>• copyQuipDocument</td>
<td>copyQuipDocument</td>
<td>creates a copy of an existing Quip document, spreadsheet, or slide and gives it a new title. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>• createDraftFromOnlineKnowledgeArticle</td>
<td>createDraftFromOnlineKnowledgeArticle</td>
<td>creates a draft from a published Knowledge article. Available in API version 45.0 and later.</td>
</tr>
<tr>
<td>• createQuipChat</td>
<td>createQuipChat</td>
<td>creates a Quip chat room. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>• createQuipDocument</td>
<td>createQuipDocument</td>
<td>creates a Quip document, spreadsheet, or slide. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>• createQuipFolder</td>
<td>createQuipFolder</td>
<td>creates a Quip folder. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>• customNotificationAction</td>
<td>customNotificationAction</td>
<td>sends a custom notification. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>• deactivateSessionPermSet</td>
<td>deactivateSessionPermSet</td>
<td>deactivates a session-based permission set for the running user.</td>
</tr>
<tr>
<td>• deleteKnowledgeArticle</td>
<td>deleteKnowledgeArticle</td>
<td>deletes a draft version (translation or master-language) or an entire archived Knowledge article. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>• editQuipDocument</td>
<td>editQuipDocument</td>
<td>modifies the contents of an existing Quip document, spreadsheet, or slide. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>• emailAlert</td>
<td>emailAlert</td>
<td>sends an email by referencing a workflow email alert.</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
emailSimple | string | sends an email by using flow resources.
externalService | string | invokes an External Service action (making an HTTP request to a remote system), made available by an External Service schema registered through Setup. Available in API version 46.0 and later.
flow | string | invokes an autolaunched flow. This action type isn’t available for flows with a processType of “Flow” or “AutolaunchedFlow”. To invoke an autolaunched flow from one of those types, use FlowSubflow. Available in API version 32.0 and later.
quickAction | string | invokes a QuickAction.
publishKnowledgeArticles | string | mass publishes Knowledge articles from article list views. Available in API version 44.0 and later.
restoreKnowledgeArticleVersion | string | restores an archived version of a Knowledge article. Available in API version 45.0 and later.
submitKnowledgeArticleForTranslation | string | submits a published or draft Knowledge article for translation. Available in API version 46.0 and later.
submit | string | submits a record for approval.
thanks | string | These values are reserved for future use.
metricRefresh | string |

| connector | FlowConnector | Specifies which node to execute after this action call. |
| faultConnector | FlowConnector | Specifies which node to execute if the action call results in an error. |
| inputParameters | FlowActionCallInputParameter[] | An array of input parameters from the flow to the action. |
| outputParameters | FlowActionCallOutputParameter[] | An array of output parameters from the action to the flow. |

**FlowActionCallInputParameter**

Defines an input parameter from the flow to the action. It extends FlowBaseElement and inherits all its fields. Available in API version 31.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the input parameter.</td>
</tr>
<tr>
<td>value</td>
<td>FlowElementReferenceOrValue</td>
<td>Defines the value of the input parameter.</td>
</tr>
</tbody>
</table>
FlowActionCallOutputParameter

Defines an output parameter from the action to the flow. It extends FlowBaseElement and inherits all its fields. Available in API version 31.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignToReference</td>
<td>string</td>
<td>Required. Specifies the variable to which you want to assign the output parameter value.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the output parameter.</td>
</tr>
</tbody>
</table>

FlowApexPluginCall

Defines a call to an Apex plug-in from the flow. It extends FlowNode and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apexClass</td>
<td>string</td>
<td>Required. The name of the Apex class.</td>
</tr>
<tr>
<td>connector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute after this Apex plug-in call.</td>
</tr>
<tr>
<td>faultConnector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute if the Apex plug-in call results in an error.</td>
</tr>
<tr>
<td>inputParameters</td>
<td>FlowApexPluginCallInputParameter[]</td>
<td>An array of input parameters from the flow to the Apex plug-in.</td>
</tr>
<tr>
<td>outputParameters</td>
<td>FlowApexPluginCallOutputParameter[]</td>
<td>An array of output parameters from the Apex plug-in to the flow.</td>
</tr>
</tbody>
</table>

FlowApexPluginCallInputParameter

Defines an input parameter from the flow to the Apex plug-in. It extends FlowBaseElement and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the input parameter.</td>
</tr>
<tr>
<td>value</td>
<td>FlowElementReferenceOrValue</td>
<td>Defines the value of the input parameter.</td>
</tr>
</tbody>
</table>

FlowApexPluginCallOutputParameter

Defines an output parameter from the Apex plug-in to the flow. It extends FlowBaseElement and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignToReference</td>
<td>string</td>
<td>Required. Specifies the variable to which you want to assign the output parameter value.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the output parameter.</td>
</tr>
</tbody>
</table>
FlowAssignment

Defines an assignment node that can dynamically change the value of a variable in the flow. It extends FlowNode and inherits all of its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignmentItems</td>
<td>FlowAssignmentItem[]</td>
<td>An array of assignment operations that is executed in the given order, starting from the index 0.</td>
</tr>
<tr>
<td>connector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute after this assignment node.</td>
</tr>
</tbody>
</table>

FlowAssignmentItem

Defines an operation to apply to a variable. It extends FlowBaseElement and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignToReference</td>
<td>string</td>
<td>Required. Reference to the variable to which you want to apply the specified operator.</td>
</tr>
<tr>
<td>operator</td>
<td>FlowAssignmentOperator (enumeration of type string)</td>
<td>Required. Operation to apply to the variable reference in the assignToReference field. For valid values, see FlowAssignmentOperator.</td>
</tr>
<tr>
<td>value</td>
<td>FlowElementReferenceOrValue</td>
<td>Defines the value that you want the operator to apply to the variable reference in the assignToReference field.</td>
</tr>
</tbody>
</table>

FlowAssignmentOperator

An enumeration of type string that specifies the operation to apply to the variable in the assignToReference field. See “Flow Operators in Assignment Elements” in Salesforce Help.

Valid values are:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>When the assignToReference field is a variable of type number or currency, this operator adds the value to the variable.</td>
</tr>
<tr>
<td></td>
<td>When the assignToReference field is a variable of type date, this operator adds the value in days to the variable.</td>
</tr>
<tr>
<td></td>
<td>When the assignToReference field is a variable of type string, this operator appends the value to the end of the string.</td>
</tr>
<tr>
<td></td>
<td>When the assignToReference field is a variable of type picklist, this operator appends the value to the end of the last item in the picklist.</td>
</tr>
<tr>
<td></td>
<td>When the assignToReference field is a variable of type multipicklist, this operator appends the value to the end of the last item in the multi-select picklist. To instead add an item to the end of the multi-select picklist, use the AddItem operator.</td>
</tr>
<tr>
<td>Enumeration Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>When the <code>assignToReference</code> field is the <code>$Flow.ActiveStages</code> global variable, this operator appends the value as a new item at the end of <code>$Flow.ActiveStages</code>. When the <code>assignToReference</code> field is a collection variable, this operator appends the value to the end of the collection. Support for a collection variable as the value is available in API version 43.0 and later but only via Metadata API. From Flow Builder, you can't save an Assignment element that contains a collection variable in the Value column for the Add operator. The Add operator is not supported when the <code>assignToReference</code> field is a variable of type boolean, dateTime, or sObject.</td>
<td></td>
</tr>
<tr>
<td>AddAtStart</td>
<td>Supported only when the <code>assignToReference</code> field is a collection variable or the <code>$Flow.ActiveStages</code> global variable. Adds the value as a new item at the beginning of the collection. When the value is a collection variable, the operator adds all items at the beginning of the collection. This operator is available in API version 43.0 and later.</td>
</tr>
<tr>
<td>AddItem</td>
<td>Supported only when the <code>assignToReference</code> field is a variable of type multipicklist. Adds the value to the picklist, including the semi-colon that's required to mark a value as a separate item. This operator is available in API version 34.0 and later.</td>
</tr>
<tr>
<td>Assign</td>
<td>Assigns the value to the variable in the <code>assignToReference</code> field.</td>
</tr>
<tr>
<td>AssignCount</td>
<td>Supported only when the value is a collection variable or the <code>$Flow.ActiveStages</code> global variable. Counts the number of stages or items in the collection and assigns that number to the variable in the <code>assignToReference</code> field. Corresponds to equals count in the user interface. This operator is available in API version 43.0 and later.</td>
</tr>
<tr>
<td>RemoveAfterFirst</td>
<td>Supported only when the <code>assignToReference</code> field is a collection variable or the <code>$Flow.ActiveStages</code> global variable. Finds the first instance of the value within the variable in the <code>assignToReference</code> field. Removes everything after that first instance from the variable. This operator is available in API version 43.0 and later.</td>
</tr>
<tr>
<td>RemoveAll</td>
<td>Supported only when the <code>assignToReference</code> field is a collection variable or the <code>$Flow.ActiveStages</code> global variable. Removes all instances of the value from the variable in the <code>assignToReference</code> field. When the value is a collection variable, the operator removes all instances of each item from the variable in the <code>assignToReference</code> field. This operator is available in API version 43.0 and later.</td>
</tr>
<tr>
<td>RemoveBeforeFirst</td>
<td>Supported only when the <code>assignToReference</code> field is a collection variable or the <code>$Flow.ActiveStages</code> global variable. Finds the first instance of the value within the variable in the <code>assignToReference</code> field. Removes everything before that first instance from the variable. This operator is available in API version 43.0 and later.</td>
</tr>
<tr>
<td>RemoveFirst</td>
<td>Supported only when the <code>assignToReference</code> field is a collection variable or the <code>$Flow.ActiveStages</code> global variable. Removes the first instance of the value from the variable in the <code>assignToReference</code> field. This operator is available in API version 43.0 and later.</td>
</tr>
<tr>
<td>RemovePosition</td>
<td>Supported only when the <code>assignToReference</code> field is a collection variable or the <code>$Flow.ActiveStages</code> global variable. Removes the item at the specified position. For example, if the collection contains three items (Red, Green, Blue) and the value is 2, the...</td>
</tr>
</tbody>
</table>
second item (Green) is removed from the collection variable. This operator is available in API version 43.0 and later.

Make sure that the value at run time will be a positive integer within the range of the number of items in the collection variable.

RemoveUncommon

Supported only when assignToReference and value are both collection variables. Keeps items that are in both collections and removes the rest from the collection variable in the assignToReference field. This operator is available in API version 43.0 and later.

Subtract

Supported only when the assignToReference field is a variable of type currency, date, or number.

When the assignToReference field is a variable of type number or currency, this operator subtracts the value from the variable.

When the assignToReference field is a variable of type date, this operator subtracts the value in days from the variable.

FlowChoice

A choice resource is a standalone choice option that you can reference or reuse throughout the flow. It extends FlowElement and inherits all of its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>choiceText</td>
<td>string</td>
<td>Required. Choice label to display in the screen.</td>
</tr>
<tr>
<td>dataType</td>
<td>FlowDataType (enumeration of type string)</td>
<td>Required. Valid types are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Currency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Date</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Boolean</td>
</tr>
<tr>
<td>userInput</td>
<td>FlowChoiceUserInput</td>
<td>Enables the choice to allow user input when the choice is selected. Not supported for choices in multi-select fields.</td>
</tr>
<tr>
<td>value</td>
<td>FlowElementReferenceOrValue</td>
<td>Actual value that’s used during flow execution, for example, in assignments, calls to Apex plug-ins, and record elements. If null, this choice always has the value of null.</td>
</tr>
</tbody>
</table>

FlowChoiceUserInput

Allows the choice to include a user input field that appears when the choice is selected by the user. User input isn’t supported for choices in multi-select fields. It extends FlowBaseElement and inherits all its fields.
Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isRequired</td>
<td>boolean</td>
<td>Indicates whether users are required to enter something into the field when</td>
</tr>
<tr>
<td></td>
<td></td>
<td>they select the choice.</td>
</tr>
<tr>
<td>promptText</td>
<td>string</td>
<td>Text that is displayed to prompt the user for input at runtime.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Supports merge fields.</td>
</tr>
<tr>
<td>validationRule</td>
<td>FlowInputValidationRule</td>
<td>Rule used at runtime to validate the user input.</td>
</tr>
</tbody>
</table>

**FlowCondition**

Defines a condition for a rule. It extends FlowBaseElement and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>leftValueReference</td>
<td>string</td>
<td>Required. Unique name of the element that serves as the left side of the condition expression.</td>
</tr>
<tr>
<td>operator</td>
<td>FlowComparisonOperator (enumeration of</td>
<td>Required. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>type string)</td>
<td>• EqualTo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NotEqualTo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• GreaterThan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LessThan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• GreaterThanOrEqualTo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LessThanOrEqualTo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• StartsWith</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• EndsWith</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Contains</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IsNull</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• WasSet—This value is available in API version 30.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• WasSelected—Requires a choice on the left side.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• WasVisited—Requires a node on the left side.</td>
</tr>
<tr>
<td>rightValue</td>
<td>FlowElementReferenceOrValue</td>
<td>Unique name of an element or the actual value (such as text or a number) for the right side of the condition expression.</td>
</tr>
</tbody>
</table>

**FlowConnector**

Connectors determine the order in which the nodes of the flow are executed. A connector defines and links to the subsequent node. It extends FlowBaseElement and inherits all its fields.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetReference</td>
<td>string</td>
<td>Required. Which node to execute after completing the current node.</td>
</tr>
</tbody>
</table>

**FlowConstant**

A constant resource defines a fixed value that can be used throughout your flow. It extends `FlowElement` and inherits all of its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataType</td>
<td>FlowDataType (enumeration of type string)</td>
<td>Required. Valid types are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Currency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Date</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Boolean</td>
</tr>
<tr>
<td>value</td>
<td>FlowElementReferenceOrValue</td>
<td>Default value of the constant. This field can’t have merge fields, nor can it reference another resource besides <code>$GlobalConstant.EmptyString</code>.</td>
</tr>
</tbody>
</table>

**FlowDecision**

Decision node that evaluates a set of rules and routes the flow execution based on the first rule that evaluates to true. It extends `FlowNode` and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultConnector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute if none of the rules evaluate to true.</td>
</tr>
<tr>
<td>defaultConnectorLabel</td>
<td>string</td>
<td>Label for the default connector.</td>
</tr>
<tr>
<td>rules</td>
<td>FlowRule[]</td>
<td>An array of rules for the decision. The rules are evaluated in the order they’re listed, and the connector of the first true rule is used. If no rules are true, then the default connector is used. In Flow Builder, rules are referred to as “decision outcomes.”</td>
</tr>
</tbody>
</table>

**FlowDynamicChoiceSet**

Looks up data or metadata from an sObject and dynamically generates a set of choices at run time. It extends `FlowElement` and inherits all of its fields. Depending on the fields that are set, this element represents either a record choice or a picklist choice.

- A record choice dynamically generates choices based on records that meet specified filter criteria. If a dynamic choice doesn’t have the picklistField and picklistObject parameters set, it is a record choice and can’t have a data type of Picklist or Multipicklist.
- A **picklist choice** dynamically generates choices based on the available values for a picklist or multi-select picklist field. If a dynamic choice has the `picklistField` and `picklistObject` parameters set, it is a picklist choice and must have a data type of `Picklist` or `Multipicklist`.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>dataType</code></td>
<td><code>FlowdataType (enumeration of type string)</code></td>
<td>Required. Valid types are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Currency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Date</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Boolean</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Picklist—Picklist choices only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Multipicklist—Picklist choices only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Picklist and Multipicklist are available in API version 35.0 and later.</td>
</tr>
<tr>
<td><code>displayField</code></td>
<td><code>string</code></td>
<td>Required for record choices. Which field from the sObject to display to the user as choice labels.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, for an account, use the DisplayField “Name” if you want the dynamically generated choices to be displayed as the account names from the records that are retrieved from the database.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Not supported for picklist choices. Picklist choices always display the labels for the retrieved picklist values.</td>
</tr>
<tr>
<td><code>filters</code></td>
<td><code>FlowRecordFilter[]</code></td>
<td>An array of filters to apply to the records that are retrieved from the database. For example, filter accounts to include only the accounts that were created in the past three months.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Not supported for picklist choices.</td>
</tr>
<tr>
<td><code>limit</code></td>
<td><code>int</code></td>
<td>Maximum number of choices to include in the generated set of choices. Maximum and default: 200.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If <code>sortField</code> and <code>sortOrder</code> are also specified, the records are sorted before the <code>limit</code> takes effect.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 25.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is nillable in API version 45.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>object</code></td>
<td><code>string</code></td>
<td>Required for record choices. The <code>sObject</code> whose fields you want to retrieve from the database and use to generate the set of choices. For example, use “Account” to dynamically generate choices from the information in account records in the database.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Not supported for picklist choices.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>outputAssignments</td>
<td>FlowOutputFieldAssignment[]</td>
<td>An array that assigns fields from the user-selected record to variables that can be used elsewhere in the flow. For example, when the user selects an account name from the dynamically generated list of choice options, outputAssignments can assign the Id and AnnualRevenue from the user-selected account to variables that you specify. Not supported for picklist choices.</td>
</tr>
<tr>
<td>picklistField</td>
<td>string</td>
<td>Required for picklist choices. The field whose available values you want to retrieve from the database and use to generate the picklist choice. For example, use “Industry” to dynamically generate one choice for each available value on the Industry picklist field. Not supported for record choices.</td>
</tr>
<tr>
<td>picklistObject</td>
<td>string</td>
<td>Required for picklist choices. The sObject whose field metadata you want to retrieve from the database and use to generate the picklist choice. For example, use “Account” to dynamically generate choices from a picklist field on the Account object. Not supported for record choices.</td>
</tr>
<tr>
<td>sortField</td>
<td>string</td>
<td>Field that is used for sorting the records that meet the filter criteria. If this field isn’t specified then the returned records are not sorted. You can only sort records by fields that have the Sort API field property, as specified in SOAP API. Not supported for picklist choices.</td>
</tr>
<tr>
<td>sortOrder</td>
<td>SortOrder (enumeration of type string)</td>
<td>Order in which to sort the records. If this field isn’t specified, then the results are not sorted. Valid values are: Asc—Ascending Desc—Descending Not supported for picklist choices.</td>
</tr>
<tr>
<td>valueField</td>
<td>string</td>
<td>Stored value for the choice, which can differ from what is displayed to the user as the choice options (displayField). For example, the displayField</td>
</tr>
</tbody>
</table>
FlowElement

Base class for all flow elements. This is an abstract class. It extends FlowBaseElement and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the flow element.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name of the flow element.</td>
</tr>
</tbody>
</table>

FlowBaseElement

Base class for all flow elements that require contextual information in metadata values. This is an abstract class. FlowBaseElement is available in API version 32.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>processMetadataValues</td>
<td>FlowMetadataValue[]</td>
<td>Contextual information for the element.</td>
</tr>
</tbody>
</table>

FlowMetadataValue

Defines contextual information that can be passed between elements in a flow. Flow metadata values can be used in an application that produces or consumes flows. FlowMetadataValue is available in API version 31.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Name for the metadata value. This name doesn’t need to be unique across all elements.</td>
</tr>
<tr>
<td>value</td>
<td>FlowElementReferenceOrValue</td>
<td>Reference or value for the metadata value.</td>
</tr>
</tbody>
</table>

FlowElementReferenceOrValue

Defines a reference to an existing element or a particular value that you specify. Make sure that you specify only one of the fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanValue</td>
<td>boolean</td>
<td>Use this field to specify a boolean value. Do not use this field if you want to specify a different data type or an element reference.</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
dateTimeValue | dateTime | Use this field to specify a dateTime value. Do not use this field if you want to specify a different data type or an element reference. This field is available in API version 30.0 and later.
dateValue | date | Use this field to specify a date value. Do not use this field if you want to specify a different data type or an element reference.
elementReference | string | Use this field to specify the name of an existing element. Do not use this field if you want to specify a value instead of an element reference.
numberValue | double | Use this field to specify a double value. Do not use this field if you want to specify a different data type or an element reference.
stringValue | string | Use this field to specify a string value. Do not use this field if you want to specify a different data type or an element reference.

#### FlowFormula
Calculates a value using functions and elements in the flow. It extends `FlowElement` and inherits all its fields.

### Field Name | Field Type | Description
--- | --- | ---
dataType | FlowDataType (enumeration of type string) | The data type for the formula. Valid values are:
- Boolean
- Currency
- Date
- DateTime
- Number
- String
dataType defaults to Number if it isn’t defined in a formula. This field is available in API version 31.0 and later.
expression | string | Required. Salesforce formula expression. The return value must match the data type. See “Flow Formula Considerations” in Salesforce Help.
For API version 30.0 and earlier, the return value must be numeric.
scale | int | Scale of the return value, specifically, the number of digits to the right of the decimal point. Only supported for Currency and Number data types.

#### FlowInputFieldAssignment
Assigns the value for a record field based on a resource or static value. It extends `FlowBaseElement` and inherits all its fields.
FlowInputValidationRule
Validation rules verify that the data entered by the user meets the specified requirements. If the validation rule evaluates to false, then the specified error message is displayed.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorMessage</td>
<td>string</td>
<td>Required. Error message to display when <code>formulaExpression</code> evaluates to false.</td>
</tr>
<tr>
<td>formulaExpression</td>
<td>string</td>
<td>Required. Boolean formula used to validate the user input. See &quot;Flow Formula Considerations&quot; in Salesforce Help.</td>
</tr>
</tbody>
</table>

FlowLoop
A construct for iterating through a collection. It extends FlowNode and inherits all of its fields. FlowLoop is available in API version 30.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nextValueConnector</td>
<td>FlowConnector</td>
<td>Points to the element that the flow navigates to for each of the entries in the collection. This is where the flow goes for the next value in the collection.</td>
</tr>
<tr>
<td>noMoreValuesConnector</td>
<td>FlowConnector</td>
<td>Points to the element to navigate to when all entries in the collection have been looped through.</td>
</tr>
<tr>
<td>collectionReference</td>
<td>string</td>
<td>Required. The collection being looped through.</td>
</tr>
<tr>
<td>assignNextValueToReference</td>
<td>string</td>
<td>Required. The variable to which the current value in the collection is assigned before navigating to the target of <code>nextValueConnector</code>.</td>
</tr>
<tr>
<td>iterationOrder</td>
<td>iterationOrder (enumeration of type string)</td>
<td>Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Asc—Iterate through the collection in the order the values are listed (first to last).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Desc—Iterate through the collection in the reverse order the values are listed (last to first).</td>
</tr>
</tbody>
</table>

FlowNode
A node is a type of element that is visible in the flow diagram. It extends FlowElement and inherits all its fields.
### Field Name | Field Type | Description
---|---|---
label | string | Required. Name of the node. This non-unique label is different from the unique name of the node, which is inherited from FlowElement.
locationX | int | Required. Horizontal location of the node, in pixels from the left.
locationY | int | Required. Vertical location of the node, in pixels from the top.

**FlowOutputFieldAssignment**

Assigns a record field’s value from a record to a variable that can be used elsewhere in the flow. The record may be selected by a record lookup or via a user selection for a choice. It extends FlowBaseElement and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignToReference</td>
<td>string</td>
<td>Required. Reference to the variable where you want to store the value of the record field.</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>Required. Name of the field whose value is to be assigned after a record lookup.</td>
</tr>
</tbody>
</table>

**FlowRecordCreate**

Create a new record in the database using values from the flow. It extends FlowNode and inherits all its properties.

닛: The flow record create, lookup, update, and delete operations are different from the CRUD-based metadata calls create(), retrieve(), update(), and delete(). The flow record methods apply to record operations from within a flow, which aren’t the same as doing any metadata calls to CRUD setup entities.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignRecordIdToReference</td>
<td>string</td>
<td>Reference to the variable where you want to store the ID after the record is created.</td>
</tr>
<tr>
<td>connector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute after creating the record.</td>
</tr>
<tr>
<td>faultConnector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute if the attempt to create a record results in an error.</td>
</tr>
<tr>
<td>inputAssignments</td>
<td>FlowInputFieldAssignment[]</td>
<td>An array that assigns values to the specified fields of the record being created.</td>
</tr>
<tr>
<td>object</td>
<td>string</td>
<td>Required. sObject for the record to be created by this element</td>
</tr>
<tr>
<td>inputReference</td>
<td>string</td>
<td>Specifies the sObject variable whose field values are used to populate the new record’s fields.</td>
</tr>
</tbody>
</table>
**FlowRecordDelete**

Deletes one or more records in the database. It extends FlowNode and inherits all its fields.

> **Note:** The flow record create, lookup, update, and delete operations are different from the CRUD-based metadata calls create(), retrieve(), update(), and delete(). The flow record methods apply to record operations from within a flow, which aren’t the same as doing any metadata calls to CRUD setup entities.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute after deleting the record.</td>
</tr>
<tr>
<td>faultConnector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute if the attempt to delete a record results in an error.</td>
</tr>
<tr>
<td>filters</td>
<td>FlowRecordFilter[]</td>
<td>An array that specifies the criteria used to select which records to delete from the database. For example, delete accounts whose last activity was older than a specified date.</td>
</tr>
<tr>
<td>object</td>
<td>string</td>
<td>Required. The name of the object whose records are deleted.</td>
</tr>
<tr>
<td>inputReference</td>
<td>string</td>
<td>Specifies the sObject variable whose record ID is used to identify which record to delete in the database.</td>
</tr>
</tbody>
</table>

**FlowRecordFilter**

Sets the criteria for searching records in the database. It extends FlowBaseElement and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>Required. The field to be used for filtering records.</td>
</tr>
<tr>
<td>operator</td>
<td>FlowRecordFilterOperator (enumeration of type string)</td>
<td>Required. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• EqualTo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NotEqualTo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• GreaterThan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LessThan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• GreaterThanOrEqualTo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LessThanOrEqualTo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• StartsWith</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• EndsWith</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Contains</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IsNull</td>
</tr>
<tr>
<td>value</td>
<td>FlowElementReferenceOrValue</td>
<td>Reference or value used with the field and operator to filter records.</td>
</tr>
</tbody>
</table>
**FlowRecordLookup**

Finds a record in the database and uses or stores the values from its fields in the flow. It extends `FlowNode` and inherits all of its fields.

**Note:** The flow record create, lookup, update, and delete operations are different from the CRUD-based metadata calls `create()`, `retrieve()`, `update()`, and `delete()`. The flow record methods apply to record operations from within a flow, which aren’t the same as doing any metadata calls to CRUD setup entities.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignNullValuesInRecordFound</td>
<td>boolean</td>
<td>Specifies that all values are set to null if the record is not found. This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>connector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute after completing the record lookup.</td>
</tr>
<tr>
<td>faultConnector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute if the attempt to look up a record results in an error.</td>
</tr>
<tr>
<td>filters</td>
<td>FlowRecordFilter[]</td>
<td>An array that specifies the criteria used to select the record from the database.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the filters return more than one record, they are sorted according to the specified <code>sortField</code> and <code>sortOrder</code>. The first record in the sorted list is then selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If either the <code>sortField</code> or <code>sortOrder</code> is not specified, then the first record returned is selected. Note, however, that records are not returned in any particular order.</td>
</tr>
<tr>
<td>limit</td>
<td>int</td>
<td>The maximum number of records to return, to limit the amount of data received. This field is available in API version 30.0 and later.</td>
</tr>
<tr>
<td>object</td>
<td>string</td>
<td>Required. Name of the sObject from which to select the record.</td>
</tr>
<tr>
<td>outputAssignments</td>
<td>FlowOutputFieldAssignment[]</td>
<td>An array that assigns fields from the selected record to variables that can be used elsewhere in the flow.</td>
</tr>
<tr>
<td>outputReference</td>
<td>string</td>
<td>Specifies the sObject variable that stores the queried fields’ values.</td>
</tr>
<tr>
<td>queriedFields</td>
<td>string[]</td>
<td>An array that specifies which fields from the selected record is saved to the specified sObject variable.</td>
</tr>
<tr>
<td>sortField</td>
<td>string</td>
<td>Field that is used for sorting the records that meet the filter criteria. If this field isn’t specified then the returned records are not sorted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can only sort records by fields that have the Sort API field property, as specified in SOAP API.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 25.0 and later.</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---

- **sortOrder** | SortOrder (enumeration of type string) | Order in which to sort the records. If this field isn’t specified, then the results are not sorted. Valid values are:
  - Asc—Ascending
  - Desc—Descending
  
  This field is available in API version 25.0 and later.

---

### FlowRecordUpdate
Finds records in the database and updates them with values from the flow. It extends FlowNode and inherits all of its fields.

*Note:* The flow record create, lookup, update, and delete operations are different from the CRUD-based metadata calls `create()`, `retrieve()`, `update()`, and `delete()`. The flow record methods apply to record operations from within a flow, which aren’t the same as doing any metadata calls to CRUD setup entities.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute after completing the record update.</td>
</tr>
<tr>
<td>faultConnector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute if the attempt to update a record results in an error.</td>
</tr>
<tr>
<td>filters</td>
<td>FlowRecordFilter[]</td>
<td>An array that specifies the criteria used to select the records to update in the database.</td>
</tr>
<tr>
<td>inputAssignments</td>
<td>FlowInputFieldAssignment[]</td>
<td>An array that assigns values to the specified fields of the record being updated.</td>
</tr>
<tr>
<td>object</td>
<td>string</td>
<td>Required. Name of the sObject whose records are updated.</td>
</tr>
<tr>
<td>inputReference</td>
<td>string</td>
<td>Specifies the sObject variable whose field values are used to update the record’s fields.</td>
</tr>
</tbody>
</table>

---

### FlowRule
Defines the conditions and logic that would enable a rule to evaluate to true. It extends FlowElement and inherits all of its fields.

| Field Name     | Field Type | Description
|----------------|------------|---------------|
| conditionLogic | string     | Specifies logic for the conditions. Value can be:
  - and—evaluates to true only if all its conditions evaluate to true
  - or—evaluates to true if any of its conditions evaluate to true
Advanced logic like 1 AND (2 OR 3) — evaluates to true if the first condition is true and either the second or third condition is true.

When you use advanced logic, the string must consist of 1,000 or fewer characters.

Advanced logic is available in API version 33.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>conditions</td>
<td>FlowCondition[]</td>
<td>An array of conditions for the rule.</td>
</tr>
<tr>
<td>connector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute if this is the first rule that evaluates to true in a decision.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Label for the connector.</td>
</tr>
</tbody>
</table>

**FlowScreen**

Screens provide the ability to capture information from users and display information to users. It extends FlowNode and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowBack</td>
<td>boolean</td>
<td>Indicates whether to show (true) or hide (false) the Previous button on the screen at runtime. When true, the Previous button appears only if the user visited a previous screen in the flow path and if showFooter for the screen is set to true. Set this to false when revisiting the previous screen would trigger an action that should not be repeated, such as a credit card transaction. This field is available in API version 26.0 and later. Default: true. You can set either allowBack or allowFinish to false, but not both.</td>
</tr>
<tr>
<td>allowFinish</td>
<td>boolean</td>
<td>Indicates whether to show (true) or hide (false) the Finish button on the screen at runtime. When true, the Finish button appears only if the screen element is the end of a flow path and if showFooter for the screen is set to true. Set this to false if you need the user to go back to a previous screen to continue or complete the flow. For example, you wouldn’t want to offer a Finish button on a screen that tells the user to go back and make corrections on a previous screen. This field is available in API version 26.0 and later. Default: true. You can set either allowBack or allowFinish to false, but not both.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>allowPause</td>
<td>boolean</td>
<td>Indicates whether to show (true) or hide (false) the <strong>Pause</strong> button on the screen at runtime. The default value is true. A flow screen displays the <strong>Pause</strong> button if all the following conditions are true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• In the organization’s process automation settings, <strong>Let users pause flows</strong> is enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• allowPause for the screen is set to true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the flow is embedded in a Visualforce page, the <code>&lt;flow:interview&gt;</code> component has its <strong>showAllowPause</strong> attribute set to true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The <strong>showFooter</strong> field for the screen is set to true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 33.0 and later.</td>
</tr>
<tr>
<td>connector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute after the screen node.</td>
</tr>
<tr>
<td>fields</td>
<td>FlowScreenField[]</td>
<td>An array of fields to display on the screen.</td>
</tr>
<tr>
<td>helpText</td>
<td>string</td>
<td>Text that appears if the end user clicks a link for help text. Supports merge fields in API version 26.0 and later.</td>
</tr>
<tr>
<td>pausedText</td>
<td>string</td>
<td>Confirmation message that appears when an end-user clicks <strong>Pause</strong>. This field is available in API version 33.0 and later.</td>
</tr>
<tr>
<td>rules</td>
<td>FlowScreenRule[]</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>showFooter</td>
<td>boolean</td>
<td>Indicates whether to show (true) or hide (false) the screen’s footer at Lightning runtime. Classic runtime isn’t supported. The default value is true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The footer includes navigation actions for the screen. If <strong>showFooter</strong> is hidden, use Lightning components on the screen to show navigation actions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 42.0 and later.</td>
</tr>
<tr>
<td>showHeader</td>
<td>boolean</td>
<td>Indicates whether to show (true) or hide (false) the screen’s header at Lightning runtime. Classic runtime isn’t supported. The default value is true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The header includes access to help text for the screen. If <strong>showHeader</strong> is hidden, use Lightning components on the screen to show help text.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 42.0 and later.</td>
</tr>
</tbody>
</table>

**FlowScreenField**

Configurable field on a screen. It extends **FlowElement** and inherits all its fields.
### Field Name | Field Type | Description
--- | --- | ---
choiceReferences | string[] | An array of references to FlowChoices or FlowDynamicChoiceSets. The resulting choice options appear in the order specified in this array, where the element at index 0 provides the top-most choice option. Supported for the following screen field types:
- RadioButtons
- DropdownBox
- MultiSelectCheckboxes
- MultiSelectPicklist

Multi-select checkboxes and multi-select picklist fields are available in API version 26.0 and later.

dataType | FlowDataType (enumeration of type string) | Required. Data type of this screen field. Only supported for the InputField, RadioButtons, and DropdownBox screen field types. Valid types are:
- Boolean
- Currency
- Date
- DateTime
- Number
- String

Boolean input fields, which appear as checkbox fields at runtime, are available in API version 26.0 and later.

Only the string data type is supported for multi-select checkboxes and multi-select picklist fields. Multi-select fields are available in API version 26.0 and later.

Date/time input fields are available in API version 43.0 and later.

defaultSelectedChoiceReference | string | The name of the FlowChoice element to use as the default value for the screen field. Supported for the following screen field types:
- RadioButtons
- DropdownBox
- MultiSelectCheckboxes
- MultiSelectPicklist
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultValue</td>
<td>FlowElementReferenceOrValue</td>
<td>The value that is used by default when this screen field requires users to provide input. Only supported for InputField, LargeTextArea, and PasswordField.</td>
</tr>
<tr>
<td>extensionName</td>
<td>string</td>
<td>The name of the Lightning component to display. This field is available in API version 42.0 and later.</td>
</tr>
<tr>
<td>fieldText</td>
<td>string</td>
<td>Field label that is displayed on the screen. Supports merge fields.</td>
</tr>
<tr>
<td>fieldType</td>
<td>FlowScreenFieldType (enumeration of type string)</td>
<td>Required. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DisplayText</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• InputField</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LargeTextArea</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PasswordField</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RadioButtons</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DropdownBox</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MultiSelectCheckboxes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MultiSelectPicklist</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ComponentInstance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>At runtime, each multi-select field stores its field value as a concatenation of the user-selected choice values, separated by semicolons. Any semicolons in the selected choice values are removed when added to the multi-select field value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Multi-select checkboxes and multi-select picklist fields are available in API version 26.0 and later. Lightning component fields are available in API version 42.0.</td>
</tr>
</tbody>
</table>
## Metadata Types

### FlowScreenFieldInputParameter

Defines an input parameter from the flow to the extension. It extends FlowBaseElement and inherits all its fields. FlowScreenFieldInputParameter is available in API version 42.0.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the input parameter.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Defines the value of the input parameter.</td>
</tr>
</tbody>
</table>

### FlowScreenFieldOutputParameter

Defines an output parameter from the extension to the flow. It extends FlowBaseElement and inherits all its fields. FlowScreenFieldOutputParameter is available in API version 42.0.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the output parameter.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Defines the value of the output parameter.</td>
</tr>
</tbody>
</table>

---

**FlowMetadata Types**
**FlowStage**

A section of your flow that can be represented in the UI, such as with breadcrumbs. It extends FlowElement and inherits all its fields. When an interview starts, any stages where `isActive` is true are added to the `$Flow.ActiveStages` global variable, which holds a collection of stages. Each stage's `stageOrder` determines the order they're added in. The stage with the lowest `stageOrder` is assigned to the `$Flow.CurrentStage` global variable.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Indicates whether the stage is active by default.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>A user-friendly label for this stage.</td>
</tr>
<tr>
<td>stageOrder</td>
<td>int</td>
<td>Indicates how the stage should be ordered against other stages. The <code>stageOrder</code> value must be unique within the flow.</td>
</tr>
</tbody>
</table>

**FlowStep**

Steps function as placeholders when you’re building a flow. It extends FlowNode and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectors</td>
<td>FlowConnector[]</td>
<td>Specifies which node to execute after the step node.</td>
</tr>
</tbody>
</table>

**FlowSubflow**

A subflow element references another flow, which it calls at run time. The flow that contains the subflow element is referred to as the master flow. FlowSubflow extends FlowNode and inherits all of its fields. It is available in API version 25.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute after the subflow.</td>
</tr>
<tr>
<td>flowName</td>
<td>string</td>
<td>References the flow to call at runtime. The value must be an API name of a flow and can’t contain an appended hyphen and version number.</td>
</tr>
<tr>
<td>inputAssignments</td>
<td>FlowSubflowInputAssignment[]</td>
<td>An array of input variable assignments that are set at the start of the referenced flow.</td>
</tr>
<tr>
<td>outputAssignments</td>
<td>FlowSubflowOutputAssignment[]</td>
<td>An array of output variable assignments that are set at the end of the referenced flow.</td>
</tr>
</tbody>
</table>
FlowSubflowInputAssignment

Assigns an element or value from the master flow to a variable in the referenced flow. Input assignments occur when the subflow calls the referenced flow. It extends FlowBaseElement and inherits all its fields. It is available in API version 25.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the variable in the referenced flow.</td>
</tr>
<tr>
<td>value</td>
<td>FlowElementReferenceOrValue</td>
<td>Defines the value to assign to the variable.</td>
</tr>
</tbody>
</table>

FlowSubflowOutputAssignment

Assigns the value of a variable from the referenced flow to a variable in the master flow. Output assignments occur when the referenced flow is finished running. It extends FlowBaseElement and inherits all its fields. It is available in API version 25.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignToReference</td>
<td>string</td>
<td>Required. Unique name for the variable in the master flow.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the variable in the referenced flow.</td>
</tr>
</tbody>
</table>

FlowTextTemplate

Defines a text template that can be used throughout the flow. It extends FlowElement and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>string</td>
<td>Actual text of the template. Supports merge fields.</td>
</tr>
</tbody>
</table>

FlowVariable

Variables allow you to create updatable values to use in the flow. FlowVariable extends FlowElement and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apexClass</td>
<td>string</td>
<td>The Apex class of this variable if its data type is Apex. Available in API version 46.0 and later.</td>
</tr>
</tbody>
</table>
| dataType        | FlowDataType (enumeration of type string) | Required. Valid types are:  
- Apex—This value is available in API version 46.0 and later.  
- Boolean  
- Currency  
- Date |
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isCollection</td>
<td>boolean</td>
<td>Indicates whether the variable is a collection of values. This field is available in API version 30.0 and later. In API version 32.0 and later, a collection variable can be of any data type. Default value is <code>False</code>.</td>
</tr>
<tr>
<td>isInput</td>
<td>boolean</td>
<td>Indicates whether the variable can be set at the start of the flow using URL parameters, Visualforce controllers, or subflow inputs. This field is available in API version 25.0 and later. Default value:</td>
</tr>
<tr>
<td>isOutput</td>
<td>boolean</td>
<td>Indicates whether the variable’s value can be accessed from Visualforce controllers and other flows. This field is available in API version 25.0 and later. Default value:</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- False for a variable created in API version 25.0 and later or in the Flow Builder in Summer ’12 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- True for a variable created in API version 24.0 or in Flow Builder in Spring ’12 and earlier.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Warning:</strong> Disabling input or output access for an existing variable can break the functionality of applications and pages that call the flow and access the variable. For example, you can access variables from URL parameters, processes, and other flows.</td>
</tr>
<tr>
<td>objectType</td>
<td>string</td>
<td>Object type of this variable if its data type is sObject.</td>
</tr>
<tr>
<td>scale</td>
<td>int</td>
<td>Scale of this variable if its data type is Number or Currency.</td>
</tr>
<tr>
<td>value</td>
<td>FlowElementReferenceOrValue</td>
<td>Default value of this variable. Default values aren’t supported if the variable’s data type is Picklist or Multipicklist.</td>
</tr>
</tbody>
</table>

## FlowWait

Waits for one or more defined events to occur. FlowWait extends FlowNode and inherits all its fields. FlowWait is available in API version 32.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultConnector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute if the conditions are false for every event in the Wait element.</td>
</tr>
<tr>
<td>defaultConnectorLabel</td>
<td>string</td>
<td>Label for the default connector.</td>
</tr>
<tr>
<td>faultConnector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute if the attempt to wait results in an error. If any of the wait events fail, the flow takes the fault connector.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>waitEvents</td>
<td>FlowWaitEvent[]</td>
<td>An array of events that the Wait element is waiting for. If the conditions for every event evaluate to false, the defaultConnector is used.</td>
</tr>
</tbody>
</table>

**FlowWaitEvent**

An event that a FlowWait element is waiting for. FlowWaitEvent extends FlowElement and inherits all its fields. FlowWaitEvent is available in API version 32.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>conditionLogic</td>
<td>string</td>
<td>Specifies logic for the conditions. Value can be:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• and—evaluates to true only if all its conditions evaluate to true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• or—evaluates to true if any of its conditions evaluate to true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Advanced logic like 1 AND (2 OR 3) —evaluates to true if the first condition is true and either the second or third condition is true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When you use advanced logic, the string must consist of 1,000 or fewer characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Advanced logic is available in API version 33.0 and later.</td>
</tr>
<tr>
<td>conditions</td>
<td>FlowCondition[]</td>
<td>An array of conditions that must be true for the flow to wait for this event.</td>
</tr>
<tr>
<td>connector</td>
<td>FlowConnector</td>
<td>Specifies which node to execute if this is the first event that occurs.</td>
</tr>
<tr>
<td>eventType</td>
<td>string</td>
<td>Required. The event’s type. The type determines which input parameters are available to define this event. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AlarmEvent—This event is an alarm based off of an absolute date/time value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DateRefAlarmEvent—This event is an alarm based off of a date/time field on a record.</td>
</tr>
</tbody>
</table>
FlowWaitEventInputParameter

An input parameter for FlowWaitEvent. The parameter’s value is set by using values from the flow. It extends FlowBaseElement and inherits all its fields. FlowWaitEventInputParameter is available in API version 32.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the input parameter.</td>
</tr>
<tr>
<td>value</td>
<td>FlowElementReferenceOrValue</td>
<td>Defines the value of the input parameter.</td>
</tr>
</tbody>
</table>

FlowWaitEventOutputParameter

An output parameter for FlowWaitEvent. The parameter’s value is assigned to a variable in the flow so that it can be referenced in another part of the flow. It extends FlowBaseElement and inherits all its fields. FlowWaitEventOutputParameter is available in API version 32.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignToReference</td>
<td>string</td>
<td>Required. Specifies the variable to which you want to assign the output parameter value.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the output parameter.</td>
</tr>
</tbody>
</table>

Upgrade Flow Files to API Version 44.0 or Later

In API version 43.0 and earlier, the Flow object’s fullName field included the flow’s version number. Starting in API version 44, the field no longer includes the version number. Before you deploy using API version 44.0 via Metadata API or Salesforce CLI, make sure that:

- The flows directory doesn’t include any unused flow versions.
- For each active flow, the status field is Active. Any flow without a status value is deployed or retrieved with a status value of Draft.
- The flowDefinitions directory is empty.

For Metadata API only.
The `package.xml` file is set to API version 44.0.

For the latest version of each flow, the file name doesn’t include a version number. For example, change `myflow-3.flow` to `myflow.flow`.

For Salesforce CLI only.

The `sfdx-project.json` file is set to "sourceApiVersion": "44.0".

For the latest version of each flow, the file name doesn’t include a version number. For example, change `myflow-1.flow-meta.xml` to `myflow.flow-meta.xml`.

As part of this upgrade, flow definitions are no longer necessary when you deploy or retrieve via Metadata API. If you deploy with flow definitions, the active version numbers in the flow definitions override the `status` fields in the flows. For example, the active version number in the flow definition is version 3, and the latest version of the flow is version 4 with the `status` field as `Active`. After you deploy your flow, the active version is version 3.

After you finished this upgrade, you can integrate with a version control system without worrying about flow file names changing. To reduce deployment issues when you push the source code into a scratch org, make sure you don’t reuse an existing scratch org.

### Declarative Metadata Sample Definition

A sample XML definition of a flow is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Flow xmlns="http://soap.sforce.com/2006/04/metadata">
  <choices>
    <name>Bad</name>
    <choiceText>Bad</choiceText>
    <dataType>Number</dataType>
    <value>
      <numberValue>0.0</numberValue>
    </value>
  </choices>
  <choices>
    <name>Excellent</name>
    <choiceText>Excellent</choiceText>
    <dataType>Number</dataType>
    <value>
      <numberValue>20.0</numberValue>
    </value>
  </choices>
  <choices>
    <name>Fair</name>
    <choiceText>Fair</choiceText>
    <dataType>Number</dataType>
    <value>
      <numberValue>10.0</numberValue>
    </value>
  </choices>
  <choices>
    <name>Good</name>
    <choiceText>Good</choiceText>
    <dataType>Number</dataType>
    <value>
      <numberValue>15.0</numberValue>
    </value>
  </choices>
</Flow>
```
Simple Flow app to calculate a Tip according to corporate policies.

**formulas**

- **fTipAmount**
  - **dataType**: Number
  - **expression**: {!Bill_Amount} * {!Service_Quality} / 100
  - **scale**: 0

- **fTotalAmount**
  - **dataType**: Number
  - **expression**: {!fTipAmount} + {!Bill_Amount}
  - **scale**: 0

**screens**

- **Simple_Tip_Calculator**
  - **Bill_Amount**
    - **dataType**: Currency
    - **fieldText**: Bill Amount
    - **fieldType**: InputField
    - **isRequired**: false
    - **scale**: 2

- **Service_Quality**
  - **choiceReferences**: Excellent, Good, Fair, Bad
  - **dataType**: Number
  - **fieldText**: Service Quality
  - **fieldType**: RadioButtons
  - **isRequired**: false
  - **scale**: 2

**screens**

- **TipAmount**
  - **Bill_Amount**
    - **dataType**: Currency
    - **fieldText**: Bill Amount
    - **fieldType**: InputField
    - **isRequired**: false
    - **scale**: 2

- **Tip Amount**
  - **Bill_Amount**
    - **dataType**: Currency
    - **fieldText**: Bill Amount
    - **fieldType**: InputField
    - **isRequired**: false
    - **scale**: 2
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

FlowCategory

Represents a list of flows that are grouped by category. Flows aren’t added directly to a Lightning Bolt Solution. Instead, add the category the flows are in to the Lightning Bolt Solution. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

FlowCategory components have the suffix .flowCategory and are stored in the flowCategories folder.

Version

FlowCategory components are available in API version 43.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The description of this flow category.</td>
</tr>
<tr>
<td>flowCategoryItems</td>
<td>FlowCategoryItems[]</td>
<td>The list of flows in this flow category.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The label for this flow category, which appears in Setup.</td>
</tr>
</tbody>
</table>
FlowCategoryItems

Represents the list of flows in a flow category.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flow</td>
<td>string</td>
<td>Required. The name of the flow.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a FlowCategory component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<FlowCategory xmlns="http://soap.sforce.com/2006/04/metadata"
xmns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <flowCategoryItems>
    <flow>PausableFlow</flow>
  </flowCategoryItems>
  <flowCategoryItems>
    <flow>BankingFlow</flow>
  </flowCategoryItems>
  <masterLabel>updateBenefits</masterLabel>
  <description>All the update benefits.</description>
</FlowCategory>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>updateBenefits</members>
    <name>FlowCategory</name>
  </types>
  <version>43.0</version>
</Package>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

FlowDefinition

Represents the flow definition’s description and active flow version number.

⚠️ Important: In API version 44.0, we recommend upgrading your flows to flow metadata file names without version numbers and discontinue using the FlowDefinition object to activate or deactivate a flow. Then use the Flow object to activate or deactivate a flow. For more information, see Upgrade Flow Files to API Version 44.0 on page 522.
If you deploy with flow definitions, the active version numbers in the flow definitions override the status fields in the flows. For example, the active version number in the flow definition is version 3, and the latest version of the flow is version 4 with the status field as Active. After you deploy your flow, the active version is version 3.

**Declarative Metadata File Suffix and Directory Location**

FlowDefinitions are stored in the `flowDefinitions` directory of the corresponding package directory. The file name matches the flow definition's unique full name, and the extension is `.flowDefinition`.

**Version**

FlowDefinition is available in API version 34.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>activeVersionNumber</td>
<td>int</td>
<td>The version number of the active flow.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the flow definition.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Label for the flow definition. In managed packages, this field inherits the flow’s active version name. To change this label from a subscriber’s org, edit the packaged flow name.</td>
</tr>
</tbody>
</table>

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**Folder**

Represents a folder. This type extends the Metadata metadata type and inherits its `fullName` field.

Four folder types currently exist in Salesforce:

- Document folder
- Email folder
- Report folder
- Dashboard folder

Folder type names end with the “Folder” suffix. For example, the type name of an email folder is “EmailFolder”.

**File Suffix and Directory Location**

Folders are stored in the corresponding component directory of the package. These directories are named `documents`, `email`, `reports`, and `dashboards`. Folders do not have a text file representation—they are containers for files. For each folder, an accompanying metadata file named `FolderName-meta.xml` is created at the same directory level. The `FolderName-meta.xml` metadata file contains the metadata information for that folder, such as the `accessType`. For example, for a documents folder named sampleFolder, there’s a `sampleFolder-meta.xml` within the `documents` folder of the package.
Version

Folders are available in API version 11.0 and later.

Fields

This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessType</td>
<td>FolderAccessType (enumeration of type string)</td>
<td>Required. The type of access for this folder. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Shared. This folder is accessible only by the specified set of users.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Public. This folder is accessible by all users, including portal users.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- PublicInternal. This folder is accessible by all users, excluding</td>
</tr>
<tr>
<td></td>
<td></td>
<td>portal users. This setting is available for report and dashboard folders</td>
</tr>
<tr>
<td></td>
<td></td>
<td>in organizations with a partner portal or Customer Portal enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Hidden. This folder is hidden from all users.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The name used as a unique identifier for API access. The fullName can</td>
</tr>
<tr>
<td></td>
<td></td>
<td>contain only underscores and alphanumeric characters. It must be unique,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>begin with a letter, not include spaces, not end with an underscore, and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>not contain two consecutive underscores. This field is inherited from the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Metadata component.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the document folder.</td>
</tr>
<tr>
<td>publicFolderAccess</td>
<td>PublicFolderAccessType (enumeration of type string)</td>
<td>If Public is the value for accessType, this field indicates the type of</td>
</tr>
<tr>
<td></td>
<td></td>
<td>access all users will have to the contents of the folder. Valid values</td>
</tr>
<tr>
<td></td>
<td></td>
<td>include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ReadOnly. All users can read the contents of the folder, but no user can</td>
</tr>
<tr>
<td></td>
<td></td>
<td>change the contents.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ReadWrite. All users can read or change the contents of the folder.</td>
</tr>
<tr>
<td>sharedTo</td>
<td>SharedTo</td>
<td>Sharing access for the folder. See “Sharing Considerations” in the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Salesforce online help.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is the package manifest definition of a document folder that contains a document:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <fullName>basic</fullName>
  <types>
    <members>sampleFolder</members>
    <members>sampleFolder/TestDocument.txt</members>
    <name>Document</name>
  </types>
</Package>
```
The following is an example of the `sampleFolder-meta.xml` metadata file for the sampleFolder document folder:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<DocumentFolder xmlns="http://soap.sforce.com/2006/04/metadata">
  <accessType>Public</accessType>
  <name>sampleFolder</name>
  <publicFolderAccess>ReadWrite</publicFolderAccess>
</DocumentFolder>
```

**Wildcard Support in the Manifest File**

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- Dashboard
- Document
- EmailTemplate
- Report

**FolderShare**

Represents the settings for enhanced analytics folder sharing. Users can control access to reports or dashboards by giving others Viewer, Editor or Manager access to the folder that contains the report or dashboard.

**File Suffix and Directory Location**

FolderShare objects are stored in the `reports` and `dashboards` directories. For each report or dashboard folder it contains, there is a metadata file named `FolderName-meta.xml`. The `FolderName-meta.xml` metadata file contains the metadata information for that folder, such as the `accessLevel`. For example, if the `reports` directory contains a reports folder named `myReportsFolder`, it also has a `myReportsFolder-meta.xml` file at the same level as `myReportsFolder`.

**Version**

FolderShare components are available in API version 28 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>accessLevel</code></td>
<td><code>FolderShareAccessLevel</code> (enumeration of type string)</td>
<td>Required. Specifies the combination of actions that can be taken on the folder. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- View. User can run a report or refresh a dashboard, but can’t edit them. All users have at least Viewer access to report and dashboard</td>
</tr>
</tbody>
</table>
folders that have been shared with them. (Some users may have administrative permissions that give them greater access.)

- **EditAllContents.** Users can view and modify the reports or dashboards in the folder, and move them to and from any other folders that they have equivalent access to.
- **Manage.** Users can do everything Viewers and Editors can do, plus control other users’ access to a folder.

#### Required. Specifies the user, group, or role that has the specified access level to the folder.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sharedTo</code></td>
<td>string</td>
<td>Required. Specifies the user, group, or role that has the specified access level to the folder.</td>
</tr>
<tr>
<td><code>sharedToType</code></td>
<td><code>FolderSharedToType</code> (enumeration of type string)</td>
<td>Required. Specifies the type of entity that the folder is shared with. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Group.</strong> Users in a specified public group have the specified access level to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Manager.</strong> Available in API version 29.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>ManagerAndSubordinatesInternal.</strong> Available in API version 29.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Role.</strong> Users with a specified role have the specified access level to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>RoleAndSubordinates.</strong> Users with a specified role, and users with a role subordinate to that role, have the specified access level to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>RoleAndSubordinatesInternal.</strong> Users with a specified role and users with a role subordinate to that role, except public portal users, have the specified access level to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Organization.</strong> All internal users have the specified access level to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Territory.</strong> Users in a specified territory have the specified access level to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>TerritoryAndSubordinates.</strong> Users in a specified territory, and users in territories subordinate to that, have the specified access level to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>AllPrmUsers.</strong> All PRM Portal users have the specified level of access to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>User.</strong> The specified individual user has the specified level of access to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>PartnerUser.</strong> The specified individual user of a partner portal has the specified level of access to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>AllCspUsers.</strong> All Customer Success Portal users have the specified level of access to the folder.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>CustomerPortalUser.</strong> The specified individual user of a customer portal has the specified level of access to the folder.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example of a FolderShare component for a dashboard folder:

```
<?xml version="1.0" encoding="UTF-8"?>
<DashboardFolder xmlns="http://soap.sforce.com/2006/04/metadata">
  <folderShares>
    <accessLevel>View</accessLevel>
    <sharedTo>R1</sharedTo>
    <sharedToType>Role</sharedToType>
  </folderShares>
</DashboardFolder>
```

This is an example of a FolderShare component for a report folder:

```
<?xml version="1.0" encoding="UTF-8"?>
<ReportFolder xmlns="http://soap.sforce.com/2006/04/metadata">
  <folderShares>
    <accessLevel>View</accessLevel>
    <sharedTo>R1</sharedTo>
    <sharedToType>Role</sharedToType>
  </folderShares>
</ReportFolder>
```

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

GlobalPicklist

Represents a global picklist, or the set of shared picklist values that custom picklist fields can use. (In contrast, the custom picklist fields that are based on a global picklist are of type CustomValue.) This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

GlobalPicklist components have the suffix .globalPicklist and are stored in the globalPicklist folder.
**Version**

GlobalPicklist components are available in API version 37.0 only. In API version 38.0 and later, GlobalPicklist is replaced by the GlobalValueSet on page 536 type.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>It’s useful to state the global picklist’s purpose, and which objects it’s intended for. Limit: 255 characters.</td>
</tr>
<tr>
<td>globalPicklistValues</td>
<td>GlobalPicklistValue on page 533</td>
<td>Requires at least one value. The list of values (or “picklist value set”) that’s defined for a global picklist. The picklist value set is inherited by any custom picklist field that’s based on that global picklist. Each value is of type GlobalPicklistValue. A global picklist can have up to 1,000 total values (inclusive of inactive values).</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. A global picklist’s name, which is defined when the global picklist is created. Appears as Label in the user interface.</td>
</tr>
<tr>
<td>sorted</td>
<td>string</td>
<td>Indicates whether a global picklist’s value set is sorted in alphabetical order. By default this value is false.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following Territories.globalPicklist is an example of a GlobalPicklist component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<GlobalPicklist xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>Updated:This is a basic global picklist</description>
  <globalPicklistValues>
    <fullName>Northwest</fullName>
    <default>false</default>
  </globalPicklistValues>
  <globalPicklistValues>
    <fullName>Northeast</fullName>
    <default>false</default>
  </globalPicklistValues>
  <globalPicklistValues>
    <fullName>South</fullName>
    <default>true</default>
  </globalPicklistValues>
  <globalPicklistValues>
    <fullName>Southwest</fullName>
    <default>false</default>
    <isActive>false</isActive>
  </globalPicklistValues>
  <masterLabel>Territories</masterLabel>
  <sorted>true</sorted>
</GlobalPicklist>
```
The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Territories</members>
    <name>GlobalPicklist</name>
  </types>
  <version>37.0</version>
</Package>
```

**GlobalPicklistValue**

Represents the definition of a value used in a global picklist. Custom picklist fields can inherit the picklist value set from a global picklist. This type extends the `Metadata` metadata type and inherits its `fullName` field.

**Note:** GlobalPicklistValue components don’t have file suffixes or directories because they’re lists of values and not custom fields. For file-based operations they can be accessed through `GlobalPicklist` (which is in API v37.0 only).

**Version**

GlobalPicklistValue components are available in API version 37.0 only. In API version 38.0 and later, GlobalPicklistValue is replaced by CustomValue on page 379.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>string</td>
<td>The color assigned to the picklist value when it’s used in charts on reports and dashboards. The color is in hexadecimal format; for example, #FF6600. If a color is not specified, it’s assigned dynamically upon chart generation.</td>
</tr>
<tr>
<td>default</td>
<td>boolean</td>
<td>Required. Indicates whether this value is the default selection for the global picklist and the custom picklists that share its picklist value set. This field is set to <code>true</code> by default.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The global picklist value’s description. It’s useful to include a description for a global picklist value so the reason for creating it can be tracked. Limit: 255 characters.</td>
</tr>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Indicates whether this value is currently active or inactive. The default value is <code>true</code>. Users can select only active values from a picklist. An API retrieve operation for global picklist values returns all active and inactive values in the picklist. (Meanwhile, retrieving the values of a non-global, unrestricted picklist returns only the active values.)</td>
</tr>
</tbody>
</table>
PicklistValue

This metadata type defines a value in the picklist and specifies whether this value is the default value. This type extends the GlobalPicklistValue metadata type and inherits all its fields. In API version 36.0 and earlier, PicklistValue extends the Metadata type and inherits its fullName field.

Note the following when working with picklist values:

- When you retrieve a standard object, all picklist values are retrieved, not just the customized picklist values.
- When you deploy changes to standard picklist fields, picklist values are added as needed.
- To deactivate a global picklist value, you can invoke an update() call on GlobalPicklist with the value omitted, or with the value’s isActive field set to false. Or, you can invoke an update() call directly on GlobalPicklistValue with the isActive field set to false.
- If picklist values are missing from a component definition, they get deactivated when deployed. Deactivation occurs for picklist values of both standard and custom fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowEmail</td>
<td>boolean</td>
<td>Indicates whether this value lets users email a quote PDF (true), or not (false). This field is only relevant for the Status field in quotes. This field is available in API version 18.0 and later.</td>
</tr>
<tr>
<td>closed</td>
<td>boolean</td>
<td>Indicates whether this value is associated with a closed status (true), or not (false). This field is only relevant for the standard Status field in cases and tasks. This field is available in API version 16.0 and up to version 36.0. In version 37.0, this field is in GlobalPicklistValue.</td>
</tr>
</tbody>
</table>
| controllingFieldValues | string[] | A list of values in the controlling field that are linked to this picklist value. The controlling field can be a checkbox or a picklist. This field is available in API version 14.0 and later. The values in the list depend on the field type: 
  - Checkbox: checked or unchecked.
  - Picklist: The fullname of the picklist value in the controlling field. |
| converted        | boolean    | Indicates whether this value is associated with a converted status (true), or not (false). This field is relevant for only the standard Lead Status field in leads. Your organization can set its own guidelines for determining when a lead is qualified, but typically, you want to convert a lead as soon as it becomes a real opportunity that you want to forecast. For more information, see “Convert Qualified Leads” in the Salesforce online help. This field is available in API version 16.0 and later. |
| cssExposed       | boolean    | Indicates whether this value is available in your Self-Service Portal (true), or not (false). This field is only relevant for the standard Case Reason field in cases. 

Self-Service provides an online support channel for your customers - allowing them to resolve their inquiries without contacting a customer service representative. For more information about Self-Service, see “Setting Up Your Self-Service Portal” in the Salesforce online help.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| forecastCategory | ForecastCategories (enumeration of type string) | Indicates whether this value is associated with a forecast category (true), or not (false). This field is only relevant for the standard Stage field in opportunities.  
- Omitted  
- Pipeline  
- BestCase  
- Forecast  
- Closed  
This field is available in API version 16.0 and later. |
| highPriority | boolean | Indicates whether this value is a high priority item (true), or not (false). This field is only relevant for the standard Priority field in tasks. For more information about tasks, see “Guidelines for Using Tasks” in the Salesforce online help. This field is available in API version 16.0 and later. |
| probability | int | Indicates whether this value is a probability percentage (true), or not (false). This field is only relevant for the standard Stage field in opportunities. This field is available in API version 16.0 and later. |
| reverseRole | string | A picklist value corresponding to a reverse role name for a partner. If the role is “subcontractor”, then the reverse role might be “general contractor”. Assigning a partner role to an account in Salesforce creates a reverse partner relationship so that both accounts list the other as a partner. This field is only relevant for partner roles.  
For more information, see “Partner Fields” in the Salesforce online help.  
This field is available in API version 18.0 and later. |
| reviewed | boolean | Indicates whether this value is associated with a reviewed status (true), or not (false). This field is only relevant for the standard Status field in solutions. For more information about opportunities, see “Creating Solutions” in the Salesforce online help. This field is available in API version 16.0 and later. |
| won | boolean | Indicates whether this value is associated with a closed or won status (true), or not (false). This field is only relevant for the standard Stage field in opportunities. This field is available in API version 16.0 and later. |
Declarative Metadata Sample Definition

For an example of GlobalPicklistValue components with a package.xml that references them, see GlobalPicklist.

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

GlobalValueSet

Represents the metadata for a global picklist value set, which is the set of shared values that custom picklist fields can use. A global value set isn’t a field itself. (In contrast, the custom picklist fields that are based on a global picklist are of type ValueSet.) This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

GlobalValueSet components have the suffix .globalValueSet and are stored in the globalValueSets folder.

Version

GlobalValueSet components are available in API version 38.0 and later. In API version 37.0, this is the GlobalPicklist type.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customValue</td>
<td>CustomValue[]</td>
<td>Requires at least one value. The list of values (or &quot;global value set&quot;) that’s defined for a global picklist. The global value set is inherited by any custom picklist field that uses that value set. Each value is of type customValue. A global value set can have up to 1,000 total values (inclusive of inactive values).</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>It’s useful to state the global value set’s purpose, and which objects it’s intended for. Limit: 255 characters.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. A global value set’s name, which is defined when the global value set is created. Appears as Label in the user interface.</td>
</tr>
<tr>
<td>sorted</td>
<td>boolean</td>
<td>Required. Indicates whether a global value set is sorted in alphabetical order. By default this value is false.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following UpsellGlobal.globalValueSet is an example of a GlobalValueSet component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<GlobalValueSet xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>Updated:This is a basic global value set.</description>
</GlobalValueSet>
```
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
<types>
  <members>UpsellGlobal</members>
  <name>GlobalValueSet</name>
</types>
<version>40.0</version>
</Package>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](#).

### GlobalValueSetTranslation

Contains details for a global value set translation. Global value sets are lists of values that can be shared by multiple custom picklist fields, optionally across objects. This type extends the Metadata metadata type and inherits its fullName field.

### File Suffix and Directory Location

GlobalValueSetTranslation components have the suffix `.globalValueSetTranslation` and are stored in the `globalValueSetTranslations` folder.

Translations are stored in a file with a format of `ValueSetName-lang.globalValueSetTranslation`, where `ValueSetName` is the global value set’s name, and `lang` is the translation language.

### Version

GlobalValueSetTranslation components are available in API version 38.0 and later.
**Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>valueTranslation</td>
<td>ValueTranslation[]</td>
<td>The translated name of a value in a translated global value set. Each valueTranslation is paired with a masterLabel, which is the original (untranslated) name of the value.</td>
</tr>
</tbody>
</table>

**ValueTranslation**

The original value name and the translated value name in a translated global value set.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The original (untranslated) name of a value in a global value set. Each valueTranslation has a masterLabel paired with its translation.</td>
</tr>
<tr>
<td>translation</td>
<td>string</td>
<td>The translated name of a value in a translated global value set.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example of a GlobalValueSetTranslation component. When a value isn’t translated, its translation becomes a comment that’s paired with its masterLabel.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<GlobalValueSetTranslation xmlns="http://soap.sforce.com/2006/04/metadata">
  <valueTranslation>
    <masterLabel>Three</masterLabel>
    <translation>Trois</translation>
  </valueTranslation>
  <valueTranslation>
    <masterLabel>Four</masterLabel>
    <translation>Quatre</translation>
  </valueTranslation>
  <valueTranslation>
    <masterLabel>Five</masterLabel>
    <translation><!-- Five --></translation>
  </valueTranslation>
</GlobalValueSetTranslation>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Numbers-fr</members>
    <name>GlobalValueSetTranslation</name>
  </types>
  <version>38.0</version>
</Package>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
  - Translations

Group

Represents a set of public groups, which can have users, roles, and other groups.

Declarative Metadata File Suffix and Directory Location

The file suffix for group components is .group and components are stored in the groups directory of the corresponding package directory.

Version

Group components are available in API version 24.0 and later.

Fields

Note: Members of the public group are not migrated when you deploy the group type.

This metadata type represents the valid values that define a group:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doesIncludeBosses</td>
<td>boolean</td>
<td>Indicates whether the managers have access (true) or do not have access (false) to records shared with members of the group. This field is only available for public groups.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The unique identifier for API access. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the Metadata component. Corresponds to Group Name in the user interface.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the group. Corresponds to Label in the user interface.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is the definition of a group.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Group xmlns="http://soap.sforce.com/2006/04/metadata">
  <doesIncludeBosses>true</doesIncludeBosses>
  <fullName>admin</fullName>
  <name>test</name>
</Group>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

HomePageComponent

Represents the metadata associated with a home page component. You can customize the Home tab in Salesforce Classic to include components such as sidebar links, a company logo, a dashboard snapshot, or custom components that you create. Use to create, update, or delete home page component definitions.

For more information, see “Customize Salesforce Classic Home Tab Page Layouts” in the Salesforce online help. This type extends the Metadata metadata type and inherits its fullName field.

Declarative Metadata File Suffix and Directory Location

The file suffix for home page components is .HomePageComponent and components are stored in the homepagecomponents directory of the corresponding package directory.

Version

Home page components are available in API version 12.0 and later.

HomePageComponent

This metadata type represents the valid values that define a home page component:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>string</td>
<td>If this is an HTML page component, this is the body of the HTML.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The name can only contain characters, letters, and the underscore (_) character, must start with a letter, and cannot end with an underscore or contain two consecutive underscore characters. Inherited from the Metadata component, this field is not defined in the WSDL for this component. It must be specified when creating, updating, or deleting. See create() to see an example of this field specified for a call.</td>
</tr>
</tbody>
</table>

540
**Metadata Types**

### HomePageComponent

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>int</td>
<td>Required for Visualforce Area components. Indicates the height (in pixels) of the component. This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>links</td>
<td>string[]</td>
<td>If the pageComponentType is links, then zero or more names of custom page links can be specified.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ObjectWebLink</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CustomPageWebLink</td>
</tr>
<tr>
<td>page</td>
<td>string</td>
<td>This field is only available for Visualforce Area components and indicates the API name of the Visualforce page that is associated with the component. This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>pageComponentType</td>
<td>PageComponentType (enumeration of type string)</td>
<td>Required. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• links</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• htmlArea</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• imageOrNote</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• visualforcePage (This value is available in API version 31.0 and later.)</td>
</tr>
<tr>
<td>showLabel</td>
<td>boolean</td>
<td>This field is only available for Visualforce Area components and specifies whether the component displays with a label (true) or not (false). This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>showScrollbars</td>
<td>boolean</td>
<td>This field is only available for Visualforce Area components and specifies whether the component displays with scrollbars (true) or not (false). This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>width</td>
<td>PageComponentWidth (enumeration of type string)</td>
<td>This field is only available for HTML and Visualforce Area components, and indicates whether this is a narrow or wide home page component. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• narrowComponents</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• wideComponents</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is the definition of a home page component. See HomePageLayout and WebLink for related samples.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<HomePageComponent xmlns="http://soap.sforce.com/2006/04/metadata">
  <height>200</height>
  <page>MyVisualforcePage</page>
  <pageComponentType>visualforcePage</pageComponentType>
</HomePageComponent>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- HomePageLayout
- WebLink

HomePageLayout

Represents the metadata associated with a home page layout. You can customize home page layouts and assign the layouts to users based on their user profile.

For more information, see “Customize Salesforce Classic Home Tab Page Layouts” in the Salesforce online help.

File Suffix and Directory Location

Home page layouts are stored in the homePageLayouts directory of the corresponding package directory. The extension is .homePageLayout.

Version

Home page components are available in API version 12.0 and later. This type extends the Metadata metadata type and inherits its fullName field.

Fields

This metadata type represents the valid values that define a home page layout:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fullName</td>
<td>string</td>
<td>The name can only contain characters, letters, and the underscore (_) character, must start with a letter, and cannot end with an underscore or contain two consecutive underscore characters. Inherited from the Metadata component, this field is not defined in the WSDL for this component. It must be specified when creating, updating, or deleting. See create() to see an example of this field specified for a call.</td>
</tr>
<tr>
<td>narrowComponents</td>
<td>string[]</td>
<td>The list of elements in the narrow column on the left side of the home page.</td>
</tr>
</tbody>
</table>
### InstalledPackage

Represents a package to be installed or uninstalled. Deploying a newer version of a currently installed package upgrades the package.

**Note:** You can’t deploy a package along with other metadata types. When you deploy InstalledPackage, it must be the only metadata type specified in the manifest file.

### File Suffix and Directory Location

The package is specified in the `installedPackages` directory, in a file named after the package’s namespace prefix. The file extension is `.installedPackage`.

### Version

InstalledPackage is available in API version 28.0 and later.
### Fields

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Description</th>
<th>Required</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>activateRSS</td>
<td>boolean</td>
<td>true</td>
<td>Determines the state of Remote Site Settings (RSS) and Content Security Policy (CSP) at the time of installing the package and must be set to either of these values.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>false</td>
<td>Override the isActive state of any RSS or CSP in the package and set it to false. The default value is false. Available in API version 43.0 and later.</td>
</tr>
<tr>
<td>password</td>
<td>string</td>
<td>true</td>
<td>Specifies the package password.</td>
</tr>
<tr>
<td>versionNumber</td>
<td>string</td>
<td>true</td>
<td>Required. The version number of the package. The version number has the format majorNumber.minorNumber.patchNumber (for example, 2.1.3).</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following example specifies a sample package to be installed or uninstalled.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<InstalledPackage xmlns="http://soap.sforce.com/2006/04/metadata">
  <versionNumber>1.0</versionNumber>
  <password>optional_password</password>
  <activateRSS>true</activateRSS>
</InstalledPackage>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](#).

### KeywordList

Represents a list of keywords used in community moderation. This keyword list is a type of moderation criteria that defines offensive language or inappropriate content that you don’t want in your community. This type extends the Metadata metadata type and inherits its fullName field.

Keep the following things in mind when creating keyword list criteria:

- Your org can have up to 30 keyword list criteria. This limit is per org, not per community.
- A keyword list can have up to 2,000 keywords.
- Capitalization and trailing punctuation are ignored when matching your keywords to user-generated content. For example, if your criteria includes BadWord, it’s matched when a user types BADWORD or badword.
File Suffix and Directory Location

KeywordList components have the suffix .keywords and are stored in the moderation directory of the corresponding package directory. The file name format follows community_name.keyword_list_developer_name.keywords.

Version

KeywordList components are available in API version 36.0 and later.

Special Access Rules

To view, create, edit, and delete a keyword list, you need the Manage Communities or Create and Set Up Communities permission.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>string</td>
<td>A description of the keyword list.</td>
</tr>
<tr>
<td>keywords</td>
<td>Keyword[]</td>
<td>The keywords you want moderate in your community.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Label for the keyword list.</td>
</tr>
</tbody>
</table>

Keyword

Keywords in the keyword list.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>keyword</td>
<td>string</td>
<td>Required. Keywords you want to moderate.</td>
</tr>
</tbody>
</table>

- Keywords can only be up to 100 characters and can include letters, numbers, spaces, and special characters.
- Wildcard characters aren't supported.

Declarative Metadata Sample Definition

The following is an example of a KeywordList component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<KeywordList xmlns="http://soap.sforce.com/2006/04/metadata">
  <masterLabel>Bad Word List</masterLabel>
  <description>List of bad words updated by Joe in Nov 2015.</description>
  <keywords>
    <keyword>bad-word</keyword>
  </keywords>
  <keywords>
    <keyword>bad word</keyword>
  </keywords>
</KeywordList>
```
The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <name>KeywordList</name>
    <members>community1.badword_list</members>
  </types>
  <version>36.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**Layout**

Represents the metadata associated with a page layout. For more information, see “Page Layouts” in the Salesforce Help.

This type extends the Metadata metadata type and inherits its `fullName` field.

**Note:** To edit the Ideas layout, specify it by name in the `package.xml` file. In `package.xml`, use the following code to retrieve the Ideas layout:

```xml
<types>
  <members>Idea-Idea Layout</members>
  <name>Layout</name>
</types>
```

**File Suffix and Directory Location**

Layouts are stored in the `layouts` directory of the corresponding package directory. The extension is `.layout`.

**Note:** Retrieving a component of this metadata type in a project makes the component appear in any Profile and PermissionSet components that are retrieved in the same package.

**Version**

Layouts are available in API version 13.0 and later.

**Fields**

This metadata type represents the valid values that define a page layout.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customButtons</td>
<td>string[]</td>
<td>The custom buttons for this layout. Each button is a reference to a WebLink on the same object. For example, a ButtonLink refers to a Weblink on the same standard or custom object named 'ButtonLink'.</td>
</tr>
<tr>
<td>customConsoleComponents</td>
<td>CustomConsoleComponents</td>
<td>Represents custom console components (Visualforce pages, lookup fields, or related lists; Canvas apps not available) on a page layout. Custom console components only display in the Salesforce console.</td>
</tr>
<tr>
<td>emailDefault</td>
<td>boolean</td>
<td>Only relevant if showEmailCheckbox is set; indicates the default value of that checkbox.</td>
</tr>
<tr>
<td>excludeButtons</td>
<td>string[]</td>
<td>List of standard buttons to exclude from this layout. For example, &lt;excludeButtons&gt;Delete&lt;/excludeButtons&gt; excludes the Delete button from this layout.</td>
</tr>
<tr>
<td>feedLayout</td>
<td>FeedLayout</td>
<td>Represents the values that define the feed view of a feed-based page layout. Feed-based layouts are available on Account, Case, Contact, Lead, Opportunity, custom, and external objects. They include a feed view and a detail view.</td>
</tr>
<tr>
<td>headers</td>
<td>LayoutHeader[]</td>
<td>Layout headers are currently only used for tagging, and only appear in the UI if tagging is enabled. Valid string values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• PersonalTagging—tag is private to user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PublicTagging—tag is viewable any other user who can access the record.</td>
</tr>
<tr>
<td>layoutSections</td>
<td>LayoutSection[]</td>
<td>The main sections of the layout containing fields, s-controls, and custom links. The order here determines the layout order.</td>
</tr>
<tr>
<td>miniLayout</td>
<td>MiniLayout</td>
<td>A mini layout is used in the mini view of a record in the Console tab, hover details, and event overlays.</td>
</tr>
<tr>
<td>multilineLayoutFields</td>
<td>string[]</td>
<td>Fields for the special multiline layout fields which appear in OpportunityProduct layouts. These fields are otherwise similar to miniLayoutFields miniLayout.</td>
</tr>
<tr>
<td>platformActionList</td>
<td>PlatformActionList</td>
<td>The list of actions, and their order, that display in the Salesforce app action bar for the layout. This field is available in API version 34.0 and later.</td>
</tr>
<tr>
<td>quickActionList</td>
<td>QuickActionList</td>
<td>The list of quick actions that display in the full Salesforce site for the page layout. This field is available in API version 28.0 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>relatedContent</td>
<td>RelatedContent</td>
<td>The Related Content section of the page layout. This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>relatedLists</td>
<td>RelatedListItem[]</td>
<td>The related lists for the layout, listed in the order they appear in the user interface.</td>
</tr>
<tr>
<td>relatedObjects</td>
<td>string[]</td>
<td>The list of related objects that appears in the mini view of the console. In database terms, these objects are foreign key fields on the object for the layout. For more information, see “Choose Related Objects for the Agent Console’s Mini View” in the Salesforce Help.</td>
</tr>
<tr>
<td>runAssignmentRulesDefault</td>
<td>boolean</td>
<td>Only relevant if showRunAssignmentRulesCheckbox is set; indicates the default value of that checkbox.</td>
</tr>
<tr>
<td>showEmailCheckbox</td>
<td>boolean</td>
<td>Only allowed on Case, CaseClose, and Task layouts. If set, a checkbox appears to show email.</td>
</tr>
<tr>
<td>showHighlightsPanel</td>
<td>boolean</td>
<td>If set, the highlights panel displays on pages in the Salesforce console. This field is available in API version 22.0 and later.</td>
</tr>
<tr>
<td>showInteractionLogPanel</td>
<td>boolean</td>
<td>If set, the interaction log displays on pages in the Salesforce console. This field is available in API version 22.0 and later.</td>
</tr>
<tr>
<td>showKnowledgeComponent</td>
<td>boolean</td>
<td>Only allowed on Case layouts. If set, the Knowledge sidebar displays on cases in the Salesforce console. This field is available in API version 20.0 and later.</td>
</tr>
<tr>
<td>showRunAssignmentRulesCheckbox</td>
<td>boolean</td>
<td>Only allowed on Lead and Case objects. If set, a checkbox appears on the page to show assignment rules.</td>
</tr>
<tr>
<td>showSolutionSection</td>
<td>boolean</td>
<td>Only allowed on CaseClose layout. If set, the built-in solution information section shows up on the page.</td>
</tr>
<tr>
<td>showSubmitAndAttachButton</td>
<td>boolean</td>
<td>Only allowed on Case layout. If set, the Submit &amp; Add Attachment button displays on case edit pages to portal users in the Customer Portal.</td>
</tr>
<tr>
<td>summaryLayout</td>
<td>SummaryLayout</td>
<td>Controls the appearance of the highlights panel in Salesforce Classic, which summarizes key fields in a grid at the top of a page layout, when Case Feed is enabled. This field is available in API version 18.0 and later.</td>
</tr>
</tbody>
</table>

**CustomConsoleComponents**

Represents custom console components (Visualforce pages, lookup fields, or related lists; Canvas apps not available) on a page layout. Custom console components only display in the Salesforce console. Available in API version 25.0 and later.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>primaryTabComponents</td>
<td>PrimaryTabComponents</td>
<td>Represents custom console components on primary tabs in the Salesforce console. Available in API version 25.0 and later.</td>
</tr>
<tr>
<td>subtabComponents</td>
<td>SubtabComponents</td>
<td>Represents custom console components on subtabs in the Salesforce console. Available in API version 25.0 and later.</td>
</tr>
</tbody>
</table>

### PrimaryTabComponents

Represents custom console components on primary tabs in the Salesforce console. Available in API version 25.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>component</td>
<td>ConsoleComponent[]</td>
<td>Represents a custom console component (Visualforce page, lookup field, or related lists; Canvas apps not available) on a section of a page layout. Custom console components only display in the Salesforce console. This field is available in API version 29.0 and earlier.</td>
</tr>
<tr>
<td>containers</td>
<td>Container[]</td>
<td>Represents a location and style in which to display more than one custom console component on the sidebars of the Salesforce console. You can specify up to five components for each of the four locations (left, right, top, and bottom). This field is available in API version 30.0 and later.</td>
</tr>
</tbody>
</table>

### ConsoleComponent

Represents a custom console component (Visualforce page, lookup field, or related lists; Canvas apps not available) on a section of a page layout. Custom console components only display in the Salesforce console. Available in API version 25.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>int</td>
<td>Required for components with a location of top or bottom. The height of the custom console component. The value must be specified in pixels and be greater than 0 but less than 999.</td>
</tr>
<tr>
<td>location</td>
<td>string</td>
<td>Required. The location of the custom console component on the page layout. Valid values are right, left, top, and bottom. A component can have one location for each page layout.</td>
</tr>
<tr>
<td>visualforcePage</td>
<td>string</td>
<td>Required. The unique name of the custom console component. For example, ConsoleComponentPage.</td>
</tr>
<tr>
<td>width</td>
<td>int</td>
<td>Required for components with a location of left or right. The width of the custom console component. The value must be specified in pixels and be greater than 0 but less than 999.</td>
</tr>
</tbody>
</table>
Container

Represents a location and style in which to display more than one custom console component in the sidebars of the Salesforce console. For example, you can display multiple components in the right sidebar of the console with a style of either stack, tabs, or accordion. Available in API version 30.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>int</td>
<td>Required for components with a location of top or bottom. The height of the components' container. The unit field determines the unit of measurement, in pixels or percent.</td>
</tr>
<tr>
<td>isContainerAutoSizeEnabled</td>
<td>boolean</td>
<td>Required. If set to true, stacked console components in the sidebars autosize vertically. Set to true by default for newly created console components. Available in API version 32.0 and later.</td>
</tr>
<tr>
<td>region</td>
<td>string</td>
<td>Required. The location of the components' container. Valid values include: right, left, top, bottom.</td>
</tr>
<tr>
<td>sidebarComponents</td>
<td>SidebarComponent[]</td>
<td>Represents a specific custom console component to display in the components' container.</td>
</tr>
<tr>
<td>style</td>
<td>string</td>
<td>Required. The style of the container in which to display multiple components. Valid values include: stack—a content area with multiple frames, tabs—a single content area with a list of multiple panels, accordion—a collapsible content area.</td>
</tr>
<tr>
<td>unit</td>
<td>string</td>
<td>Required. The unit of measurement, in pixels or percent, for the height or width of the components’ container. Pixel values are simply the number of pixels, for example, 500, and must be greater than 0 but less than 999. Percentage values must include the percent sign, for example, 20%, and must be greater than 0 but less than 100.</td>
</tr>
<tr>
<td>width</td>
<td>int</td>
<td>Required for components with a location of right or left. The width of the components’ container. The unit field determines the unit of measurement, in pixels or percent.</td>
</tr>
</tbody>
</table>

SidebarComponent

Represents a specific custom console component to display in a container that hosts multiple components in one of the sidebars of the Salesforce console. You can specify up to five components for each of the four container locations (left, right, top, and bottom). Available in API version 30.0 and later.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentType</td>
<td>string</td>
<td>Specifies the component type. Valid values are KnowledgeOne, Lookup, Milestones, RelatedList, Topics, Files, and CaseExperts. This field is available in API version 31.0 and later. The Files and CaseExperts values are available in API version 32.0 and later.</td>
</tr>
<tr>
<td>createAction</td>
<td>string</td>
<td>If the component is a lookup field, the name of the quick action used to create a record. This field is available in API version 42.0 and later.</td>
</tr>
<tr>
<td>enableLinking</td>
<td>boolean</td>
<td>If the component is a lookup field, lets users associate a record with this field. This field is available in API version 42.0 and later.</td>
</tr>
<tr>
<td>height</td>
<td>int</td>
<td>Required for components with a location of top or bottom. The height of the component in the container. The unit field determines the unit of measurement, in pixels or percent.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The name of the component as it displays to console users. Available for components in a container with the style of tabs or accordion.</td>
</tr>
<tr>
<td>lookup</td>
<td>string</td>
<td>If the component is a lookup field, the name of the field.</td>
</tr>
<tr>
<td>page</td>
<td>string</td>
<td>If the component is a Visualforce page, the name of the Visualforce page.</td>
</tr>
<tr>
<td>relatedlists</td>
<td>RelatedList[]</td>
<td>If the component is a related list, the name of the list. This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>unit</td>
<td>string</td>
<td>The unit of measurement, in pixels or percent, for the height or width of the component in the container.</td>
</tr>
<tr>
<td>updateAction</td>
<td>string</td>
<td>If the component is a lookup field, the name of the quick action used to update a record. This field is available in API version 42.0 and later.</td>
</tr>
<tr>
<td>width</td>
<td>int</td>
<td>Required for components with a location of right or left. The width of the component in the container. The unit field determines the unit of measurement, in pixels or percent.</td>
</tr>
</tbody>
</table>

**RelatedList**

Represents related list custom components on the sidebars of the Salesforce console. Available in API version 31.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hideOnDetail</td>
<td>boolean</td>
<td>If set to true, the related list is hidden from detail pages where it appears as a component to prevent duplicate information from showing.</td>
</tr>
</tbody>
</table>
### SubtabComponents

Represents custom console components on subtabs in the Salesforce console. Available in API version 25.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the component as it appears to console users.</td>
</tr>
<tr>
<td>component</td>
<td>ConsoleComponent[]</td>
<td>Represents a custom console component (Visualforce page, lookup field, or related lists; Canvas apps not available) on a section of a page layout. Custom console components only display in the Salesforce console. This field is available in API version 29.0 and earlier.</td>
</tr>
<tr>
<td>containers</td>
<td>Container[]</td>
<td>Represents a location and style in which to display more than one custom console component on the sidebars of the Salesforce console. You can specify up to five components for each of the four locations (left, right, top, and bottom). This field is available in API version 30.0 and later.</td>
</tr>
</tbody>
</table>

### FeedLayout

Represents the values that define the feed view of a feed-based page layout. Feed-based layouts are available on Account, Case, Contact, Lead, Opportunity, custom, and external objects. They include a feed view and a detail view. Available in API version 30.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>autocollapsePublisher</td>
<td>boolean</td>
<td>Specifies whether the publisher is automatically collapsed when the page loads (true) or not (false).</td>
</tr>
<tr>
<td>compactFeed</td>
<td>boolean</td>
<td>Specifies whether the feed-based page layout uses a compact feed (true) or not (false). If set to true, feed items on the page are collapsed by default, and the feed view has an updated design.</td>
</tr>
</tbody>
</table>
| feedFilterPosition       | FeedLayoutFilterPosition (enumeration of type string) | Where the feed filters list is included in the layout.  Valid values are:  
- centerDropDown—as a dropdown list in the center column.  
- leftFixed—as a fixed list in the left column.  
- leftFloat—as a floating list in the left column. |
| feedFilters              | FeedLayoutFilter[]  | The individual filters displayed in the feed filters list.                                                                                |
| fullWidthFeed            | boolean             | Specifies whether the feed expands horizontally to take up all available space on the page (true) or not (false).                            |
| hideSidebar              | boolean             | Specifies whether the sidebar is hidden (true) or not (false).                                                                            |
| leftComponents           | FeedLayoutComponent[] | The individual components displayed in the left column of the feed view.                                                               |
| rightComponents          | FeedLayoutComponent[] | The individual components displayed in the right column of the feed view.                                                              |
FeedLayoutComponent

Represents a component in the feed view of a feed-based page layout. Available in API version 30.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentType</td>
<td>FeedLayoutComponentType (enumeration of type string)</td>
<td>Required. The type of component. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HelpAndToolLinks—icons that link to the help topic for the page, the page layout, and, the printable view of the page. Available only on Case layouts.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CustomButtons—a custom button.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Following—an icon that toggles between a Follow button (if the user viewing a record doesn’t already follow it) and a Following indicator (if the user viewing a record does follow it).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Followers—a list of users who follow the record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CustomLinks—a custom link.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Milestones—the milestone tracker, which lets users see the status of a milestone on a case. Available only on Case layouts.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Topics—a list of topics related to the record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CaseUnifiedFiles—a list of all files that are attached to the case.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Visualforce—a custom Visualforce component.</td>
</tr>
<tr>
<td>height</td>
<td>int</td>
<td>The height, in pixels, of the component. Doesn’t apply to standardComponents</td>
</tr>
<tr>
<td>page</td>
<td>string</td>
<td>The name of a Visualforce page being used as a custom component.</td>
</tr>
</tbody>
</table>

FeedLayoutFilter

Represents a feed filter option in the feed view of a feed-based page layout. A filter must have only standardFilter or feedItemType set. Available in API version 30.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>feedFilterName</td>
<td>string</td>
<td>The name of a CustomFeedFilter component. Names are prefixed with the name of the parent object. For example, Case.MyCustomFeedFilter.</td>
</tr>
<tr>
<td>feedFilterType</td>
<td>FeedLayoutFilterType (enumeration of type string)</td>
<td>The type of filter. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AllUpdates—shows all feed items on a record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FeedItemType—shows feed items only for a particular type of activity on the record.</td>
</tr>
<tr>
<td>feedItemType</td>
<td>FeedItemType (enumeration of type string)</td>
<td>The type of feed item to display. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ActivityEvent—feed items related to activity on tasks and events associated with a case. Available only on Case layouts.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AdvancedTextPost</td>
<td>Feed items related to group announcements posted on a feed. This value is available in API version 31.0 and later.</td>
<td></td>
</tr>
<tr>
<td>AnnouncementPost</td>
<td>Not used.</td>
<td></td>
</tr>
<tr>
<td>ApprovalPost</td>
<td>Feed items related to approvals that are submitted on a feed.</td>
<td></td>
</tr>
<tr>
<td>AttachArticleEvent</td>
<td>Feed items for activity related to attaching articles to cases. Available only on Case layouts.</td>
<td></td>
</tr>
<tr>
<td>BasicTemplateFeedItem</td>
<td>Not used.</td>
<td></td>
</tr>
<tr>
<td>CallLogPost</td>
<td>Feed items for activity from the Log a Call action. Available only on layouts for objects that support Activities (tasks and events).</td>
<td></td>
</tr>
<tr>
<td>CanvasPost</td>
<td>Feed items related to posts that a canvas app makes on a feed.</td>
<td></td>
</tr>
<tr>
<td>CaseCommentPost</td>
<td>Feed items for activity from the Case Note action. Available only on Case layouts.</td>
<td></td>
</tr>
<tr>
<td>ChangeStatusPost</td>
<td>Feed items for activity from the Change Status action. Available only on Case layouts.</td>
<td></td>
</tr>
<tr>
<td>ChatTranscriptPost</td>
<td>Feed items for activity related to attaching Chat transcripts to cases. Available only on Case layouts.</td>
<td></td>
</tr>
<tr>
<td>CollaborationGroupCreated</td>
<td>Feed items related to creating a public group.</td>
<td></td>
</tr>
<tr>
<td>CollaborationGroupUnarchived</td>
<td>Not used.</td>
<td></td>
</tr>
<tr>
<td>ContentPost</td>
<td>Feed items related to attaching a file to a post.</td>
<td></td>
</tr>
<tr>
<td>CreatedRecordEvent</td>
<td>Feed items related to creating a record from the publisher.</td>
<td></td>
</tr>
<tr>
<td>DashboardComponentSnapshot</td>
<td>Feed items related to posting a dashboard snapshot on a feed.</td>
<td></td>
</tr>
<tr>
<td>EmailMessageEvent</td>
<td>Feed items for activity from the Email action. Available only on Case layouts.</td>
<td></td>
</tr>
<tr>
<td>FacebookPost</td>
<td>Not used.</td>
<td></td>
</tr>
<tr>
<td>LinkPost</td>
<td>Feed items related to attaching a URL to a post.</td>
<td></td>
</tr>
<tr>
<td>MilestoneEvent</td>
<td>Feed items for changes to the milestone status on a case. Available only on Case layouts.</td>
<td></td>
</tr>
<tr>
<td>PollPost</td>
<td>Feed items related to posting a poll on a feed.</td>
<td></td>
</tr>
<tr>
<td>ProfileSkillPost</td>
<td>Feed items related to skills added to a user’s Chatter profile. This value is available in API version 31.0 and later.</td>
<td></td>
</tr>
<tr>
<td>QuestionPost</td>
<td>Feed items related to posting a question on a feed. This value is available in API version 31.0 and later.</td>
<td></td>
</tr>
<tr>
<td>ReplyPost</td>
<td>Feed items for activity from the Portal action. Available only on Case layouts.</td>
<td></td>
</tr>
</tbody>
</table>
### Metadata Types

#### RypplePost
- Feed items related to creating a Thanks badge in Work.com.

#### SocialPost
- Feed items for activity on Twitter from the Social Post action.

#### TextPost
- Feed items for creating a text post from the publisher.

#### TrackedChange
- Feed items related to a change or group of changes to a tracked field.

- **UserStatus**—Not used.

---

#### MiniLayout

Represents a mini view of a record in the Console tab, hover details, and event overlays.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>string[]</td>
<td>The fields for the mini-layout, listed in the order they appear in the UI. Fields that appear here must appear in the main layout.</td>
</tr>
<tr>
<td>relatedLists</td>
<td>RelatedListItem[]</td>
<td>The mini related list, listed in the order they appear in the UI. You cannot set sorting on mini related lists. Fields that appear here must appear in the main layout.</td>
</tr>
</tbody>
</table>

---

#### LayoutSection

LayoutSection represents a section of a page layout, such as the Custom Links section.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customLabel</td>
<td>boolean</td>
<td>Indicates if this section's label is custom or standard (built-in). Custom labels can be any text, but must be translated. Standard labels have a predefined set of valid values, for example 'System Information', which are automatically translated.</td>
</tr>
<tr>
<td>detailHeading</td>
<td>boolean</td>
<td>Controls if this section appears in the detail page. In the UI, this setting corresponds to the checkbox in the section details dialog.</td>
</tr>
<tr>
<td>editHeading</td>
<td>boolean</td>
<td>Controls if this section appears in the edit page.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label; either standard or custom, based on the customLabel flag.</td>
</tr>
<tr>
<td>layoutColumns</td>
<td>LayoutColumn[]</td>
<td>The columns of the layout, depending on the style. 1, 2, or 3 columns, ordered left to right, are possible.</td>
</tr>
<tr>
<td>style</td>
<td>LayoutSectionStyle (enumeration of type string)</td>
<td>The style of the layout:</td>
</tr>
</tbody>
</table>

- **TwoColumnsTopToBottom** - Two columns, tab goes top to bottom
### LayoutColumn

LayoutColumn represents the items in a column within a layout section.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layoutItems</td>
<td>LayoutItem[]</td>
<td>The individual items within a column (ordered from top to bottom).</td>
</tr>
<tr>
<td>reserved</td>
<td>string</td>
<td>This field is reserved for Salesforce. The field resolves an issue with some SOAP libraries. Any value entered in the field is ignored.</td>
</tr>
</tbody>
</table>

### LayoutItem

LayoutItem represents the valid values that define a layout item. An item must have only one of the following values set: component, customLink, field, s-control, page, analyticsCloudComponent, or reportChartComponent.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| behavior    | UIBehavior (enumeration of type string) | Determines the field behavior. **Note:** KAVs, attempting to explicitly specify UIBehavior will result in an exception. UIBehavior must not be specified. Valid string values:  
  - **Edit**—The layout field can be edited but is not required  
  - **Required**—The layout field can be edited and is required  
  - **ReadOnly**—The layout field is read-only |
| canvas      | string                               | Reference to a canvas app.  
This field is available in API version 31.0 and later. |
| component   | string                               | Reference to a component. Value must be sfa:socialCard.  
This field is available in API version 30.0 and later. This is only allowed inside a RelatedContentItem. sfa:socialCard is only supported on page layouts for contacts, accounts, and leads. |
<p>| customLink  | string                               | The customLink reference. This is only allowed inside a CustomLink layoutSection. |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emptySpace</td>
<td>boolean</td>
<td>Controls if this layout item is a blank space.</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>The field name reference, relative to the layout object, for example Description or MyField__c.</td>
</tr>
<tr>
<td>height</td>
<td>int</td>
<td>For s-control and pages only, the height in pixels.</td>
</tr>
<tr>
<td>page</td>
<td>string</td>
<td>Reference to a Visualforce page.</td>
</tr>
<tr>
<td>analyticsCloudComponent</td>
<td>string</td>
<td>Refers to an Analytics dashboard that you can add to a standard or custom object page. This field is available in API version 34.0 and later.</td>
</tr>
<tr>
<td>reportChartComponent</td>
<td>string</td>
<td>Refers to a report chart that you can add to a standard or custom object page.</td>
</tr>
<tr>
<td>scontrol</td>
<td>string</td>
<td>Reference to an s-control.</td>
</tr>
<tr>
<td>showLabel</td>
<td>boolean</td>
<td>For s-control and pages only, whether to show the label.</td>
</tr>
<tr>
<td>showScrollbars</td>
<td>boolean</td>
<td>For s-control and pages only, whether to show scrollbars.</td>
</tr>
<tr>
<td>width</td>
<td>string</td>
<td>For s-control and pages only, the width in pixels or percent. Pixel values are simply the number of pixels, for example, 500. Percentage values must include the percent sign, for example, 20%.</td>
</tr>
</tbody>
</table>

### AnalyticsCloudComponentLayoutItem

Represents the settings for a Analytics dashboard on a standard or custom page. Available in API version 34.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assetType</td>
<td>string</td>
<td>Required. Specifies the type of Analytics asset to add. The available asset type is dashboard.</td>
</tr>
<tr>
<td>devName</td>
<td>string</td>
<td>Required. Unique development name of the dashboard to add.</td>
</tr>
<tr>
<td>error</td>
<td>string</td>
<td>Error string; only populated if an error occurred in the underlying dashboard.</td>
</tr>
<tr>
<td>filter</td>
<td>string</td>
<td>Communicates initial dashboard filters for mapping data fields in the dashboard to the object’s fields, so that the dashboard shows only the data that’s relevant for the record being viewed.</td>
</tr>
<tr>
<td>height</td>
<td>int</td>
<td>Specifies the height of the dashboard, in pixels. The default is 400.</td>
</tr>
<tr>
<td>hideOnError</td>
<td>boolean</td>
<td>Controls whether users see a dashboard that has an error. When this attribute is set to true, if the dashboard has an error, the dashboard doesn’t appear on the page. When set to false, the dashboard appears but doesn’t show any data except the error. An error can happen when</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
showSharing | boolean | If set to `true`, and the dashboard is shareable, then the dashboard shows the Share icon. Users can click the icon to open the Share dialog and post or download from the dashboard. If set to `false`, the dashboard doesn’t show the Share icon. This field is available in API version 37.0 and later.

showTitle | boolean | If `true`, includes the dashboard’s title above the dashboard. If `false`, the dashboard appears without a title. The default is `true`.

width | string | Specifies the width of the dashboard, in pixels or percent. Pixel values are simply the number of pixels, for example, 500. Percentage values must include the percent sign, for example, 20%. The default is `100%`.

---

### ReportChartComponentLayoutItem

Represents the settings for a report chart on a standard or custom page.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cacheData</td>
<td>boolean</td>
<td>Indicates whether to use cached data when displaying the chart. When the attribute is set to <code>true</code>, data is cached for 24 hours. If the attribute is set to <code>false</code>, the report is run every time the page is refreshed. This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>contextFilterableField</td>
<td>string</td>
<td>Unique development name of the field by which a report chart is filtered to return data relevant to the page. If set, the ID field for the parent object of the page or report type is the chart data filter. The parent object for the report type and the page must match for a chart to return relevant data.</td>
</tr>
<tr>
<td>error</td>
<td>string</td>
<td>Error string; only populated if an error occurred in the underlying report. This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>hideOnError</td>
<td>boolean</td>
<td>Controls whether users see a chart that has an error. When there’s an error and this attribute is not set, the chart doesn’t show any data except the error. An error can happen for many reasons, such as when a user doesn’t have access to fields used by the chart or a chart has been removed from the report. Set the attribute to <code>true</code> to hide the chart from a page on error. This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>includeContext</td>
<td>boolean</td>
<td>If <code>true</code>, filters the report chart to return data that’s relevant to the page.</td>
</tr>
<tr>
<td>reportName</td>
<td>string</td>
<td>Unique development name of a report that includes a chart.</td>
</tr>
</tbody>
</table>
### PlatformActionList

PlatformActionList represents the list of actions, and their order, that display in the Salesforce app action bar for the layout. Available in API version 34.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionListContext</td>
<td>PlatformActionListContext</td>
<td>Required. The context of the action list. Valid values:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• ActionDefinition—Reserved for future use.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Assistant</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BannerPhoto</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Chatter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Dockable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedElement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Flexipage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Global</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ListView</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ListViewDefinition</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ListViewRecord</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Lookup</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MruList</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MruRow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ObjectHomeChart</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Photo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Record</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RecordEdit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RelatedList</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RelatedListRecord</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>platformActionListItems</td>
<td>PlatformActionListItem[]</td>
<td>The actions in the PlatformActionList.</td>
</tr>
<tr>
<td>relatedSourceEntity</td>
<td>string</td>
<td>When the ActionListContext is RelatedList or RelatedListRecord, this field represents the API name of the related list to which the action belongs.</td>
</tr>
</tbody>
</table>
PlatformActionListItem

PlatformActionListItem represents an action in the PlatformActionList. Available in API version 34.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>string</td>
<td>The API name for the action in the list.</td>
</tr>
</tbody>
</table>
| actionType     | PlatformActionType (enumeration of type string) | The type of action. Valid values are:  
  - ActionLink—An indicator on a feed element that targets an API, a web page, or a file, represented by a button in the Salesforce Chatter feed UI.  
  - CustomButton—When clicked, opens a URL or a Visualforce page in a window or executes JavaScript.  
  - InvocableAction  
  - ProductivityAction—Productivity actions are predefined and attached to a limited set of objects. Productivity actions include Send Email, Call, Map, View Website, and Read News. Except for the Call action, you can't edit or delete productivity actions.  
  - QuickAction—A global or object-specific action.  
  - StandardButton—A predefined Salesforce button such as New, Edit, and Delete. |
| sortOrder      | int        | The placement of the action in the list.                                    |
| subtype        | string     | The subtype of the action. For quick actions, the subtype is QuickActionType. For custom buttons, the subtype is WebLinkTypeEnum. For action links, subtypes are Api, ApiAsync, Download, and Ui. Standard buttons and productivity actions have no subtype. |

QuickActionList

QuickActionList represents the list of actions associated with the page layout. Available in API version 28.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quickActionListItems</td>
<td>QuickActionListItem[]</td>
<td>Array of zero or more QuickActionList objects.</td>
</tr>
</tbody>
</table>

QuickActionListItem

QuickActionListItem represents an action in the QuickActionList. Available in API version 28.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quickActionName</td>
<td>string</td>
<td>The API name of the action.</td>
</tr>
</tbody>
</table>
RelatedContent

RelatedContent represents the Mobile Cards section of the page layout. Available in API version 29.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relatedContentItems</td>
<td>RelatedContentItem</td>
<td>A list of layout items in the Mobile Cards section of the page layout.</td>
</tr>
</tbody>
</table>

RelatedContentItem

RelatedContentItem represents an individual item in the RelatedContentItem list. Available in API version 29.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layoutItem</td>
<td>LayoutItem</td>
<td>An individual LayoutItem in the Mobile Cards section.</td>
</tr>
</tbody>
</table>

RelatedListItem

RelatedListItem represents a related list in a page layout.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customButtons</td>
<td>string[]</td>
<td>A list of custom buttons used in the related list. For more information, see “Define Custom Buttons and Links” in the Salesforce Help.</td>
</tr>
<tr>
<td>excludeButtons</td>
<td>string[]</td>
<td>A list of excluded related-list buttons.</td>
</tr>
<tr>
<td>fields</td>
<td>string[]</td>
<td>A list of fields displayed in the related list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Retrieval of standard fields on related lists uses aliases instead of field or API names. For example, the Fax, Mobile, and Home Phone fields are retrieved as Phone2, Phone3, and Phone4, respectively.</td>
</tr>
<tr>
<td>relatedList</td>
<td>string</td>
<td>Required. The name of the related list.</td>
</tr>
<tr>
<td>sortField</td>
<td>string</td>
<td>The name of the field that is used for sorting.</td>
</tr>
<tr>
<td>sortOrder</td>
<td>SortOrder</td>
<td>If the sortField is set, the sortOrder field determines the sort order.</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• Asc - sort in ascending order</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Desc - sort in descending order</td>
</tr>
</tbody>
</table>

SummaryLayout

Controls the appearance of the highlights panel in Salesforce Classic, which summarizes key fields in a grid at the top of a page layout, when Case Feed is enabled. Available in API version 25.0 and later.
### Field Name | Field Type | Description
--- | --- | ---
masterLabel | string | Required. The name of the layout label.

**sizeX** | int | Required. Number of columns in the highlights pane, from 1 through 4 (inclusive).

**sizeY** | int | Required. Number of rows in each column, either 1 or 2.

**sizeZ** | int | Reserved for future use. If provided, the setting is not visible to users.

**summaryLayoutItems** | SummaryLayoutItem[] | Controls the appearance of an individual field and its column and row position within the highlights panel grid, when Case Feed is enabled. At least one is required.

**summaryLayoutStyle** | SummaryLayoutStyle (enumeration of type string) | Highlights panel style. Valid string values are:
- Default
- QuoteTemplate
- DefaultQuoteTemplate
- CaseInteraction
- QuickActionLayoutLeftRight (Available in API version 28.0 and later.)
- QuickActionLayoutTopDown (Available in API version 28.0 and later.)

---

**SummaryLayoutItem**

Controls the appearance of an individual field and its column and row position within the highlights panel grid, when Case Feed is enabled. You can have two fields per each grid in a highlights panel. Available in API version 25.0 and later.

### Field Name | Field Type | Description
--- | --- | ---
**customLink** | string | If the item is a custom link, this is the customLink reference.

**field** | string | The field name reference, relative to the page layout. Must be a standard or custom field that also exists on the detail page.

**posX** | int | Required. The item’s column position in the highlights panel grid. Must be within the range of sizeX.

**posY** | int | Required. The item’s row position in the highlights panel grid. Must be within the range of sizeY.

**posZ** | int | Reserved for future use. If provided, the setting is not visible to users.

---

### Declarative Metadata Sample Definition

The following is the definition of a page layout:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Layout xmlns="http://soap.sforce.com/2006/04/metadata">
  <customConsoleComponents>
    ...  
  </customConsoleComponents>
</Layout>
```
Metadata Types

Layout

<primaryTabComponents>
<container>
<region>left</region>
<style>Stack</style>
<unit>Pixel</unit>
<width>101</width>
<sidebarComponent>
<width>60</width>
<page>simplepage1</page>
<unit>Percentage</unit>
</sidebarComponent>
<sidebarComponent>
<width>40</width>
<page>Hello_World</page>
<unit>Percentage</unit>
</sidebarComponent>
</container>
</primaryTabComponents>
<subtabComponents>
<component>
<location>top</location>
<visualforcePage>ConsoleComponentPage2</visualforcePage>
<height>200</height>
</component>
</subtabComponents>
</customConsoleComponents>
<customButtons>ButtonLink</customButtons>
<layoutSections>
<editHeading>true</editHeading>
<label>Information</label>
<layoutColumns>
<layoutItems>
<behavior>Required</behavior>
<field>Name</field>
</layoutItems>
<layoutItems>
<height>180</height>
<scontrol>LayoutSControl</scontrol>
<showLabel>true</showLabel>
<showScrollbars>true</showScrollbars>
<width>50%</width>
</layoutItems>
<layoutItems>
<reportChartComponent>
<contextFilterableField>CUST_ID</contextFilterableField>
<includeContext>true</includeContext>
<reportName>Open_Accounts_by_Cases</reportName>
<showTitle>false</showTitle>
<size>LARGE</size>
<reportChartComponent>
</layoutItems>
</layoutColumns>
<layoutColumns>
<layoutItems>

563


<behavior>Edit</behavior>
<field>OwnerId</field>
</layoutItems>
<layoutItems>
<behavior>Edit</behavior>
<field>CurrencyIsoCode</field>
</layoutItems>
</layoutColumns>
<style>TwoColumnsTopToBottom</style>
</layoutSections>
<layoutSections>
<editHeading>true</editHeading>
<label>System Information</label>
<layoutColumns>
<layoutItems>
<behavior>Readonly</behavior>
<field>CreatedById</field>
</layoutItems>
<layoutItems>
<behavior>Readonly</behavior>
<field>Alpha1__c</field>
</layoutItems>
<layoutItems>
<height>200</height>
<page>mcanvasPage</page>
<showLabel>true</showLabel>
<showScrollbars>false</showScrollbars>
<width>100%</width>
</layoutItems>
</layoutColumns>
<layoutColumns>
<layoutItems>
<behavior>Readonly</behavior>
<field>LastModifiedById</field>
</layoutItems>
<layoutItems>
<behavior>Edit</behavior>
<field>TextArea__c</field>
</layoutItems>
</layoutColumns>
<style>TwoColumnsTopToBottom</style>
</layoutSections>
<layoutSections>
<customLabel>true</customLabel>
<detailHeading>true</detailHeading>
<label>Custom Links</label>
<layoutColumns>
<layoutItems>
<customLink>CustomWebLink</customLink>
</layoutItems>
</layoutColumns>
<style>CustomLinks</style>
</layoutSections>
<quickActionList>
The following is an example of a layout using `<summaryLayout>`:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Layout xmlns="http://soap.sforce.com/2006/04/metadata">
    <layoutSections>
        <editHeading>true</editHeading>
        <label>System Information</label>
    </layoutSections>
    <layoutColumns>
        <layoutItems>
            <behavior>ReadOnly</behavior>
            <field>CreatedById</field>
        </layoutItems>
    </layoutColumns>
</Layout>
```
The following is an example of a feed-based layout:

```xml
<feedLayout>
  <leftComponents>
    <componentType>customLinks</componentType>
  </leftComponents>
  <rightComponents>
    <componentType>follow</componentType>
  </rightComponents>
  <rightComponents>
    <componentType>followers</componentType>
  </rightComponents>
  <rightComponents>
    <componentType>visualforce</componentType>
    <page>accountCustomWidget</page>
    <height>200</height>
  </rightComponents>
  <hideSidebar>true</hideSidebar>
  <feedFilterPosition>centerDropDown</feedFilterPosition>
</feedLayout>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Letterhead

Represents formatting options for the letterhead in an email template. A letterhead defines the logo, page color, and text settings for your HTML email templates. Use letterheads to ensure a consistent look and feel in your company’s emails.

For more information, see “Create Classic Letterheads for Email Templates” in the Salesforce online help. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

The file suffix for letterheads is .letter and components are stored in the letterhead directory of the corresponding package directory.

Version

Letterheads are available in API version 12.0 and later.

Fields

With the exception of logo, and horizontal and vertical alignment, all of these fields are required.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>available</td>
<td>boolean</td>
<td>Required. Indicates whether this letterhead can be used (true) or not (false), for example, in an email template.</td>
</tr>
<tr>
<td>backgroundColor</td>
<td>string</td>
<td>Required. The background color, in hexadecimal, for example #FF660.</td>
</tr>
<tr>
<td>bodyColor</td>
<td>string</td>
<td>Required. The body color in hexadecimal.</td>
</tr>
</tbody>
</table>
| bottomLine  | LetterheadLine (enumeration of type string) | Required. The style for the bottom line. Valid style values include:  
  • color. The color of the line in hexadecimal, as a string value.  
  • height. The height of the line, as an int value. |
### Metadata Types

**Letterhead**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Text description of how this letterhead differs from other letterheads.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The internal name of the letterhead, based on the name, but with white spaces and special characters escaped out for validity.</td>
</tr>
<tr>
<td>footer</td>
<td>LetterheadHeaderFooter</td>
<td>Required. The style for the footer.</td>
</tr>
<tr>
<td>header</td>
<td>LetterheadHeaderFooter</td>
<td>Required. The style for the header.</td>
</tr>
<tr>
<td>middleLine</td>
<td>LetterheadLine</td>
<td>Required. The style for the middle border line in your letterhead. Valid style values include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• color. The color of the line in hexadecimal, as a string value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• height. The height of the line, as an int value.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the letterhead.</td>
</tr>
<tr>
<td>topLine</td>
<td>LetterheadLine</td>
<td>Required. The style for the top horizontal line below the header. Valid style values include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• color. The color of the line in hexadecimal, as a string value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• height. The height of the line, as an int value.</td>
</tr>
</tbody>
</table>

**LetterheadHeaderFooter**

LetterheadHeaderFooter represents the properties of a header or footer.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>backgroundColor</td>
<td>string</td>
<td>Required. The background color of the header or footer in hexadecimal format.</td>
</tr>
<tr>
<td>height</td>
<td>DashboardComponent[]</td>
<td>Required. The height of the header or footer.</td>
</tr>
<tr>
<td>horizontalAlignment</td>
<td>LetterheadHorizontalAlignment (enumeration of type string)</td>
<td>The horizontal alignment of the header or footer. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Left</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Center</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Right</td>
</tr>
<tr>
<td>logo</td>
<td>string</td>
<td>The logo which is a reference to a document, for example MyFolder/MyDocument.gif.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>verticalAlignment</td>
<td>LetterheadVerticalAlignment (string)</td>
<td>The vertical alignment of the header or footer. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Top</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Middle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Bottom</td>
</tr>
</tbody>
</table>

**LetterheadLine**

LetterheadLine represents the properties of a line.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>string</td>
<td>Required. The color of the line in hexadecimal format.</td>
</tr>
<tr>
<td>height</td>
<td>int</td>
<td>Required. The height of the line.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Letterhead xmlns="http://soap.sforce.com/2006/04/metadata">
  <available>true</available>
  <backgroundColor>#CCCCCC</backgroundColor>
  <bodyColor>#33FF33</bodyColor>
  <bottomLine>
    <color>#3333FF</color>
    <height>5</height>
  </bottomLine>
  <description>INITIAL</description>
  <footer>
    <backgroundColor>#FFFFFF</backgroundColor>
    <height>100</height>
    <horizontalAlignment>Left</horizontalAlignment>
    <verticalAlignment>Top</verticalAlignment>
  </footer>
  <header>
    <backgroundColor>#FFFFFF</backgroundColor>
    <height>100</height>
    <horizontalAlignment>Left</horizontalAlignment>
    <verticalAlignment>Top</verticalAlignment>
  </header>
  <middleLine>
    <color>#AAAAFF</color>
    <height>5</height>
  </middleLine>
  <name>SimpleLetterheadLabel</name>
  <topLine>
    <color>#FF99FF</color>
  </topLine>
</Letterhead>
```
Metadata Types

**Wildcard Support in the Manifest File**

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**LightningBolt**

Represents the definition of a Lightning Bolt Solution, which can include custom apps, flow categories, and Lightning Community templates. This type extends the Metadata metadata type and inherits its fullName field.

**File Suffix and Directory Location**

LightningBolt components have the suffix .lightningBolt and are stored in the lightningBolts folder.

**Version**

LightningBolt components are available in API version 43.0 and later.

**Special Access Rules**

To add Lightning Community templates to a Lightning Bolt Solution, enable Salesforce Communities in your org.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>category</td>
<td>LightningBoltCategory (enumeration of type string)</td>
<td>Required. The primary industry that the Lightning Bolt Solution is aimed at. Valid values are: • Communications • Education • FinancialServices • GeneralBusiness • Government • HealthcareLifeSciences • HighTech • Manufacturing • Media • Nonprofits • ProfessionalServices</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The label of the Lightning Bolt Solution, which appears on the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>solution detail page.</td>
</tr>
<tr>
<td>publisher</td>
<td>string</td>
<td>Required. The name of the partner organization associated with this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lightning Bolt Solution.</td>
</tr>
<tr>
<td>summary</td>
<td>string</td>
<td>Required. The summary description of the Lightning Bolt Solution.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lightningBoltFeatures</td>
<td>LightningBoltFeatures[]</td>
<td>The list of feature descriptions of this Lightning Bolt Solution.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lightningBoltImages</td>
<td>LightningBoltImages[]</td>
<td>The list of images of this Lightning Bolt Solution.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lightningBoltItems</td>
<td>LightningBoltItems[]</td>
<td>The list of items (custom apps, flow categories, and Lightning Community</td>
</tr>
<tr>
<td></td>
<td></td>
<td>templates) that comprise this Lightning Bolt Solution.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**LightningBoltFeatures**

Represents the list of feature descriptions of a Lightning Bolt Solution.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the feature of the Lightning Bolt Solution.</td>
</tr>
<tr>
<td>order</td>
<td>int</td>
<td>Required. An integer specifying the position of this feature relative to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>others in the list. 1 is the first position, and 4 is the max position.</td>
</tr>
<tr>
<td>title</td>
<td>string</td>
<td>Required. The title of the feature, which appears on the solution detail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page.</td>
</tr>
</tbody>
</table>

**LightningBoltImages**

Represents the list of images of a Lightning Bolt Solution.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>image</td>
<td>string</td>
<td>Required. The developer name of the ContentAsset type, which is used as a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>preview image for this Lightning Bolt Solution.</td>
</tr>
<tr>
<td>order</td>
<td>int</td>
<td>Required. An integer specifying the position of this image relative to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>others in the list. 1 is the first position, and 3 is the max position.</td>
</tr>
</tbody>
</table>

**LightningBoltItems**

Represents the list of items (custom apps, flow categories, and Lightning Community templates) that comprise a Lightning Bolt Solution.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the item, which appears on the solution detail page.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Required. The type of the item included in the Lightning Bolt Solution. Valid values are: CommunityTemplateDefinition, CustomApplication, FlowCategory</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example of a LightningBolt component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LightningBolt xmlns="http://soap.sforce.com/2006/04/metadata"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <category>Sales</category>
  <lightningBoltFeatures>
    <description>bb</description>
    <order>1</order>
    <title>aa</title>
  </lightningBoltFeatures>
  <lightningBoltImages>
    <image>prm1</image>
    <order>1</order>
  </lightningBoltImages>
  <lightningBoltItems>
    <name>PolaConsole</name>
    <type>CustomApplication</type>
  </lightningBoltItems>
  <lightningBoltItems>
    <name>Banking_Service_Console</name>
    <type>CustomApplication</type>
  </lightningBoltItems>
  <lightningBoltItems>
    <name>Banking_Service_Portal</name>
    <type>CommunityTemplateDefinition</type>
  </lightningBoltItems>
  <lightningBoltItems>
    <name>Banking_Sales_Portal</name>
    <type>CommunityTemplateDefinition</type>
  </lightningBoltItems>
  <lightningBoltItems>
    <name>myorgdev__updatebenefits</name>
    <type>FlowCategory</type>
  </lightningBoltItems>
  <masterLabel>BoltTe</masterLabel>
  <publisher>aaaa</publisher>
  <summary>This is a summary.</summary>
</LightningBolt>
```
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>BoltTe</members>
        <name>LightningBolt</name>
    </types>
    <version>43.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**LightningComponentBundle**

Represents a Lightning web component bundle. A bundle contains Lightning web component resources.

**Special Access Rules**

LightningComponentBundle components can be created only in organizations with defined namespaces.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>double</td>
<td>A double value that binds the component to a Salesforce API version.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the Lightning web component.</td>
</tr>
<tr>
<td>isExplicitImport</td>
<td>boolean</td>
<td>Indicates whether imports between files are done explicitly by the developer (true) or implicitly by the framework (false).</td>
</tr>
<tr>
<td>isExposed</td>
<td>boolean</td>
<td>Indicates whether a component is usable in a managed package (true) or not (false).</td>
</tr>
<tr>
<td>lwcResources</td>
<td>LwcResources[]</td>
<td>A list of resources inside a bundle.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>The component title that appears in the list view.</td>
</tr>
<tr>
<td>targetConfigs</td>
<td>base64Binary</td>
<td>Configurations for each target. Each target is a Lightning page type. For example, this configuration allows a Lightning web component to be used on a Contact record page in Lightning App Builder.</td>
</tr>
</tbody>
</table>

```xml
<targetConfigs>
    <targetConfig targets="lightning__RecordPage">
        <objects>
            <object>Contact</object>
        </objects>
    </targetConfig>
</targetConfigs>
```
A list of targets where the Lightning web component is supported. Each target is a Lightning page type that can be configured in Lightning App Builder.

LwcResources

Represents a list of resources inside a LightningComponentBundle.

LwcResource

Represents a resource inside a LightningComponentBundle.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filePath</td>
<td>string</td>
<td>Required. The file path of a resource.</td>
</tr>
<tr>
<td>source</td>
<td>base64Binary</td>
<td>Required. The content of a resource.</td>
</tr>
</tbody>
</table>

Targets

Represents a list of supported containers for a Lightning web component.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| target | string | Specifies the type of Lightning page the component can be added to in Lightning App Builder. Valid values are:  
  - lightning__AppPage—Enables a component to be used on a Lightning app page.  
  - lightning__HomePage—Enables a component to be used on a custom Lightning Home page.  
  - lightning__RecordPage—Enables a component to be used on a Lightning record page, such as Account or Contact. |
Declarative Metadata Sample Definition

This package.xml file retrieves all the LightningComponentBundle components in an org.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>*</members>
        <name>LightningComponentBundle</name>
    </types>
    <version>45.0</version>
</Package>
```

In the retrieved zip file, each Lightning web component is nested under an lwc folder.

This example shows the directory structure in the zip file of one component with a name of hello.

```
lwc
    hello
    hello.html
    hello.js
    hello.js-meta.xml
```

Here are the contents of the files in the hello directory.

Content of hello.html:

```html
<template>
    <lightning-card title="Hello" icon-name="custom:custom14">
        <div class="slds-m-around_medium">
            Hello, {greeting}!
        </div>
    </lightning-card>
</template>
```

Content of hello.js:

```javascript
import { LightningElement, track } from 'lwc';

export default class Hello extends LightningElement {
    @track greeting = 'World';
}
```

Content of hello.js-meta.xml.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>45.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__AppPage</target>
        <target>lightning__RecordPage</target>
        <target>lightning__HomePage</target>
    </targets>
</LightningComponentBundle>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

LightningExperienceTheme

Represents the details of a custom theme, including the BrandingSet. Themes enable admins to specify configurable attributes, such as three colors and five images. The colors and some of the images override SLDS token values and influence the generation of app.css.

File Suffix and Directory Location

LightningExperienceTheme components have the suffix .lightningExperienceTheme and are stored in the lightningExperienceThemes folder.

Version

LightningExperienceTheme components are available in API version 42.0 and later.

Special Access Rules

The LightningExperienceTheme type is available when the S1DesktopAllowed permission is enabled in your org.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultBrandingSet</td>
<td>string</td>
<td>Required. The ID of the BrandingSet properties associated with this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LightningExperienceTheme.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The optional description text of this LightningExperienceTheme. Limited</td>
</tr>
<tr>
<td></td>
<td></td>
<td>to 1000 characters.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The label for this LightningExperienceTheme, which displays</td>
</tr>
<tr>
<td></td>
<td></td>
<td>in Setup. Limited to 70 characters.</td>
</tr>
<tr>
<td>shouldOverrideLoadingImage</td>
<td>boolean</td>
<td>If true, the LightningExperienceTheme overrides the splash screen image.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a LightningExperienceTheme component. See BrandingSet on page 203 for an example of the BrandingSet component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LightningExperienceTheme xmlns="http://soap.sforce.com/2006/04/metadata">
    <defaultBrandingSet>SummerCelebrationBrand</defaultBrandingSet>
    <description>Theme for summer celebration week.</description>
</LightningExperienceTheme>
```
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>LEXTHEMINGThemeName</members>
    <name>BrandingSet</name>
  </types>
  <types>
    <members>Summer Celebration</members>
    <name>LightningExperienceTheme</name>
  </types>
  <version>42.0</version>
</Package>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### LiveChatAgentConfig

Represents the configuration of an organization’s Chat deployment, such as how many chats can be assigned to an agent and whether or not chat sounds are enabled. This type extends the Metadata metadata type and inherits its fullName field.

### File Suffix and Directory Location

LiveChatAgentConfig configurations are referenced in the `<developer_name>.liveChatAgentConfig` file in the `liveChatAgentConfigs` directory.

### Version

LiveChatAgentConfig is available in API version 28.0 and later.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignments</td>
<td>AgentConfigAssignments</td>
<td>Specifies how agent configurations are assigned to Chat users. Agent configurations can be assigned to sets of users or sets of profiles.</td>
</tr>
<tr>
<td>autoGreeting</td>
<td>string</td>
<td>Specifies the greeting that displays when a customer begins a chat with an agent.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>capacity</td>
<td>int</td>
<td>Specifies the maximum number of chats in which an agent can be engaged at a time.</td>
</tr>
<tr>
<td>criticalWaitTime</td>
<td>int</td>
<td>Specifies the number of seconds an agent can wait to answer an engaged chat before the chat tab flashes to alert the agent to answer it.</td>
</tr>
<tr>
<td>customAgentName</td>
<td>string</td>
<td>Specifies the custom name for an agent, if one has been set. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>enableAgentFileTransfer</td>
<td>boolean</td>
<td>Indicates whether file transfer is enabled for agents (true) or not (false). Available in API version 31.0 and later.</td>
</tr>
<tr>
<td>enableAgentSneakPeek</td>
<td>boolean</td>
<td>Specifies whether a supervisor can see the content of an agent’s message before they send it to a customer (true) or not (false).</td>
</tr>
<tr>
<td>enableAssistanceFlag</td>
<td>boolean</td>
<td>Indicates whether agents can raise an assistance flag to notify a supervisor that they need help. Available in API version 35.0 and later.</td>
</tr>
<tr>
<td>enableAutoAwayOnDecline</td>
<td>boolean</td>
<td>Indicates whether an agent appears as “away” (true) or not (false) when an agent declines a chat with a customer.</td>
</tr>
<tr>
<td>enableAutoAwayOnPushTimeout</td>
<td>boolean</td>
<td>Indicates whether an agent appears as “away” (true) or not (false) when a chat request that’s been pushed to the agent times out. Available in API version 34.0 and later.</td>
</tr>
<tr>
<td>enableChatConferencing</td>
<td>boolean</td>
<td>Indicates whether chat conferencing is enabled for agents (true) or not (false). Available in API version 34.0 and later.</td>
</tr>
<tr>
<td>enableChatMonitoring</td>
<td>boolean</td>
<td>Indicates whether chat monitoring is enabled for support supervisors (true) or not (false). Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>enableChatTransferToAgent</td>
<td>boolean</td>
<td>Indicates whether agents can transfer a chat to another agent (true) or not (false). Available in API version 36.0 and later.</td>
</tr>
<tr>
<td>enableChatTransferToButton</td>
<td>boolean</td>
<td>Indicates whether agents can transfer a chat to a button (true) or not (false). Available in API version 36.0 and later.</td>
</tr>
<tr>
<td>enableChatTransferToSkill</td>
<td>boolean</td>
<td>Indicates whether agents can transfer a chat to a skill group (true) or not (false). Available in API version 36.0 and later.</td>
</tr>
<tr>
<td>enableLogoutSound</td>
<td>boolean</td>
<td>Indicates whether a sound will play (true) or not (false) when an agent logs out of Chat.</td>
</tr>
<tr>
<td>enableNotifications</td>
<td>boolean</td>
<td>Indicates whether notifications of incoming chats appear for agents (true) or not (false).</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>enableRequestSound</td>
<td>boolean</td>
<td>Indicates whether a sound will play (true) or not (false) when a customer requests to chat with an agent.</td>
</tr>
<tr>
<td>enableSneakPeek</td>
<td>boolean</td>
<td>Indicates whether previews of customers’ messages are displayed as customers type (true) or not (false) in the agent’s Chat window. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>enableVisitorBlocking</td>
<td>boolean</td>
<td>Indicates whether an agent can block a visitor by IP address (true) or not (false). Available in API version 34.0 and later.</td>
</tr>
<tr>
<td>enableWhisperMessage</td>
<td>boolean</td>
<td>Indicates whether support supervisors can send whisper messages to agents during a chat (true) or not (false). Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Specifies the name of the configuration for agents’ default chat settings.</td>
</tr>
<tr>
<td>supervisorDefaultAgentStatusFilter</td>
<td>SupervisorAgentStatusFilter (enumeration of type string)</td>
<td>Specifies the Chat status for filtering the Agent Status list in the Supervisor Panel. Valid values are: Online, Away, Offline. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>supervisorDefaultButtonFilter</td>
<td>string</td>
<td>Specifies the default button for filtering the Agent Status list in the Supervisor Panel. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>supervisorDefaultSkillFilter</td>
<td>string</td>
<td>Specifies the default skill for filtering the Agent Status list in the Supervisor Panel. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>supervisorSkills</td>
<td>SupervisorAgentConfigSkills</td>
<td>Specifies the list of agent skills that are assigned to a supervisor, as specified in their assigned Chat configuration. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>transferableButtons</td>
<td>AgentConfigButtons</td>
<td>Specifies the list of chat buttons that agents can transfer chats to. Available in API version 31.0 and later.</td>
</tr>
<tr>
<td>transferableSkills</td>
<td>AgentConfigSkills</td>
<td>Specifies the list of skill groups that agents can transfer chats to. Available in API version 31.0 and later.</td>
</tr>
</tbody>
</table>

**AgentConfigAssignments**

Represents the assignments of an organization’s profiles and users to a Chat configuration.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>profiles</td>
<td>AgentConfigProfileAssignments</td>
<td>Specifies the profiles that are associated with a specific agent configuration.</td>
</tr>
<tr>
<td>users</td>
<td>AgentConfigUserAssignments</td>
<td>Specifies the users that are associated with a specific agent configuration.</td>
</tr>
</tbody>
</table>

### AgentConfigButtons

Represents the chat buttons that agents who are associated with the Chat configuration can transfer chats to.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>button</td>
<td>string[]</td>
<td>Specifies the chat buttons that agents can transfer chats to.</td>
</tr>
</tbody>
</table>

### AgentConfigProfileAssignments

Represents the profiles associated with a specific Chat configuration.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>profile</td>
<td>string</td>
<td>Specifies the custom name of the profile associated with a specific agent configuration.</td>
</tr>
</tbody>
</table>

### AgentConfigSkills

Represents the skill groups that agents who are associated with the Chat configuration can transfer chats to.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>skill</td>
<td>string[]</td>
<td>Specifies the skill groups that agents can transfer chats to.</td>
</tr>
</tbody>
</table>

### AgentConfigUserAssignments

Represents the users associated with a specific Chat configuration.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>string</td>
<td>Specifies the username of the user associated with a specific agent configuration.</td>
</tr>
</tbody>
</table>

### SupervisorAgentConfigSkills

Represents the agent skills associated with a supervisor's Chat configuration. Available in API version 29.0 and later.
### Declarative Metadata Sample Definition

This is a sample of a `liveChatAgentConfig` file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LiveChatAgentConfig xmlns="http://soap.sforce.com/2006/04/metadata">
  <label>My Agent Configuration 1</label>
  <autoGreeting>Hi, how can I help you?</autoGreeting>
  <capacity>5</capacity>
  <enableAutoAwayOnDecline>true</enableAutoAwayOnDecline>
  <enableLogoutSound>true</enableLogoutSound>
  <enableNotifications>true</enableNotifications>
  <enableRequestSound>true</enableRequestSound>
  <enableSneakPeek>true</enableSneakPeek>
  <enableWhisperMessage>true</enableWhisperMessage>
  <assignments>
    <profiles>
      <profile>standard</profile>
    </profiles>
    <users>
      <user>jdoe@acme.com</user>
    </users>
  </assignments>
</LiveChatAgentConfig>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### LiveChatButton

Represents a Chat deployment’s settings for the button that customers click to chat with an agent and the chat window, such as the label that appears on the button and the pre-chat form that appears before a chat begins. This type extends the Metadata metadata type and inherits its `fullName` field.

**Note:** Chats routed with Omni-Channel aren’t supported in the Metadata API.

### File Suffix and Directory Location

LiveChatButton configurations are stored in the `<developer_name>.liveChatButton` file in the `liveChatButtons` directory.
## Version

LiveChatButton is available in API version 28.0 and later.

## Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>animation</td>
<td>LiveChatButtonPresentation (enumeration of type string)</td>
<td>The type of animation for a chat invitation. Valid values are: Slide, Fade, Appear, Custom.</td>
</tr>
<tr>
<td>autoGreeting</td>
<td>string</td>
<td>The customized greeting message that the customer receives when an agent accepts a chat request from the chat button or invitation. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>chasitorIdleTimeout</td>
<td>int</td>
<td>Specifies the amount of idle time before the chat times out. The idle time starts being counted after the agent sends the last chat message. Available in API version 35.0 and later.</td>
</tr>
<tr>
<td>chasitorIdleTimeoutWarning</td>
<td>int</td>
<td>Specifies the amount of idle time before a warning appears. The idle time starts being counted after the agent sends the last chat message. Available in API version 35.0 and later.</td>
</tr>
<tr>
<td>chatPage</td>
<td>string</td>
<td>Specifies the page that hosts your chat if that page differs from the Chat window.</td>
</tr>
<tr>
<td>customAgentName</td>
<td>string</td>
<td>The agent’s name as it appears to customers in the chat window. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>deployments</td>
<td>LiveChatButtonDeployments</td>
<td>Specifies the deployments associated with the button.</td>
</tr>
<tr>
<td>enableQueue</td>
<td>boolean</td>
<td>Indicates whether queuing is enabled (true) or not (false).</td>
</tr>
<tr>
<td>inviteEndPosition</td>
<td>LiveChatButtonInviteEndPosition (enumeration of type string)</td>
<td>The end position of the chat invitation. Valid values include: TopLeft, Top, TopRight.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>inviteImage</td>
<td>string</td>
<td>The custom button graphic that appears for the invitation.</td>
</tr>
<tr>
<td>inviteStartPosition</td>
<td>LiveChatButtonInviteStartPosition (enumeration of type string)</td>
<td>The start position of the chat invitation. Valid values include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TopLeft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TopLeftTop</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Top</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TopRightTop</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TopRight</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TopRightRight</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Right</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BottomRightRight</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BottomRight</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BottomRightBottom</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BottomLeft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BottomLeftBottom</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BottomLeftLeft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Left</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TopLeftLeft</td>
</tr>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Specifies whether the chat button or invitation is active.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Specifies the text that appears on the button.</td>
</tr>
<tr>
<td>numberOfReroutingAttempts</td>
<td>int</td>
<td>Specifies the number of times a chat request can be rerouted to available agents if all agents reject the chat request. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td>offlineImage</td>
<td>string</td>
<td>Specifies the image that appears on the button when no agents are available to chat.</td>
</tr>
<tr>
<td>onlineImage</td>
<td>string</td>
<td>Specifies the image that appears on the button when agents are available to chat.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>optionsCustomRoutingIsEnabled</td>
<td>boolean</td>
<td>Indicates whether custom routing is enabled for incoming chat requests (true) or not (false). Available in API version 30.0 and later.</td>
</tr>
<tr>
<td>optionsHasChasitorIdleTimeout</td>
<td>boolean</td>
<td>Indicates whether the visitor idle timeout feature is enabled. Available in API version 35.0 and later.</td>
</tr>
<tr>
<td>optionsHasInviteAfterAccept</td>
<td>boolean</td>
<td>Indicates whether a new chat invitation triggers after a customer accepts a previous chat invitation (true) or not (false).</td>
</tr>
<tr>
<td>optionsHasInviteAfterReject</td>
<td>boolean</td>
<td>Indicates whether a new chat invitation triggers after a customer rejects a previous chat invitation (true) or not (false).</td>
</tr>
<tr>
<td>optionsHasRerouteDeclinedRequest</td>
<td>boolean</td>
<td>Indicates whether a chat request, which has been rejected by all available agents, should be rerouted to available agents again (true) or not (false). Available in API version 30.0 and later.</td>
</tr>
<tr>
<td>optionsIsAutoAccept</td>
<td>boolean</td>
<td>Indicates whether a chat request should be automatically accepted by the agent it's assigned to (true) or not (false). For chat buttons and automated chat invitations with routingType set to MostAvailable or LeastActive. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td>optionsIsInviteAutoRemove</td>
<td>boolean</td>
<td>Indicates whether a chat invitation is set to automatically disappear from a customer's screen after a certain amount of time (true) or not (false).</td>
</tr>
<tr>
<td>overallQueueLength</td>
<td>int</td>
<td>Specifies the maximum number of chat requests that are allowed to queue.</td>
</tr>
<tr>
<td>perAgentQueueLength</td>
<td>int</td>
<td>Specifies the number of chat requests that are allowed to queue for an agent with the required skills.</td>
</tr>
<tr>
<td>postChatPage</td>
<td>string</td>
<td>Specifies the name of the post-chat form to which customers are routed when the chat ends.</td>
</tr>
<tr>
<td>postChatUrl</td>
<td>string</td>
<td>Specifies the URL of the post-chat form to which customers are routed when the chat ends.</td>
</tr>
<tr>
<td>preChatFormPage</td>
<td>string</td>
<td>Specifies the name of the pre-chat form to which customers are routed before a chat begins.</td>
</tr>
<tr>
<td>preChatFormUrl</td>
<td>string</td>
<td>Specifies the URL of the pre-chat form to which customers are routed when the chat begins.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>pushTimeOut</td>
<td>int</td>
<td>Specifies the number of seconds an agent has to answer an incoming chat request before the request is routed to another agent.</td>
</tr>
<tr>
<td>routingType</td>
<td>LiveChatButtonRoutingType</td>
<td>Specifies how incoming chats should be routed to agents when a customer pushes a button. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Choice</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LeastActive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MostAvailable</td>
</tr>
<tr>
<td>site</td>
<td>string</td>
<td>Specifies the Salesforce site that hosts your custom chat button images or custom chat page.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><img src="https://example.com" alt="Note" /> You must have the &quot;CustomDomain&quot; permission enabled in your organization before you can use a Salesforce site with Chat.</td>
</tr>
<tr>
<td>skills</td>
<td>LiveChatButtonSkills</td>
<td>Specifies the skills associated with the button. When a customer clicks the button to chat, they are automatically routed to agents with those skills.</td>
</tr>
<tr>
<td>timeToRemoveInvite</td>
<td>int</td>
<td>Specifies how long the invitation is displayed (in seconds) to customers before it disappears.</td>
</tr>
<tr>
<td>type</td>
<td>LiveChatButtonType</td>
<td>Required. The chat button type. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Standard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Invite</td>
</tr>
<tr>
<td>windowLanguage</td>
<td>Language</td>
<td>Specifies the language preferences for the chat window associated with the button.</td>
</tr>
</tbody>
</table>

**LiveChatButtonSkills**

Represents the skills associated with a chat button or invitation.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>skill</td>
<td>string</td>
<td>Specifies the name of the skill.</td>
</tr>
</tbody>
</table>
LiveChatButtonDeployments

Represents the deployments associated with a chat button or invitation.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>deployment</td>
<td>string</td>
<td>Specifies the name of the deployment.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

This is a sample of a liveChatButton file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LiveChatButton xmlns="http://soap.sforce.com/2006/04/metadata">
  <deployments/>
  <enableQueue>false</enableQueue>
  <isActive>true</isActive>
  <label>CustomerSupportButton</label>
  <optionsCustomRoutingIsEnabled>false</optionsCustomRoutingIsEnabled>
  <optionsHasChasitorIdleTimeout>false</optionsHasChasitorIdleTimeout>
  <optionsHasInviteAfterAccept>false</optionsHasInviteAfterAccept>
  <optionsHasInviteAfterReject>false</optionsHasInviteAfterReject>
  <optionsHasRerouteDeclinedRequest>false</optionsHasRerouteDeclinedRequest>
  <optionsIsAutoAccept>false</optionsIsAutoAccept>
  <optionsIsInviteAutoRemove>false</optionsIsInviteAutoRemove>
  <postChatUrl>http://help.salesforce.com</postChatUrl>
  <routingType>Choice</routingType>
  <skills>
    <skill>Chat</skill>
  </skills>
  <type>Standard</type>
</LiveChatButton>
```

Note: If you update your chat button through the Metadata API, be sure to update all Web pages that use the same chat button code.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

LiveChatDeployment

Represents the configuration settings for a specific Chat deployment, such as the branding image for the deployment and whether or not chat transcripts are automatically saved. This type extends the Metadata metadata type and inherits its fullName field.
File Suffix and Directory Location

LiveChatDeployment values are stored in the <developer_name>.liveChatDeployment file in the liveChatDeployments directory.

Version

LiveChatDeployment is available in API version 28.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>brandingImage</td>
<td>string</td>
<td>Specifies the branding image for the deployment.</td>
</tr>
<tr>
<td>connectionTimeoutDuration</td>
<td>int</td>
<td>Indicates the amount of time before the chat times out, in seconds.</td>
</tr>
<tr>
<td>ConnectionWarningDuration</td>
<td>int</td>
<td>Indicates the amount of time before a time-out warning is displayed to the agent, in seconds.</td>
</tr>
<tr>
<td>displayQueuePosition</td>
<td>boolean</td>
<td>(Pilot) Determines whether a customer’s queue position is displayed in a standard chat window while the customer waits for an agent to respond to the chat request (true) or not (false). This field is available as a pilot in API version 32.0. To enable this field, contact Salesforce.</td>
</tr>
<tr>
<td>domainWhiteList</td>
<td>LiveChatDeploymentDomainWhiteList</td>
<td>Specifies the list of domains that can host the deployment.</td>
</tr>
<tr>
<td>enablePrechatApi</td>
<td>boolean</td>
<td>Indicates whether or not the pre-chat API is enabled for the deployment (true) or not (false).</td>
</tr>
<tr>
<td>enableTranscriptSave</td>
<td>boolean</td>
<td>Indicates whether chat transcripts are automatically saved after a chat ends (true) or not (false).</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Specifies the name of the deployment.</td>
</tr>
<tr>
<td>mobileBrandingImage</td>
<td>string</td>
<td>Specifies the branding image for the deployment that appears when customers access the deployment on a mobile device.</td>
</tr>
<tr>
<td>site</td>
<td>string</td>
<td>Specifies the site that hosts the images for the deployment.</td>
</tr>
</tbody>
</table>

Note: You must have the "CustomDomain" permission enabled.
in your organization before you can use a Salesforce site with Chat.

windowTitle string Specifies the title of the window associated with the deployment.

---

**LiveChatDeploymentDomainWhiteList**

Represents a Chat deployment’s domain whitelist.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>string</td>
<td>Specifies a domain that can host the deployment.</td>
</tr>
</tbody>
</table>

---

**Declarative Metadata Sample Definition**

This is a sample of a liveChatDeployment file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LiveChatDeployment xmlns="http://soap.sforce.com/2006/04/metadata">
  <label>My Deployment 1</label>
  <brandingImage>pkb_image_bannerBg</brandingImage>
  <mobileBrandingImage>pkb_image_bgBottom</mobileBrandingImage>
  <domainWhiteList>
    <domain>mydomain</domain>
    <domain>test</domain>
  </domainWhiteList>
  <enableTranscriptSave>true</enableTranscriptSave>
  <site>GL_Knowledge_Base</site>
  <windowTitle>My window title</windowTitle>
</LiveChatDeployment>
```

**Note:** If you update your deployment through the Metadata API, be sure to update all Web pages that use the same deployment code.

---

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

---

**LiveChatSensitiveDataRule**

Represents a rule for masking or deleting data of a specified pattern. Written as a regular expression (regex).
Use this object to mask or delete data of specified patterns, such as credit card, social security, phone and account numbers, or even profanity. This type extends the Metadata metadata type and inherits its fullName field.

### File Suffix and Directory Location

LiveChatSensitiveDataRule components have the suffix `.liveChatSensitiveDataRule` and are stored in the `liveChatSensitiveDataRule` folder.

### Version

LiveChatSensitiveDataRule components are available in API version 35.0 and later.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `actionType` | SensitiveDataActionType (enumeration of type string) | Required. The action to take on the text when the sensitive data rule is triggered. Possible values are:
  - Remove
  - Replace |
| `description` | string | The description of the sensitive data rule—for example, “Block social security numbers.” |
| `enforceOn` | int | Required. Determines the roles on which the rule is enforced. The value is determined using bitwise OR operation. There are seven possible values:
  1. Rule enforced on Agent
  2. Rule enforced on Visitor
  3. Rule enforced on Agent and Visitor
  4. Rule enforced on Supervisor
  5. Rule enforced on Agent and Supervisor
  6. Rule enforced on Visitor and Supervisor
  7. Rule enforced on Agent, Visitor, and Supervisor |
| `isEnabled` | boolean | Required. Specifies whether a sensitive data rule is active (true) or not (false). Default value (if none is provided) is false. |
| `pattern` | string | Required. The pattern of text blocked by the rule. Written as a JavaScript regular expression (regex). |
| `replacement` | string | The string of characters that replaces the blocked text (if ActionType Replace is selected). |
**Declarative Metadata Sample Definition**

The following is an example of a LiveChatSensitiveDataRule component.

```xml
<LiveChatSensitiveDataRule xmlns="http://soap.sforce.com/2006/04/metadata">
  <actionType>REPLACE</actionType>
  <enforceOn>7</enforceOn>
  <isEnabled>true</isEnabled>
  <pattern>[aeiou]</pattern>
  <replacement>œ</replacement>
</LiveChatSensitiveDataRule>
```

The following is an example package.xml that references the previous definition.

```xml
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <!-- To be used from support.liveagent.testsuite.unifiedouting.testDeployButtonMDAPIWithExistingQueue -->
  <apiAccessLevel>Unrestricted</apiAccessLevel>
  <types>
    <members>Change_For_all</members>
    <name>LiveChatSensitiveDataRule</name>
  </types>
  <version>35.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**ManagedTopics**

Represents navigational and featured topics managed in a community. A specific community is represented by the Network component.

**Note:** The related network must exist before you deploy managed topics. (This occurs automatically when deploying an entire organization.)

**File Suffix and Directory Location**

Components have the suffix managedTopics and are stored in the managedTopics folder. In that folder, you'll find separate files for each network (for example, NetworkNameA.managedTopics and NetworkNameB.managedTopics).

**Version**

ManagedTopics components are available in API version 32.0 and later.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedTopic</td>
<td>ManagedTopic</td>
<td>Represents a specific navigational or featured topic.</td>
</tr>
</tbody>
</table>

ManagedTopic

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The topic name.</td>
</tr>
<tr>
<td>managedTopicType</td>
<td>string</td>
<td>The topic type: “Navigational” or “Featured”</td>
</tr>
<tr>
<td>topicDescription</td>
<td>string</td>
<td>An optional description of topic contents. This field is accessible only via the API; there is no corollary in the user interface.</td>
</tr>
<tr>
<td>parentName</td>
<td>string</td>
<td>The name of a parent topic for which this topic is a child. Child topics are accessible from the subtopics section of the parent topic page and their feeds are added to the parent topic feed. Only navigational topics support parent-child relationships.</td>
</tr>
<tr>
<td>position</td>
<td>int</td>
<td>The placement of this topic relative to others of the same type. The results differ depending on topic type:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For top-level navigational topics, position arranges the Topics menu in the community.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For child navigational topics, it arranges sibling topics in the subtopics section.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For featured topics, it arranges topic thumbnail images on the community home page.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enter a number between 0 and 24. (The maximum amount of navigational or featured topics is 25.)</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following example retrieves or deploys managed topics for all networks:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>ManagedTopics</name>
  </types>
  <version>32.0</version>
</Package>
```
The following example shows a package.xml file referencing the ManagedTopics component:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>NetworkName</members>
    <name>ManagedTopics</name>
  </types>
  <version>32.0</version>
</Package>
```

The following example shows the ManagedTopics component itself:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ManagedTopics>
  <ManagedTopic>
    <name>Running</name>
    <managedTopicType>Navigational</managedTopicType>
    <topicDescription>Training advice</topicDescription>
    <parentName></parentName>
    <position>0</position>
  </ManagedTopic>
  <ManagedTopic>
    <name>Hiking</name>
    <managedTopicType>Navigational</managedTopicType>
    <topicDescription>Routes and gear</topicDescription>
    <parentName></parentName>
    <position>1</position>
  </ManagedTopic>
  <ManagedTopic>
    <name>Trails</name>
    <managedTopicType>Navigational</managedTopicType>
    <topicDescription>Maps for local favorites</topicDescription>
    <parentName>Hiking</parentName>
    <position>0</position>
  </ManagedTopic>
  <ManagedTopic>
    <name>Backpacks</name>
    <managedTopicType>Navigational</managedTopicType>
    <topicDescription>Recommended models</topicDescription>
    <parentName>Hiking</parentName>
    <position>1</position>
  </ManagedTopic>
  <ManagedTopic>
    <name>Footwear</name>
    <managedTopicType>Featured</managedTopicType>
    <topicDescription>Suggested types for each sport</topicDescription>
    <parentName></parentName>
    <position>0</position>
  </ManagedTopic>
  <ManagedTopic>
    <name>Conditioning</name>
    <managedTopicType>Featured</managedTopicType>
    <topicDescription>How to get fit for any activity</topicDescription>
    <parentName></parentName>
    <position>0</position>
  </ManagedTopic>
</ManagedTopics>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

MatchingRule

Represents a matching rule that is used to identify duplicate records. This type extends the `Metadata` metadata type and inherits its `fullName` field.

File Suffix and Directory Location

Matching rule components have the `.matchingRule` suffix and are stored in the `matchingRules` folder. The name of the component file is the standard or custom object name that is associated with the matching rule.

In API version 39.0 and later, MatchingRule supports the Person Account object.

- The component file name is `PersonAccount.matchingRule`.
- The component directory is `matchingRules`.

Version

MatchingRule is available in API version 33.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Specifies filter logic conditions.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the matching rule.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The name of the matching rule.</td>
</tr>
<tr>
<td>matchingRuleItems</td>
<td><code>MatchingRuleItem</code></td>
<td>The criteria that make up a matching rule.</td>
</tr>
<tr>
<td>ruleStatus</td>
<td><code>MatchingRuleStatus</code> (enumeration of type string)</td>
<td>Required. The activation status of the matching rule. Values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Inactive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Deactivating</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DeactivationFailed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Active</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Activating</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ActivationFailed</td>
</tr>
</tbody>
</table>

593
### MatchingRuleItem

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| blankValueBehavior | BlankValueBehavior (enumeration of type string) | Specifies how blank fields affect whether the fields being compared are considered matches. Valid values are:  
  - MatchBlanks  
  - NullNotAllowed (default) |
| fieldName          | string                                          | Required. Indicates which field to compare when determining if a record is similar enough to an existing record to be considered a match. |
| matchingMethod     | MatchingMethod (enumeration of type string)      | Required. Defines how the fields are compared. Choose between the exact matching method and various fuzzy matching methods. Valid values are:  
  - Exact  
  - FirstName  
  - LastName  
  - CompanyName  
  - Phone  
  - City  
  - Street  
  - Zip  
  - Title  
  For details on each matching method, see "Matching Methods Used with Matching Rules" in the Salesforce Help. |

### Declarative Metadata Sample Definition

The following is a sample XML definition of a matching rule. A matching rule can be associated with either a standard or a custom object.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<MatchingRules xmlns="http://soap.sforce.com/2006/04/metadata">
  <matchingRules>
    <fullName>AccountMatchingRule</fullName>
    <label>Matching rule for accounts</label>
    <description>this is sample rule description</description>
    <matchingRuleItems>
      <blankValueBehavior>NullNotAllowed</blankValueBehavior>
      <fieldName>BillingCity</fieldName>
      <matchingMethod>City</matchingMethod>
    </matchingRuleItems>
  </matchingRules>
</MatchingRules>
```
The following `package.xml` shows how to reference a matching rule by name. It specifies the type name of MatchingRule.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <name>MatchingRule</name>
    <members>Account.AccountMatchingRule</members>
  </types>
  <version>46.0</version>
</Package>
```

The following `package.xml` shows how to reference all matching rules by specifying the plural MatchingRules type name and using a wildcard to include all members.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <name>MatchingRules</name>
    <members>*</members>
  </types>
  <version>46.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**Metadata**

This is the base class for all metadata types. You cannot edit this object. A component is an instance of a metadata type. Metadata is analogous to `sObject`, which represents all standard objects. Metadata represents all components and fields in Metadata API. Instead of identifying each component with an ID, each custom object or custom field has a unique `fullName`, which must be distinct from standard object names, as it must be when you create custom objects or custom fields in the Salesforce user interface.

**Version**

Metadata components are available in API version 10.0 and later.
**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The name of the component. If a field, the name must specify the parent object, for example Account.FirstName. The __c suffix must be appended to custom object names and custom field names when you are setting the fullName. For example, a custom field in a custom object could have a fullName of MyCustomObject__c.MyCustomField__c. To reference a component in a package, prepend the package's namespace prefix to the component name in the fullName field. Use the following syntax: namespacePrefix__ComponentName. For example, for the custom field component MyCustomObject__c.MyCustomField__c and the namespace MyNS, the full name is MyNS__MyCustomObject__c.MyCustomField__c.</td>
</tr>
</tbody>
</table>

**Note:** A namespace prefix is a 1 to 15-character alphanumeric identifier that distinguishes your package and its contents from other publishers’ packages. For more information, see “Register a Namespace Prefix” in the Salesforce Help.

---

**Wildcard Support in the Manifest File**

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- CustomObject
- CustomField
- MetadataWithContent

**MetadataWithContent**

MetadataWithContent is the base type for all metadata types that contain content, such as documents or email templates. It extends Metadata. You cannot edit this object.

**Version**

MetadataWithContent components are available in API version 14.0 and later.
**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
<td>base64Binary</td>
<td>Base 64-encoded binary data. Prior to making an API call, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to binary. This conversion is usually handled for you by a SOAP client.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The name of the component. The <code>fullName</code> can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. Inherited from the Metadata component, this field is not defined in the WSDL for this component. It must be specified when creating, updating, or deleting. See <code>create()</code> to see an example of this field specified for a call.</td>
</tr>
</tbody>
</table>

**Wildcard Support in the Manifest File**

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- Metadata

**MilestoneType**

Represents the name and description of a milestone, which you can use in an entitlement process to track important steps in cases. This type extends the Metadata metadata type and inherits its `fullName` field.

**File Suffix and Directory Location**

Milestone types are stored in the `milestoneTypes` directory of the corresponding package directory. The extension is `.milestoneType`.

**Version**

MilestoneType is available in API version 27.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the milestone.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RecurrenceType</td>
<td>MilestoneRecurrenceType</td>
<td>The type of recurrence for the milestone. Available in API version 29.0 and later. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• none—Specifies no recurrence for the milestone. The milestone occurs only once until the entitlement process exits.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• recursIndependently—Specifies independent recurrence for the milestone.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• recursChained—Specifies sequential recurrence for the milestone.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

This is a sample milestone type.

```
<?xml version="1.0" encoding="UTF-8"?>
<MilestoneType xmlns="http://soap.sforce.com/2006/04/metadata">
    <description>First Response Time</description>
</MilestoneType>
```

And, here’s the sample `package.xml` file that references the `MilestoneType` component definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>* or a valid name of a milestone type</members>
        <name>MilestoneType</name>
    </types>
    <version>29.0</version>
</Package>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### MlDomain (Beta)

Represents an Einstein Intent Set. This type extends the `Metadata` metadata type and inherits its `fullName` field.

NLU features are preview and aren’t part of the “Services” under your master subscription agreement with Salesforce. Use this feature at your sole discretion, and make your purchase decisions only on the basis of generally available products and features. Salesforce doesn’t guarantee general availability of this feature within any particular time frame or at all, and we can discontinue it at any time. This feature is for evaluation purposes only, not for production use. It’s offered as is and isn’t supported, and Salesforce has no liability for any harm or damage arising out of or in connection with it. All restrictions, Salesforce reservation of rights, obligations concerning the Services, and terms for related Non-Salesforce Applications and Content apply equally to your use of this feature.
File Suffix and Directory Location

MLDomain components have the suffix `.mlDomain` and are stored in the `mlDomains` folder.

Version

MLDomain components are available in API version 43.0 and later.

Special Access Rules

This object is available only if Chat and Einstein Bots are enabled in your org.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Einstein Intent Set description.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Einstein Intent Set name.</td>
</tr>
<tr>
<td>mlIntents</td>
<td>MlIntent[]</td>
<td>List of intents under this Einstein Intent Set.</td>
</tr>
<tr>
<td>mlSlotClasses</td>
<td>MlSlotClass[]</td>
<td>List of entities under this Einstein Intent Set.</td>
</tr>
</tbody>
</table>

MlIntent

An intent in an Einstein Intent Set.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Einstein Intent Set description.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. This unique name prevents conflicts with other Einstein Intent Sets associated with the same bot version. This name can contain only underscores and alphanumeric characters and must be unique in your org. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Einstein Intent Set name.</td>
</tr>
<tr>
<td>mlIntentUtterances</td>
<td>MlIntentUtterance[]</td>
<td>List of customer inputs for this intent.</td>
</tr>
<tr>
<td>relatedMlIntents</td>
<td>MlRelatedIntent[]</td>
<td>List of intents within an Einstein Intent Set used to expand customer inputs for this intent. Only intents within local Einstein Intent Sets have related intents.</td>
</tr>
</tbody>
</table>

MlIntentUtterance

A customer input for this intent.
### MlRelatedIntent

An intent in an Einstein Intent Set used to expand customer inputs for this intent. Only intents within local Einstein Intent Sets have related intents.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relatedMlIntent</td>
<td>string</td>
<td>Name of the intent that is used to extend the customer inputs of the current parent intent.</td>
</tr>
</tbody>
</table>

### MlSlotClass

An entity in this Einstein Intent Set.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>A description of an Einstein Bot entity.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. This unique name prevents conflicts with other entities in an Einstein Intent Set. This name can contain only underscores and alphanumeric characters, and must be unique in your org. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.</td>
</tr>
<tr>
<td>extractionRegex</td>
<td>string</td>
<td>Regular expression used to extract an entity when the type is set to Pattern.</td>
</tr>
</tbody>
</table>
| extractionType  | MlSlotClassExtractionType (enumeration of type string) | Required. Valid values are:  
  • Pattern  
  • Value |
| label           | string         | Label that identifies an entity throughout the Salesforce user interface.     |
| mlSlotClassValues | MlSlotClassValue[] | List of entity values associated with an entity of type Value.               |

### MlSlotClassValue

An entity value associated with an entity of type Value.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>synonymGroup</td>
<td>SynonymGroup</td>
<td>Represents a list of terms or synonyms for the current entity value.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Single value used to extract an entity of type Value.</td>
</tr>
</tbody>
</table>
SynonymGroup

Represents a group of synonymous words or phrases.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>languages</td>
<td>Language (enumeration of type string)</td>
<td>Required. Specifies the languages the value list applies to. If value list items are specific to a single language, specify only that language. If the value list items apply to multiple languages, specify multiple languages for one value list.</td>
</tr>
<tr>
<td>terms</td>
<td>string</td>
<td>Required. A word or phrase synonymous with other terms in the value list.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of an MlDomain.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<MlDomain xmlns="http://soap.sforce.com/2006/04/metadata">
  <label>TestDomainMetadata</label>
  <description>This is domain 2 for metadata testing</description>
  <mlIntents>
    <developerName>Test_Intent_New</developerName>
    <label>Test Intent New</label>
    <mlIntentUtterances>
      <utterance>Utterance Hello</utterance>
    </mlIntentUtterances>
    <mlIntentUtterances>
      <utterance>Utterance Hi</utterance>
    </mlIntentUtterances>
    <mlIntentUtterances>
      <utterance>Utterance What</utterance>
    </mlIntentUtterances>
  </mlIntents>
  <mlIntents>
    <developerName>Test_Intent_New2</developerName>
    <label>Test Intent New 2</label>
  </mlIntents>
  <mlSlotClasses>
    <developerName>Test_Entity1</developerName>
    <label>Test Entity 1</label>
    <extractionType>Value</extractionType>
    <mlSlotClassValues>
      <value>Choice value 1</value>
    </mlSlotClassValues>
    <mlSlotClassValues>
      <value>Choice value 2</value>
    </mlSlotClassValues>
  </mlSlotClasses>
  <mlSlotClasses>
    <developerName>Test_Entity2</developerName>
    <label>Test Entity 2</label>
    <extractionType>Pattern</extractionType>
  </mlSlotClasses>
</MlDomain>
```
The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>TestDomainMetadata</members>
    <name>MlDomain</name>
  </types>
  <version>43.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**ModerationRule**

Represents a rule used in your community to moderate member-generated content. Each rule specifies the member-generated content the rule applies to, the criteria to enforce the rule on, and the moderation action to take. Moderation rules help protect your community from spammers, bots, and offensive or inappropriate content. This type extends the `Metadata` metadata type and inherits its `fullName` field.

Community moderation rules created with the Metadata API are more powerful than moderation rules set up in the Community Management UI. For example, in the UI you could create a rule that moderates posts and comments. In the Metadata API you could create a rule that moderates only the Link Name of a Link feed type. Use the Metadata API to express complex moderation rules.

**Important:** Don’t update moderation rules you create using the Metadata API in the Community Management UI. If you do, you overwrite relevant Metadata API fields or the fields are ignored.
Keep the following things in mind when creating moderation rules:

- Your org can have up to 30 rules. This limit is per org, not per community. This limit includes both content rules and rate rules.
- Each rule can have up to three keyword criteria.
- Rules that block content run first, followed by rules to review and approve content, then rules that replace content, and last by rules that flag content. If two or more rules perform the same action, the oldest rule runs first, based on the date the rule was created. Rules to replace content don’t run when the content also applies to a review rule—we want community managers to review the original content.

**File Suffix and Directory Location**

ModerationRule components have the suffix .rule and are stored in the moderation directory of the corresponding package directory. The file name format follows community_name.moderation_rule_developer_name.rule.

**Version**

ModerationRule components are available in API version 36.0 and later.

**Special Access Rules**

To view, create, edit, and delete moderation rules, you need the Manage Communities or Create and Set Up Communities permission.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>ModerationRuleAction</td>
<td>Required. Indicates the moderation action that you want to take. The valid values are:</td>
</tr>
<tr>
<td>actionLimit</td>
<td>int</td>
<td>Indicates the moderation action limit. Available in API 39.0 and later.</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Indicates whether the moderation rule is active (true) or inactive (false).</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the moderation rule.</td>
</tr>
<tr>
<td>entitiesAndFields</td>
<td>ModeratedEntityField[]</td>
<td>Indicates the types of user-generated content this moderation rule applies to.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Label for the moderation rule.</td>
</tr>
<tr>
<td>notifyLimit</td>
<td>int</td>
<td>Indicates the notification limit of the moderation rule. Available in API 39.0 and later.</td>
</tr>
</tbody>
</table>
**userCriteria**

Represents the member criteria to use in community moderation rules. Available in API 39.0 and later.

**userMessage**

The message you want your community members to see when their content is blocked. Use the `%BLOCKED_KEYWORD%` variable to display up to five blocked words in the user message. If you don’t specify a message, the member sees the standard message: "You can’t use %BLOCKED_KEYWORD% or other inappropriate words in this community. Review your content and try again."

---

**ModeratedEntityField**

The fields and entities you want to moderate.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityName</td>
<td>string</td>
<td>Required. Indicates the types of user-generated content the moderation rule applies to. Post and comments only apply to content created in groups and user profiles. All feed types, such as polls and links, are supported.</td>
</tr>
<tr>
<td>fieldName</td>
<td>string</td>
<td>Indicates the field the moderation rule applies to.</td>
</tr>
</tbody>
</table>

**Note:** To moderate feed posts, use `entityName FeedItem` with `fieldName RawBody`. To moderate feed comments, use `entityName FeedComment` with `fieldName RawCommentBody`. The `RawBody` and `RawCommentBody` fields aren’t available in any other API.

| keywordList  | KeywordList string | Indicates the keyword list that you want to moderate against. |

**ModerationRuleType**

Required. Indicates the type of rule to run on user-generated content.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>(enumeration of type string)</td>
<td>Required. Indicates the type of rule to run on user-generated content. Valid values are:</td>
</tr>
</tbody>
</table>

- Content
- Rate

Available in API 39.0 and later.

**RateLimitTimePeriod**

Required. Indicates the time period that is applied to the rate limit.
Description Field Type

Field Name | Field Type | Description
--- | --- | ---
timePeriod | (enumeration of type string) | Required. Indicates the time period that is applied to the rate limit. Valid values are:
  - Short
  - Medium
  Available in API 39.0 and later.

**Declarative Metadata Sample Definition**

The following is an example of a ModerationRule component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ModerationRule xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>Blocks Bad Word List in posts, comments, Link URLs, titles, and poll choices.</description>
  <masterLabel>Blocking Rule</masterLabel>
  <action>Block</action>
  <active>true</active>
  <userMessage>You can’t use %BLOCKED_KEYWOD% or other inappropriate words in this community. Review your content and try again.</userMessage>
  <!-- Applies the rule to FeedComment.RawCommentBody (an internal only field), if it contains words from the keyword list specified -->
  <entitiesAndFields>
    <entityName>FeedComment</entityName>
    <fieldName>RawCommentBody</fieldName>
    <keywordList>community1.badword_list</keywordList>
  </entitiesAndFields>
  <!-- Applies the rule to FeedItem.LinkUrl (an internal only field), if it contains words from the keyword list specified -->
  <entitiesAndFields>
    <entityName>FeedItem</entityName>
    <fieldName>LinkUrl</fieldName>
    <keywordList>community1.badword_list</keywordList>
  </entitiesAndFields>
  <!-- Applies the rule to FeedItem.RawBody (an internal only field), if it contains words from the keyword list specified -->
  <entitiesAndFields>
    <entityName>FeedItem</entityName>
    <fieldName>RawBody</fieldName>
    <keywordList>community1.badword_list</keywordList>
  </entitiesAndFields>
  <!-- Applies the rule to FeedItem.Title (an internal only field), if it contains words from the keyword list specified -->
  <entitiesAndFields>
    <entityName>FeedItem</entityName>
    <fieldName>Title</fieldName>
    <keywordList>community1.badword_list</keywordList>
  </entitiesAndFields>
  <!-- Applies the rule to FeedPollChoice (an internal only field), if it contains words from the keyword list specified -->
  <entitiesAndFields>
    <entityName>FeedPollChoice</entityName>
    <fieldName>ChoiceBody</fieldName>
    <keywordList>community1.badword_list</keywordList>
  </entitiesAndFields>
</ModerationRule>
```
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <name>ModerationRule</name>
    <members>community1.blocking_rule</members>
  </types>
  <version>36.0</version>
</Package>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

NamedCredential

Represents a named credential, which specifies the URL of a callout endpoint and its required authentication parameters in one definition. A named credential can be specified as an endpoint to simplify the setup of authenticated callouts.

This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

NamedCredential components have the suffix .namedCredential and are stored in the namedCredentials folder.

Version

NamedCredential components are available in API version 33.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>awsAccessKey</td>
<td>string</td>
<td>First part of the access key used to sign programmatic requests to AWS. Use when AWS Signature Version 4 is your authentication protocol. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>awsAccessSecret</td>
<td>string</td>
<td>The second part of the access key used to sign programmatic requests to AWS. Use when AWS Signature Version 4 is your authentication protocol. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>awsRegion</td>
<td>string</td>
<td>Specifies which AWS Region the named credential accesses. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>awsService</td>
<td>string</td>
<td>Specifies which AWS resource the named credential accesses. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>allowMergeFieldsInBody</td>
<td>boolean</td>
<td>Specifies whether Apex code can use merge fields to populate the HTTP request body with org data when a callout is made. Corresponds to <strong>Allow Merge Fields in HTTP Body</strong> in the user interface. Defaults to false. This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>allowMergeFieldsInHeader</td>
<td>boolean</td>
<td>Specifies whether Apex code can use merge fields to populate the HTTP header with org data when a callout is made. Corresponds to <strong>Allow Merge Fields in HTTP Header</strong> in the user interface. Defaults to false. This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>authProvider</td>
<td>string</td>
<td>The authentication provider that the AuthProvider component represents.</td>
</tr>
<tr>
<td>authTokenEndpointUrl</td>
<td>string</td>
<td>The URL where JWTs are exchanged for access tokens. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>certificate</td>
<td>string</td>
<td>If you specify a certificate, your Salesforce org supplies it when establishing each two-way SSL connection with the external system. The certificate is used for digital signatures, which verify that requests are coming from your Salesforce org.</td>
</tr>
<tr>
<td>endpoint</td>
<td>string</td>
<td>The URL or root URL of the callout endpoint. Corresponds to <strong>URL</strong> in the user interface.</td>
</tr>
<tr>
<td>generateAuthorizationHeader</td>
<td>boolean</td>
<td>Specifies whether Salesforce generates an authorization header and applies it to each callout that references the named credential. Corresponds to <strong>Generate Authorization Header</strong> in the user interface. Defaults to true. This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>jwtAudience</td>
<td>string</td>
<td>External service or other allowed recipients for the JWT. Written as JSON, with a quoted string for a single audience and an array of quoted strings for multiple audiences. Single audience example: &quot;aud1&quot; Multiple audiences example: [&quot;aud1&quot;, &quot;aud2&quot;, &quot;aud3&quot;]</td>
</tr>
<tr>
<td>jwtFormulaSubject</td>
<td>string</td>
<td>Formula string calculating the JWT's Subject. API names and constant strings, in single quotes, can be included. Allows a dynamic Subject unique per user requesting the token. For example, 'User='+${User.Id} Use this field when principalType is set to PerUser. Corresponds to <strong>Per User Subject</strong> in the user interface. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>jwtIssuer</td>
<td>string</td>
<td>Specify who issued the JWT using a case-sensitive string. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>jwtSigningCertificate</td>
<td>string</td>
<td>Certificate verifying the JWT's authenticity to external sites. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>jwtTextSubject</td>
<td>string</td>
<td>Static text, without quotes, that specifies the JWT Subject. Use this field when principalType is set to NamedUser. Corresponds to</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>jwtValidityPeriodSeconds</td>
<td>string</td>
<td>Specify the number of seconds that the token is valid. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>A user-friendly name for the named credential that appears in the Salesforce user interface, such as in list views.</td>
</tr>
<tr>
<td>oauthRefreshToken</td>
<td>string</td>
<td>The OAuth refresh token. Used to obtain a new access token for an end user when a token expires.</td>
</tr>
<tr>
<td>oauthScope</td>
<td>string</td>
<td>Specifies the scope of permissions to request for the access token. Corresponds to <strong>Scope</strong> in the user interface.</td>
</tr>
<tr>
<td>oauthToken</td>
<td>string</td>
<td>The access token that’s issued by your authorization server.</td>
</tr>
<tr>
<td>password</td>
<td>string</td>
<td>The password to be used by your org to access the external system. Ensure that the credentials have adequate privileges to access the external system. Depending on how you set up access, you might need to provide the administrator password.</td>
</tr>
</tbody>
</table>
| principalType              | External PrincipalType (enumeration of type string) | Determines whether you’re using one set or multiple sets of credentials to access the external system. Corresponds to **Identity Type** in the user interface. The valid values are:  
  - Anonymous  
  - PerUser  
  - NamedUser |
| protocol                   | Authentication Protocol (enumeration of type string) | The authentication protocol that’s required to access the external system. The valid values are:  
  - AwsSig4  
  - Jwt  
  - JwtExchange  
  - NoAuthentication  
  - Oauth  
  - Password  
  
  For connections to Amazon Web Services using Signature Version 4, use **AwsSig4**.  
  
  For connections using a direct token system, select **Jwt**. If using an intermediary authorization provider to process JWTs and return access tokens, use **JwtExchange**.  
  
  For Simple URL data sources, select **NoAuthentication**.  
  
  For cloud-based Files Connect external systems, select **Oauth**. For on-premises systems, select **Password**. |
### Declarative Metadata Sample Definition

The following is an example of a NamedCredential component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<NamedCredential xmlns="http://soap.sforce.com/2006/04/metadata">
  <allowMergeFieldsInBody>false</allowMergeFieldsInBody>
  <allowMergeFieldsInHeader>false</allowMergeFieldsInHeader>
  <endpoint>https://my_endpoint.example.com</endpoint>
  <generateAuthorizationHeader>true</generateAuthorizationHeader>
  <label>My Named Credential</label>
  <principalType>Anonymous</principalType>
  <protocol>NoAuthentication</protocol>
</NamedCredential>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### Network

Represents a community. Communities are branded spaces for your employees, customers, and partners to connect. You can customize and create communities to meet your business needs, then transition seamlessly between them. Use the Network component for Salesforce Communities. If you want to create zones that contain Chatter Answers and Ideas, use the Community (Zone) component.

This type extends the `Metadata` metadata type and inherits its `fullName` field.

### Declarative Metadata File Suffix and Directory Location

Network components are stored in the `networks` directory of the corresponding package directory. The file name matches the community name, and the extension is `.network`.

### Version

This object is available in API version 28.0 and later.
### Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowedExtensions</td>
<td>string</td>
<td>Specifies the types of files allowed in your community. This whitelist of file types lets you control what your community members upload and also prevents spammers from polluting your community with inappropriate files. Available in API version 36.0 and later.</td>
</tr>
<tr>
<td>allowInternalUserLogin</td>
<td>boolean</td>
<td>Determines whether internal users can log in with their internal credentials on the community login page. Available in API version 40.0 and later.</td>
</tr>
<tr>
<td>allowMembersToFlag</td>
<td>boolean</td>
<td>Determines whether users in the community can flag posts or comments as inappropriate. Flagged items are sent to a community moderator for review. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>branding</td>
<td>Branding</td>
<td>The color scheme, header, and footer used in the community. Deprecated in API version 41.0 and later. Replaced by the NetworkBranding type.</td>
</tr>
<tr>
<td>caseCommentEmailTemplate</td>
<td>string</td>
<td>Email template used when notifying community members when a case comment has been modified or added to a case. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>changePasswordTemplate</td>
<td>string</td>
<td>Email template used when notifying a user that their password has been reset. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>communityRoles</td>
<td>CommunityRoles</td>
<td>Identifies users with Customer, Partner, or Employee roles in a community. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the community.</td>
</tr>
<tr>
<td>disableReputationRecordConversations</td>
<td>boolean</td>
<td>Determines whether to exclude contributions to records when counting points toward reputation levels, if reputation levels are enabled for the community. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>emailFooterLogo</td>
<td>string</td>
<td>The document name of the logo that appears in the footer of community emails. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>emailFooterText</td>
<td>string</td>
<td>The text that appears in the footer of community emails. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>emailSenderAddress</code></td>
<td>string</td>
<td>Required. Email address from which community emails are sent.</td>
</tr>
<tr>
<td><code>emailSenderName</code></td>
<td>string</td>
<td>Required. Name from which community emails are sent.</td>
</tr>
<tr>
<td><code>enableCustomVFError</code></td>
<td>boolean</td>
<td>Determines whether to use custom Visualforce error pages instead of the default Visualforce error pages. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td><code>enableDirectMessages</code></td>
<td>boolean</td>
<td>Determines whether community users can send direct messages to start a private conversation with one or more members. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td><code>enableGuestChatter</code></td>
<td>boolean</td>
<td>Specifies whether guest users can access public Chatter groups in the community without logging in.</td>
</tr>
<tr>
<td><code>enableGuestFileAccess</code></td>
<td>boolean</td>
<td>Determines whether guest users view asset files shared with the community on publicly accessible pages and login pages. If public access is enabled in Community Builder at the page or community level, this property is automatically enabled. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td><code>enableInvitation</code></td>
<td>boolean</td>
<td>Determines whether users can invite others to the community.</td>
</tr>
<tr>
<td><code>enableKnowledgeable</code></td>
<td>boolean</td>
<td>Determines if community members can see who’s knowledgeable on topics and endorse people for their knowledge on a topic. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td><code>enableNicknameDisplay</code></td>
<td>boolean</td>
<td>Determines if user nicknames display instead of their first and last names in most places in the community. Set to false by default. Available in API version 32.0 and later.</td>
</tr>
<tr>
<td><code>enablePrivateMessages</code></td>
<td>boolean</td>
<td>Determines if community members can send and receive private messages. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td><code>enableReputation</code></td>
<td>boolean</td>
<td>Determines if reputation is calculated and displayed for community members. Available in API version 31.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If enabled, <code>reputationLevels</code> and <code>reputationPointsRules</code> are used. If no <code>reputationLevels</code> or <code>reputationPointsRules</code> are not defined in the data file, the default values are used.</td>
</tr>
<tr>
<td><code>enableShowAllNetworkSettings</code></td>
<td>boolean</td>
<td>Shows settings that are hidden by default based on how the community is set up. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>enableSiteAsContainer</td>
<td>boolean</td>
<td>Determines whether the community uses Site.com pages instead of Visualforce tabs.</td>
</tr>
<tr>
<td>enableTalkingAboutStats</td>
<td>boolean</td>
<td>Determines whether community users see how many people are discussing a topic. The number of people discussing the topic appears as the user types the topic and the system gives topic suggestions. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>enableTopicAssignmentRules</td>
<td>boolean</td>
<td>Enables the org to use rules to automatically assign topics to articles in a community. After it’s enabled, admins set up rules in a community to map topics to Salesforce Knowledge data categories. This field is available in API version 40.0 and later.</td>
</tr>
<tr>
<td>enableTopicSuggestions</td>
<td>boolean</td>
<td>Enables topic suggestions when users write posts. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>enableUpDownVote</td>
<td>boolean</td>
<td>Replaces the option to like a question or answer with the option to upvote or downvote. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>feedChannel</td>
<td>string</td>
<td>Displays the feed of all channel program record or group interactions, including posts, questions, and attachments. This field is available in API version 28.0 and later.</td>
</tr>
<tr>
<td>forgotPasswordTemplate</td>
<td>string</td>
<td>Required. The email template used when a user forgets their password. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>gatherCustomerSentimentData</td>
<td>boolean</td>
<td>Gathers data when a customer looks at articles and cases in communities, for use in the Community 360 feature. This field is available in API version 40.0 and later.</td>
</tr>
<tr>
<td>lockoutTemplate</td>
<td>string</td>
<td>The email template used to communicate with users when they get locked out of their org because of too many failed login attempts. Available in API version 43.0 and later. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>logoutUrl</td>
<td>string</td>
<td>Specifies the URL that community members are redirected to when they log out from your community. This field is available in API version 28.0 and later.</td>
</tr>
<tr>
<td>maxFileSizeKb</td>
<td>int</td>
<td>Specifies the maximum file size (in KBs) that members can upload in your community. Available in API version 36.0 and later. Enter a number between 3072 KB and your org’s maximum file size. To use the default limit of 2 GB, leave this field empty.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>navigationLinkSet</td>
<td>NavigationLinkSet</td>
<td>Represents the navigation menu in a community. A navigation menu consists of items that users can click to go to other parts of the community. This field is available in API version 37.0 and later.</td>
</tr>
<tr>
<td>networkMemberGroups</td>
<td>NetworkMemberGroups</td>
<td>The profiles and permission sets that have access to the community. Users with these profiles or permission sets are members of the community.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> If a Chatter customer (from a customer group) is assigned a permission set that is also associated with a community, the Chatter customer isn’t added to the community.</td>
</tr>
<tr>
<td>networkPageOverrides</td>
<td>NetworkPageOverride</td>
<td>The settings in the Administration area (in Community Management or Community Workspaces) that control which page type Change Password, Forgot Password, Home, and Login each point to. Available in API version 40.0 and later.</td>
</tr>
<tr>
<td>newSenderAddress</td>
<td>string</td>
<td>Email address that has been entered as the new value for EmailSenderAddress but has not been verified yet. After a user has requested to change the sender email address and has successfully responded to the verification email, the NewSenderAddress value overwrites the value in EmailSenderAddress. This value becomes the email address from which community emails are sent.</td>
</tr>
<tr>
<td>enableMemberVisibility</td>
<td>boolean</td>
<td>Controls community user visibility on a per-community basis. If true, the Community User Visibility preference is enabled for the selected community. Available in API version 45.0 and later.</td>
</tr>
<tr>
<td>picassoSite</td>
<td>string</td>
<td>Name of the Site.com site linked to the community.</td>
</tr>
<tr>
<td>recommendationAudience</td>
<td>RecommendationAudience</td>
<td>Creates an audience of new community members, or can be used to manage customized lists of audience members to organize and target recommendations. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>recommendationDefinition</td>
<td>RecommendationDefinition</td>
<td>Represents a custom recommendation to drive engagement for a community. Targets a specific audience and uses channels to specify a location for the recommendation. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>reputationLevels</td>
<td>ReputationLevelDefinitions</td>
<td>The reputation levels assigned to members when they accrue points by performing certain actions.</td>
</tr>
<tr>
<td>reputationPointsRules</td>
<td>ReputationPointsRules</td>
<td>The points members accrue when they perform certain defined actions.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>selfRegProfile</td>
<td>string</td>
<td>The profile assigned to users who self register. This value is used only if selfRegistration is enabled for the community. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>selfRegistration</td>
<td>boolean</td>
<td>Determines whether self-registration is available for the community.</td>
</tr>
<tr>
<td>sendWelcomeEmail</td>
<td>boolean</td>
<td>Determines whether a welcome email is sent when a new user is added to the community.</td>
</tr>
<tr>
<td>site</td>
<td>string</td>
<td>Required. The CustomSite associated with the community.</td>
</tr>
<tr>
<td>status</td>
<td>NetworkStatus[]</td>
<td>Required. Status of the community. Available values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Live — The community is online and members can access it.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- DownForMaintenance — The community was previously published but was taken offline. Members with “Create and Set Up Communities” can still access the setup for offline communities regardless of profile or membership. Members are not able to access offline communities, but they still appear in the user interface dropdown as CommunityName (Offline).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- UnderConstruction — The community has not yet been published. Users with “Create and Set Up Communities” can access communities in this status if their profile is associated with the community. After a community is published, it can never be in this status again.</td>
</tr>
<tr>
<td>tabs</td>
<td>NetworkTabSet</td>
<td>Required. The tabs that are available in the community. The user that created the community selected these tabs.</td>
</tr>
<tr>
<td>urlPathPrefix</td>
<td>string</td>
<td>The first part of the path on the site’s URL that distinguishes this site from other sites. For example, if your site URL is mycompany.force.com/partners, then partners is the urlPathPrefix.</td>
</tr>
<tr>
<td>verificationTemplate</td>
<td>string</td>
<td>The email template used to communicate with users when they must verify their identity, for example, when they log in without a password or from a new device. Available in API version 44.0 and later. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>welcomeTemplate</td>
<td>string</td>
<td>The email template used when sending welcome emails to new community members. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
</tbody>
</table>
## Branding

Represents the branding and color scheme applied to the community. Available in API version 40.0 and earlier. Replaced by **NetworkBranding** in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>loginFooterText</td>
<td>string</td>
<td>The text that appears in the footer of the community login page.</td>
</tr>
<tr>
<td>loginLogo</td>
<td>string</td>
<td>The logo that appears on the community login page for external users.</td>
</tr>
<tr>
<td>pageFooter</td>
<td>string</td>
<td>An image that appears on the footer of the community pages. Must be an .html file.</td>
</tr>
<tr>
<td>pageHeader</td>
<td>string</td>
<td>An image that appears on the header of the community pages. Can be an .html, .gif, .jpg, or .png file.</td>
</tr>
<tr>
<td>primaryColor</td>
<td>string</td>
<td>The color used for the active tab.</td>
</tr>
<tr>
<td>primaryComplementColor</td>
<td>string</td>
<td>Font color used with primaryColor.</td>
</tr>
<tr>
<td>quaternaryColor</td>
<td>string</td>
<td>The background color for pages in the community.</td>
</tr>
<tr>
<td>quaternaryComplementColor</td>
<td>string</td>
<td>Font color used with quaternaryColor.</td>
</tr>
<tr>
<td>secondaryColor</td>
<td>string</td>
<td>The color used for the top borders of lists and tables.</td>
</tr>
<tr>
<td>tertiaryColor</td>
<td>string</td>
<td>The background color for section headers on edit and detail pages.</td>
</tr>
<tr>
<td>tertiaryComplementColor</td>
<td>string</td>
<td>Font color used with tertiaryColor.</td>
</tr>
<tr>
<td>zeronaryColor</td>
<td>string</td>
<td>The background color for the header.</td>
</tr>
<tr>
<td>zeronaryComplementColor</td>
<td>string</td>
<td>Font color used with zeronaryColor.</td>
</tr>
</tbody>
</table>

## CommunityRoles

The labels used to identify users with Customer, Partner, or Employee roles in a community. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customerUserRole</td>
<td>string</td>
<td>The label for the Customer user role in a community.</td>
</tr>
<tr>
<td>employeeUserRole</td>
<td>string</td>
<td>The label for the Employee user role in a community.</td>
</tr>
</tbody>
</table>
### NavigationLinkSet

Represents the navigation menu in a community. A navigation menu consists of items that users can click to go to other parts of the community.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerUserRole</td>
<td>string</td>
<td>The label for the Partner user role in a community.</td>
</tr>
</tbody>
</table>

### NavigationMenuItem

Represents a single menu item in a NavigationLinkSet. Use this object to create, delete, or update menu items in your community’s navigation menu.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>navigationMenuItem</td>
<td>NavigationMenuItem[]</td>
<td>A list of menu items in a NavigationLinkSet. Use this object to create, delete, or update menu items in your community’s navigation menu.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultListViewId</td>
<td>string</td>
<td>If the value of the type field is SalesforceObject, the value is the ID of the default list view for the object.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The text that appears in the navigation menu for this item.</td>
</tr>
<tr>
<td>position</td>
<td>int</td>
<td>Required. The location of the menu item in the navigation menu.</td>
</tr>
<tr>
<td>publiclyAvailable</td>
<td>boolean</td>
<td>When set to true, gives access to guest users.</td>
</tr>
<tr>
<td>subMenu</td>
<td>NavigationSubMenu</td>
<td>A list of child menu items. This field is available in API 39.0 and later.</td>
</tr>
<tr>
<td>target</td>
<td>string</td>
<td>Required if type is ExternalLink, InternalLink, or SalesforceObject. If type is ExternalLink or InternalLink, the target is the URL that the link points to. For ExternalLink, your entry looks like this: <a href="http://www.salesforce.com">http://www.salesforce.com</a>. For InternalLink, use a relative URL, such as /contactsupport. If type is MenuLabel or...</td>
</tr>
</tbody>
</table>
### Metadata Types

#### Field

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetPreference</td>
<td>string</td>
<td>NavigationalTopic, target is not used. Backed by a picklist that includes preferences for the target field. Valid values are:</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>The type of navigation menu item. Required. Valid values are:</td>
</tr>
</tbody>
</table>

#### NavigationSubMenu

A list of child menu items. Only NavigationMenuItem items of type MenuLabel can have items in a NavigationSubMenu. Available in API 39.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>navigationMenuItems</td>
<td>NavigationMenuItem[]</td>
<td>A list of menu items in a NavigationSubMenu. Use</td>
</tr>
</tbody>
</table>
### NetworkMemberGroup

Represents the profiles and permission sets that are assigned to the community. Users with one of the profiles or permission sets are members of the community, unless the user is a Chatter customer (from a customer group).

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>permissionSet</td>
<td>string</td>
<td>A permission set that is assigned to the community.</td>
</tr>
<tr>
<td>profile</td>
<td>string</td>
<td>A profile that is part of the community.</td>
</tr>
</tbody>
</table>

**Note:** If a Chatter customer (from a customer group) is assigned a permission set that is also associated with a community, the Chatter customer isn’t added to the community.

### NetworkPageOverride

Represents settings in the Administration area (in Community Management or Community Workspaces) that control which page type the Change Password, Forgot Password, Home, and Login pages each point to.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| changePasswordPageOverrideSetting | NetworkPageOverrideSetting (enumeration of type string) | Required. Specifies the page type that the Change Password page setting applies to. The valid values are:  
  - Configurable—a Configurable Self-Reg page  
  - Designer—a Community Builder page  
  - Standard—the default page  
  - VisualForce—a Visualforce page |
| forgotPasswordPageOverrideSetting | NetworkPageOverrideSetting (enumeration of type string) | Required. Specifies the page type that the Forgot Password page setting applies to. The valid values are:  
  - Configurable—a Configurable Self-Reg page |
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[391x616]homePageOverrideSetting [451x616]</td>
<td>Required. Specifies the page type that the Community Home page setting applies to. The valid values are:</td>
</tr>
<tr>
<td></td>
<td>[405x674]Designer— a Community Builder</td>
<td>page</td>
</tr>
<tr>
<td></td>
<td>[451x674]Standard— the default page</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[468x674]VisualForce— a Visualforce page</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[468x603]loginPageOverrideSetting [451x603]</td>
<td>Required. Specifies the page type that the Login page setting applies to. The valid values are:</td>
</tr>
<tr>
<td></td>
<td>[405x641]Designer— a Community Builder</td>
<td>page</td>
</tr>
<tr>
<td></td>
<td>[451x641]Standard— the default page</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[468x641]VisualForce— a Visualforce page</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[468x317]selfRegProfilePageOverrideSetting</td>
<td>Required. Specifies the page type that the Self Registration page setting applies to. The valid values are:</td>
</tr>
<tr>
<td></td>
<td>[405x355]Designer— a Community Builder</td>
<td>page</td>
</tr>
<tr>
<td></td>
<td>[451x355]Standard— the default page</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[468x355]VisualForce— a Visualforce page</td>
<td></td>
</tr>
</tbody>
</table>

**RecommendationAudience**

Creates an audience of new community members, or can be used to manage customized lists of audience members to organize and target recommendations. Available in API version 41.0 and later.
## RecommendationAudienceDetail

The specific details of an audience for recommendations. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>audienceCriteriaType</td>
<td>AudienceCriteriaType (enumeration of type string)</td>
<td>The criteria for the recommendation audience type. Values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CustomList</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MaxDaysInCommunity</td>
</tr>
<tr>
<td>audienceCriteriaValue</td>
<td>string</td>
<td>For new member criteria, the maximum number of days since a user became a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>community member. Null in case of custom list criteria.</td>
</tr>
<tr>
<td>setupName</td>
<td>string</td>
<td>Name of the recommendation audience.</td>
</tr>
</tbody>
</table>

## RecommendationDefinition

Represents a list of custom recommendations to drive engagement for a community. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recommendationDetails</td>
<td>RecommendationDefinitionDetail[]</td>
<td>A list of custom recommendations and their details.</td>
</tr>
</tbody>
</table>

## RecommendationDefinitionDetail

The specific details of a custom recommendation. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionUrl</td>
<td>string</td>
<td>The URL for the button that lets users act on the recommendation.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>An explanation of the recommendation that suggests what users can do.</td>
</tr>
<tr>
<td>linkText</td>
<td>string</td>
<td>The text label for the button.</td>
</tr>
<tr>
<td>scheduledRecommendations</td>
<td>ScheduledRecommendation</td>
<td>A list of scheduled recommendations.</td>
</tr>
</tbody>
</table>
### ReputationLevel

Represents the name and lower value of the reputation level. The application calculates the upper value.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>branding</td>
<td>ReputationBranding[]</td>
<td>Represents any branding associated with the reputation level, specifically, the custom image for the reputation level. This field is optional. If not specified, the default reputation level image is used. Available in API version 32.0 and later.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Name of the reputation level. This field is optional. If not specified, one of the 10 defaults are used.</td>
</tr>
</tbody>
</table>
|         |                           | • Level 1  
|         |                           | • Level 2  
|         |                           | • Level 3  |
### ReputationPointsRules

Represents points rules in a community’s point system.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pointsRule</td>
<td>ReputationPointsRule[]</td>
<td>Represents events and their associated points.</td>
</tr>
</tbody>
</table>

### ReputationPointsRule

Represents the event and associated point value for a points rule. When a user acts, they accrue the associated points.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventType</td>
<td>string</td>
<td>Required. The type of event a member has to perform to get points. The available values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemWriteAPost</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemWriteAComment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemReceiveAComment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemLikeSomething</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemReceiveALike</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemMentionSomeone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemSomeoneMentionsYou</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemShareAPost</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemSomeoneSharesYourPost</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemPostAQuestion</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemAnswerAQuestion</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FeedItemReceiveAnAnswer</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FeedItemMarkAnswerAsBest</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FeedItemYourAnswerMarkedBest</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FeedItemEndorseSomeoneForKnowledgeOnATopic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FeedItemEndorsedForKnowledgeOnATopic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>points</td>
<td>int</td>
<td>Required. The number of points a member gets for performing the event. The default number of points per event is:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FeedItemWriteAPost</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>FeedItemWriteAComment</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>FeedItemReceiveAComment</td>
<td></td>
<td>+5</td>
</tr>
<tr>
<td>FeedItemLikeSomething</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>FeedItemReceiveALike</td>
<td></td>
<td>+5</td>
</tr>
<tr>
<td>FeedItemMentionSomeone</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>FeedItemSomeoneMentionsYou</td>
<td></td>
<td>+5</td>
</tr>
<tr>
<td>FeedItemShareAPost</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>FeedItemSomeoneSharesYourPost</td>
<td></td>
<td>+5</td>
</tr>
<tr>
<td>FeedItemPostAQuestion</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>FeedItemAnswerAQuestion</td>
<td></td>
<td>+5</td>
</tr>
<tr>
<td>FeedItemReceiveAnAnswer</td>
<td></td>
<td>+5</td>
</tr>
<tr>
<td>FeedItemMarkAnswerAsBest</td>
<td></td>
<td>+5</td>
</tr>
<tr>
<td>FeedItemYourAnswerMarkedBest</td>
<td></td>
<td>+20</td>
</tr>
<tr>
<td>FeedItemEndorseSomeoneForKnowledgeOnATopic</td>
<td></td>
<td>+5</td>
</tr>
<tr>
<td>FeedItemEndorsedForKnowledgeOnATopic</td>
<td></td>
<td>+20</td>
</tr>
</tbody>
</table>

ScheduledRecommendation

Represents a list of scheduled recommendations. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduledRecommendationDetails</td>
<td>ScheduledRecommendationDetail[]</td>
<td>A list of scheduled recommendations.</td>
</tr>
</tbody>
</table>

ScheduledRecommendationDetail

The specific details of a scheduled recommendation. Available in API version 41.0 and later.
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| channel | RecommendationChannel (enumeration of type string) | A way to group recommendations together to determine where they show up in the community. The valid values are:  
- **DefaultChannel**—The default recommendation channel. Recommendations in the default channel appear in predefined locations, such as directly in the feed in Salesforce mobile web and on the Home and Question Detail pages in communities using the Summer '15 or later version of the Customer Service (Napili) template.  
- **CustomChannel1**—A custom recommendation channel. Choose where you want recommendations to appear by adding the Recommendations Carousel component to the page in Community Builder.  
- **CustomChannel2**—A custom recommendation channel.  
- **CustomChannel3**—A custom recommendation channel.  
- **CustomChannel4**—A custom recommendation channel.  
- **CustomChannel5**—A custom recommendation channel. |
| enabled | boolean                             | Indicates whether scheduling is enabled. If **true**, the recommendation is enabled and appears in communities. If **false**, recommendations in feeds in Salesforce mobile web aren’t removed, but no new recommendations appear. In communities, disabled recommendations no longer appear. |
| rank    | int                                 | The rank of the recommendation within the channel, which determines the order in which it’s displayed. The scheduled recommendation is inserted into the position specified by the rank. The rank of all the scheduled recommendations after it is pushed down. If the specified rank is larger than the size of the list, the |
### Declarative Metadata Sample Definition

A sample XML definition of a network.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Network xmlns="http://soap.sforce.com/2006/04/metadata">
  <allowMembersToFlag>true</allowMembersToFlag>

  <changePasswordTemplate>unfiled$public/CommunityChangePasswordEmailTemplate</changePasswordTemplate>

  <description>Metadata Community</description>
  <emailSenderAddress>admin@networkMetadata.com</emailSenderAddress>
  <emailSenderName>Admin User</emailSenderName>
  <enableInvitation>false</enableInvitation>
  <enableKnowledgeable>true</enableKnowledgeable>
  <enableNicknameDisplay>false</enableNicknameDisplay>
  <enablePrivateMessages>true</enablePrivateMessages>
  <enableReputation>true</enableReputation>
  <enableUpDownVote>true</enableUpDownVote>

  <forgotPasswordTemplate>unfiled$public/CommunityForgotPasswordEmailTemplate</forgotPasswordTemplate>

  <networkMemberGroups>
    <permissionSet>Admin</permissionSet>
    <permissionSet>Standard</permissionSet>
    <permissionSet>ReadOnly</permissionSet>
    <profile>Admin</profile>
  </networkMemberGroups>
</Network>
```
<profile>Standard</profile>
<profile>ReadOnly</profile>
</networkMemberGroups>
<recommendationDefinition>
<recommendationDefinitionDetails>
<actionUrl>https://www.apple.com/iphone</actionUrl>
<description>Better specs and high performance for iPhones</description>
<linkText>iPhone 7</linkText>
<scheduledRecommendations>
<scheduledRecommendationDetails>
<channel>DefaultChannel</channel>
<enabled>false</enabled>
<rank>1</rank>
<recommendationAudience>New Member Audience</recommendationAudience>
</scheduledRecommendationDetails>
</scheduledRecommendations>
<setupName>Apple iPhone</setupName>
<title>iPhone 7</title>
</recommendationDefinitionDetails>
<recommendationDefinitionDetails>
<actionUrl>https://www.bose.com/qc35</actionUrl>
<description>New Amazing Noise cancellation Headphones</description>
<linkText>Bose QC35</linkText>
<scheduledRecommendations>
<scheduledRecommendationDetails>
<channel>DefaultChannel</channel>
<enabled>true</enabled>
<rank>2</rank>
<recommendationAudience>Custom Audience</recommendationAudience>
</scheduledRecommendationDetails>
</scheduledRecommendations>
<setupName>Bose Headphones</setupName>
<title>Bose QC35</title>
</recommendationDefinitionDetails>
</recommendationDefinition>
<reputationLevels>
<level>
<branding>
<smallImage>communities_shared_document_folder/replevel_beginner.png</smallImage>
</branding>
<label>Beginner</label>
<lowerThreshold>0</lowerThreshold>
</level>
<level>
<branding>
<smallImage>communities_shared_document_folder/replevel_apprentice.png</smallImage>
</branding>
<label>Apprentice</label>
<lowerThreshold>51</lowerThreshold>
</level>
</reputationLevels>
<reputationLevels>
  <level>
    <branding>
      <smallImage>communities_shared_document_folder/replevel_gettingthere.png</smallImage>
    </branding>
    <label>Getting There</label>
    <lowerThreshold>101</lowerThreshold>
  </level>
  <level>
    <branding>
      <smallImage>communities_shared_document_folder/replevel_skilled.png</smallImage>
    </branding>
    <label>Skilled</label>
    <lowerThreshold>151</lowerThreshold>
  </level>
  <level>
    <branding>
      <smallImage>communities_shared_document_folder/replevel_expert.png</smallImage>
    </branding>
    <label>Expert</label>
    <lowerThreshold>201</lowerThreshold>
  </level>
  <level>
    <branding>
      <smallImage>communities_shared_document_folder/replevel_mentor.png</smallImage>
    </branding>
    <label>Mentor</label>
    <lowerThreshold>251</lowerThreshold>
  </level>
  <level>
    <branding>
      <smallImage>communities_shared_document_folder/replevel_guru.png</smallImage>
    </branding>
    <label>Guru</label>
    <lowerThreshold>301</lowerThreshold>
  </level>
</reputationLevels>
<reputationPointsRules>
  <pointsRule>
    <eventType>FeedItemWriteAPost</eventType>
    <points>5</points>
  </pointsRule>
  <pointsRule>
    <eventType>FeedItemWriteAComment</eventType>
    <points>3</points>
  </pointsRule>
  <pointsRule>
    <eventType>FeedItemReceiveAComment</eventType>
    <points>10</points>
  </pointsRule>
  <pointsRule>
    <eventType>FeedItemLikeSomething</eventType>
  </pointsRule>
</reputationPointsRules>
<points>3</points>
</pointsRule>
<pointsRule>
  <eventType>FeedItemReceiveALike</eventType>
  <points>5</points>
</pointsRule>
<pointsRule>
  <eventType>FeedItemMentionSomeone</eventType>
  <points>5</points>
</pointsRule>
<pointsRule>
  <eventType>FeedItemSomeoneMentionsYou</eventType>
  <points>10</points>
</pointsRule>
<pointsRule>
  <eventType>FeedItemShareAPost</eventType>
  <points>5</points>
</pointsRule>
<pointsRule>
  <eventType>FeedItemSomeoneSharesYourPost</eventType>
  <points>10</points>
</pointsRule>
</reputationPointsRules>
<selfRegistration>false</selfRegistration>
<sendWelcomeEmail>true</sendWelcomeEmail>
<site>Network_11</site>
<status>UnderConstruction</status>
<tabs>
  <defaultTab>Chatter</defaultTab>
  <standardTab>Chatter</standardTab>
  <standardTab>Account</standardTab>
  <standardTab>Campaign</standardTab>
  <standardTab>Case</standardTab>
  <standardTab>Console</standardTab>
  <standardTab>Contact</standardTab>
  <standardTab>Contract</standardTab>
  <standardTab>Dashboard</standardTab>
  <standardTab>JigsawSearch</standardTab>
  <standardTab>File</standardTab>
  <standardTab>CollaborationGroup</standardTab>
  <standardTab>home</standardTab>
  <standardTab>Idea</standardTab>
  <standardTab>Lead</standardTab>
  <standardTab>Opportunity</standardTab>
  <standardTab>Product2</standardTab>
  <standardTab>UserProfile</standardTab>
  <standardTab>report</standardTab>
  <standardTab>Solution</standardTab>
</tabs>
<urlPathPrefix>network1</urlPathPrefix>
<welcomeTemplate>unfiled$public/CommunityWelcomeEmailTemplate</welcomeTemplate>
</Network>
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:

Community (Zone)

NetworkBranding

Represents the branding and color scheme applied to a community. (Salesforce Communities are represented by the Network component.) This type extends the MetadataWithContent type and inherits its content and fullName fields.

Declarative Metadata File Suffix and Directory Location

NetworkBranding components have the suffix .networkBranding and are stored in the networkBranding folder.

Version

This object is available in API version 41.0 and later. It replaces the Branding subtype in the Network component.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>loginBackgroundImageUrl</td>
<td>string</td>
<td>The path to the image URL that appears as the background on the community's login page. This URL can be fixed, dynamic, or an uploaded image. A dynamic URL contains the experience ID parameter, {expid}, and is resolved dynamically at runtime.</td>
</tr>
<tr>
<td>loginFooterText</td>
<td>string</td>
<td>The text that appears in the footer of the community login page.</td>
</tr>
<tr>
<td>loginLogo</td>
<td>string</td>
<td>The logo that appears on the community login page for external users.</td>
</tr>
<tr>
<td>loginLogoName</td>
<td>string</td>
<td>The name of the logo that appears on the community login page for external users.</td>
</tr>
<tr>
<td>loginLogoStaticImageUrl</td>
<td>string</td>
<td>The path to the logo that appears on the community's login page. This URL can be fixed, dynamic, or an uploaded image. A dynamic URL contains the experience ID parameter, {expid}. If the URL contains {expid}, the URL is resolved dynamically.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>loginQuaternaryColor</td>
<td>string</td>
<td>The background color for the community's login page.</td>
</tr>
<tr>
<td>loginRightFrameUrl</td>
<td>string</td>
<td>The path to the content of the right frame of the community login page. This URL can be either fixed or dynamic. A dynamic URL contains the experience ID parameter, {expid}. If the URL contains {expid}, the URL is resolved dynamically at runtime depending on the parameter's value.</td>
</tr>
<tr>
<td>network</td>
<td>string</td>
<td>The name of the community associated with the branding.</td>
</tr>
<tr>
<td>pageFooter</td>
<td>string</td>
<td>An image that appears on the footer of the community pages. Must be an .html file.</td>
</tr>
<tr>
<td>pageHeader</td>
<td>string</td>
<td>An image that appears on the header of the community pages. Can be an .html, .gif, .jpg, or .png file.</td>
</tr>
<tr>
<td>primaryColor</td>
<td>string</td>
<td>Required. The color used for the active tab.</td>
</tr>
<tr>
<td>primaryComplementColor</td>
<td>string</td>
<td>Required. Font color used with primaryColor.</td>
</tr>
<tr>
<td>quaternaryColor</td>
<td>string</td>
<td>Required. The background color for pages in the community.</td>
</tr>
<tr>
<td>quaternaryComplementColor</td>
<td>string</td>
<td>Required. Font color used with quaternaryColor.</td>
</tr>
<tr>
<td>secondaryColor</td>
<td>string</td>
<td>Required. The color used for the top borders of lists and tables.</td>
</tr>
<tr>
<td>tertiaryColor</td>
<td>string</td>
<td>Required. The background color for section headers on edit and detail pages.</td>
</tr>
<tr>
<td>tertiaryComplementColor</td>
<td>string</td>
<td>Required. Font color used with tertiaryColor.</td>
</tr>
<tr>
<td>zeronaryColor</td>
<td>string</td>
<td>Required. The background color for the header.</td>
</tr>
<tr>
<td>zeronaryComplementColor</td>
<td>string</td>
<td>Required. Font color used with zeronaryColor.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

A sample XML definition of network branding.

```xml
<?xml version="1.0" encoding="UTF-8"?><NetworkBranding xmlns="http://soap.sforce.com/2006/04/metadata">
    <loginFooterText>salesforce.com</loginFooterText>
    <loginLogo>Communities_Shared_Document_Folder/header2.png</loginLogo>
    <loginLogoName>header2.png</loginLogoName>
    <loginBackgroundImageUrl>http://identitycms.herokuapp.com/promo-background.jpg</loginBackgroundImageUrl>
    <loginQuaternaryColor>#B1BAC1</loginQuaternaryColor>
    <loginRightFrameUrl>http://www.test.com/test</loginRightFrameUrl>
    <network>Network 1</network>
    <pageFooter>Branding/footer_html.html</pageFooter>
    <pageHeader>Branding/header_image.jpg</pageHeader>
    <primaryColor>#AF5800</primaryColor>
    <primaryComplementColor>#FFFFFF</primaryComplementColor>
    <quaternaryColor>#286FB8</quaternaryColor>
    <quaternaryComplementColor>#FFFFFF</quaternaryComplementColor>
    <secondaryColor>#000000</secondaryColor>
    <tertiaryColor>#FFFFFF</tertiaryColor>
    <tertiaryComplementColor>#222222</tertiaryComplementColor>
    <zeronaryColor>#0A3764</zeronaryColor>
    <zeronaryComplementColor>#FFFFFF</zeronaryComplementColor>
</NetworkBranding>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Package

Specifies which metadata components to retrieve as part of a `retrieve()` call or defines a package of components.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiAccessLevel</td>
<td>APIAccessLevel (enumeration of type string)</td>
<td>Package components have access via dynamic Apex and the API to standard and custom objects in the organization where they are installed. Administrators who install packages may wish to restrict this access after installation for improved security. The valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Unrestricted—Package components have the same API access to standard objects as the user who is logged in when the component sends a request to the API.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Restricted—the administrator can select which standard objects the components can access. Further, the components in restricted packages can only access custom</td>
</tr>
</tbody>
</table>
### Metadata Types

#### Package

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>A short description of the package.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The package name used as a unique identifier for API access. The <code>fullName</code> can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the <code>Metadata</code> component.</td>
</tr>
<tr>
<td>namespacePrefix</td>
<td>string</td>
<td>The namespace of the developer organization where the package was created.</td>
</tr>
<tr>
<td>objectPermissions</td>
<td>ProfileObjectPermissions[]</td>
<td>Indicates which objects are accessible to the package, and the kind of access available (create, read, update, delete).</td>
</tr>
<tr>
<td>packageType</td>
<td>string</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>postInstallClass</td>
<td>string</td>
<td>The name of the Apex class that specifies the actions to execute after the package has been installed or upgraded. The Apex class must be a member of the package and must implement the Apex <code>InstallHandler</code> interface. In patch upgrades, you can't change the class name in this field but you can change the contents of the Apex class. The class name can be changed in major upgrades. This field is available in API version 24.0 and later.</td>
</tr>
<tr>
<td>setupWeblink</td>
<td>string</td>
<td>The weblink used to describe package installation.</td>
</tr>
<tr>
<td>types</td>
<td>PackageTypeMembers[]</td>
<td>The type of component being retrieved.</td>
</tr>
<tr>
<td>uninstallClass</td>
<td>string</td>
<td>The name of the Apex class that specifies the actions to execute after the package has been uninstalled. The Apex class must be a member of the package and must implement the Apex <code>UninstallHandler</code> interface. In patch upgrades, you can't change the class name in this field but you can change the contents of the Apex class. The class name can be changed in major upgrades. This field is available in API version 25.0 and later.</td>
</tr>
<tr>
<td>version</td>
<td>string</td>
<td>Required. The version of the component type.</td>
</tr>
</tbody>
</table>

### PackageTypeMembers

Use to specify the name and type of components to be retrieved in a package.
### Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character \* (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**SEE ALSO:**
- Sample package.xml Manifest Files

### PathAssistant

Represents Path records. This type extends the Metadata metadata type and inherits its fullName field.

Note the following when working with PathAssistant:

- Only one path can be created per record type for each object, including __Master__ record type.
- Rich text guidance information cannot be retrieved or deployed from or to translation workbench.
- The preference does not need to be on to retrieve or deploy PathAssistant.

### File Suffix and Directory Location

PathAssistant components have the suffix .pathAssistant and are stored in the pathAssistants folder.

### Version

PathAssistant components are available in API version 34.0 and later.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Indicates whether the path is active (true) or not (false).</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>entityName</td>
<td>string</td>
<td>Required. The entity name. This is hard coded for Opportunity, Lead, and Quote. For a custom object, this field must be specified and should be the name of the custom object. This field is not updateable.</td>
</tr>
<tr>
<td>fieldName</td>
<td>string</td>
<td>Required. The field name. This is hard coded for StageName and Status. For a custom object, this field must be specified and should be the name of the picklist field that determines the steps in the path. This field is not updateable.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The master label of the path.</td>
</tr>
<tr>
<td>pathAssistantSteps</td>
<td>PathAssistantStep[]</td>
<td>List of all the steps that have been configured with fields and guidance information. Note that a missing step in the .xml file means it has not been configured, not that it doesn’t exist.</td>
</tr>
<tr>
<td>recordTypeName</td>
<td>string</td>
<td>Required. The name of the record type associated with the path. This field is not updateable.</td>
</tr>
</tbody>
</table>

**PathAssistantStep**

Represents the steps or stages in a Path.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldNames</td>
<td>string</td>
<td>All the fields in entityName that will display in this step.</td>
</tr>
<tr>
<td>info</td>
<td>string</td>
<td>The guidance information displayed in this step.</td>
</tr>
<tr>
<td>picklistValueName</td>
<td>string</td>
<td>Required. The picklist value associated with the step.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example of a PathAssistant component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PathAssistant xmlns="http://soap.sforce.com/2006/04/metadata">
  <active>true</active>
  <entityName>Opportunity</entityName>
  <fieldName>StageName</fieldName>
  <masterLabel>Test Path</masterLabel>
  <pathAssistantSteps>
    <fieldNames>Amount</fieldNames>
    <fieldNames>CloseDate</fieldNames>
    <info>Some Text</info>
    <picklistValueName>Id. Decision Makers</picklistValueName>
  </pathAssistantSteps>
  <pathAssistantSteps>
    <fieldNames>Amount</fieldNames>
    <fieldNames>CloseDate</fieldNames>
    <info>Some Text</info>
  </pathAssistantSteps>
</PathAssistant>
```
The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Opportunity.Test_Busines_Process</members>
    <name>BusinessProcess</name>
  </types>
  <types>
    <members>Opportunity.StageName</members>
    <members>Lead.LeadSource</members>
    <members>Opportunity.Type</members>
    <name>CustomField</name>
  </types>
  <types>
    <members>Test_Path</members>
    <name>PathAssistant</name>
  </types>
  <types>
    <members>Opportunity.Test_Record_Type</members>
    <name>RecordType</name>
  </types>
  <types>
    <members>PathAssistant</members>
    <name>Settings</name>
  </types>
  <version>46.0</version>
</Package>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](https://developer.salesforce.com/docs/atlas.en-us.meta.meta.meta/).

### PermissionSet

Represents a set of permissions that’s used to grant additional access to one or more users without changing their profile or reassigning profiles. You can use permission sets to grant access but not to deny access.

This type extends the `Metadata` metadata type and inherits its `fullName` field.

**Important:** In API version 40.0 and later, when you retrieve or deploy permission set metadata, all content exposed in Metadata API for the permission sets is included. The metadata includes Apex associated with the permission set, CRUD, and so on.

In API version 39.0 and earlier, retrieving or deploying permission set metadata returns only app and system permissions assigned to the permission set. Junction metadata (such as Apex, CRUD) are included only if the metadata for the related component is also included in the package definition.
In API version 29.0 and later, you can retrieve and deploy access settings for the following managed components in profiles and permission sets:

- Apex classes
- Apps
- Custom field permissions
- Custom object permissions
- Custom tab settings
- External data sources
- Record types
- Visualforce pages

For more information, see Managed Component Access in Sample package.xml Manifest Files on page 23.

**Declarative Metadata File Suffix and Directory Location**

Permission sets are stored in the `permissionsets` directory. The file name matches the permission set API name and the extension is `.permissionset`. For example, a permission set with the name `User_Management_Perms` is stored in `permissionsets/User_Management_Perms.permissionset`.

**Version**

Permission sets are available in API version 22.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationVisibilities</td>
<td>PermissionSetApplicationVisibility[]</td>
<td>Indicates which apps are visible to users assigned to this permission set. Available in API version 29.0 and later. In API version 29.0, this field supports custom apps only. In API version 30.0 and later, this field supports both standard and custom apps.</td>
</tr>
<tr>
<td>classAccesses</td>
<td>PermissionSetApexClassAccess[]</td>
<td>Indicates which top-level Apex classes have methods that users assigned to this permission set can execute. Available in API version 23.0 and later.</td>
</tr>
<tr>
<td>customPermissions</td>
<td>PermissionSetCustomPermissions[]</td>
<td>Indicates which custom permissions are available to users assigned to this permission set. Available in API version 31.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The permission set description. Limit: 255 characters.</td>
</tr>
<tr>
<td>externalDataSourceAccess</td>
<td>PermissionSetExternalDataSourceAccess[]</td>
<td>Indicates which data sources with identity type of Per User are available to users assigned to this permission set. Available in API version 27.0 and later.</td>
</tr>
<tr>
<td>fieldPermissions</td>
<td>PermissionSetFieldPermissions[]</td>
<td>Indicates which fields are accessible to a user assigned to this permission set, and the kind of access available.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>hasActivationRequired</td>
<td>boolean</td>
<td>Indicates whether the permission set requires an associated active session (true) or not (false). Available in API version 37.0 and later.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The permission set label. Limit: 80 characters.</td>
</tr>
<tr>
<td>license</td>
<td>string</td>
<td>Either the related permission set license or the user license associated with this permission set. Available in API version 38.0 and later. Use this field instead of userLicense, which is deprecated and only available up to API Version 37.0.</td>
</tr>
<tr>
<td>objectPermissions</td>
<td>PermissionSetObjectPermissions[]</td>
<td>Indicates the objects that are accessible to a user assigned to this permission set, and the kind of access available (create, read, edit, delete, and so on). Available in API version 23.0 and later.</td>
</tr>
<tr>
<td>pageAccesses</td>
<td>PermissionSetApexPageAccess[]</td>
<td>Indicates which Visualforce pages that users assigned to this permission set can execute. Available in API version 23.0 and later.</td>
</tr>
<tr>
<td>recordTypeVisibilities</td>
<td>PermissionSetRecordTypeVisibility[]</td>
<td>Indicates which record types are visible to users assigned to this permission set. Available in API version 29.0 and later. This field is never retrieved or deployed for inactive record types.</td>
</tr>
<tr>
<td>tabSettings</td>
<td>PermissionSetTabSetting[]</td>
<td>Indicates the tab visibility settings for this permission set. Available in API version 26.0 and later.</td>
</tr>
<tr>
<td>userLicense</td>
<td>string</td>
<td>Deprecated. The user license for the permission set. A user license determines the baseline of features that the user can access. Every user must have exactly one user license. Available up to API version 37.0. In API version 38.0 and later, use license.</td>
</tr>
<tr>
<td>userPermissions</td>
<td>PermissionSetUserPermission[]</td>
<td>Specifies an app or system permission (such as “API Enabled”) and whether it’s enabled for this permission set. In API version 28.0 and earlier, this field retrieves all user permissions, enabled or disabled. In API version 29.0 and later, this field retrieves only enabled user permissions.</td>
</tr>
</tbody>
</table>

### PermissionSetApplicationVisibility

PermissionSetApplicationVisibility determines whether an app is visible to a user assigned to this permission set.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>string</td>
<td>Required. The app name.</td>
</tr>
<tr>
<td>visible</td>
<td>boolean</td>
<td>Required. Indicates whether this app is visible to users assigned to this permission set (true) or not (false).</td>
</tr>
</tbody>
</table>

**PermissionSetApexClassAccess**

PermissionSetApexClassAccess represents the Apex class access for users assigned to a permission set.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apexClass</td>
<td>string</td>
<td>Required. The Apex class name.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether users assigned to this permission set can execute methods in the top-level class (true) or not (false).</td>
</tr>
</tbody>
</table>

**PermissionSetCustomPermissions**

PermissionSetCustomPermissions represents the custom permissions access for users assigned to a permission set. Only enabled custom permissions are retrieved.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether the custom permission is enabled (true) or not (false).</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The custom permission name.</td>
</tr>
</tbody>
</table>

**PermissionSetExternalDataSourceAccess**

PermissionSetExternalDataSourceAccess represents the data source access for users with identity type of Per User. Available in API version 27.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether the data source is enabled (true) or not (false).</td>
</tr>
<tr>
<td>externalDataSource</td>
<td>string</td>
<td>The name of the external data source.</td>
</tr>
</tbody>
</table>

**PermissionSetFieldPermissions**

PermissionSetFieldPermissions represents the field permissions for users assigned to a permission set. In API version 30.0 and later, permissions for required fields can’t be retrieved or deployed.
**Note:** As of API version 38.0, you can change field permissions to make a field editable using the Metadata API for fields that you can't change through the user interface. For example, you can deploy `Asset.ProductCode` as an editable field even though you can't through the user interface.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>editable</td>
<td>boolean</td>
<td>Required. Indicates whether the field can be edited by the users assigned to this permission set (<code>true</code>) or not (<code>false</code>).</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>Required. The API name of the field (such as <code>Warehouse__c.Description__c</code>).</td>
</tr>
<tr>
<td>readable</td>
<td>boolean</td>
<td>Indicates whether the field can be read by the users assigned to this permission set (<code>true</code>) or not (<code>false</code>).</td>
</tr>
</tbody>
</table>

**PermissionSetObjectPermissions**

PermissionSetObjectPermissions represents the object permissions for a permission set. Use one of these elements for each permission.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowCreate</td>
<td>boolean</td>
<td>Required. Indicates whether the object referenced by the <code>object</code> field can be created by the users assigned to this permission set (<code>true</code>) or not (<code>false</code>).</td>
</tr>
<tr>
<td>allowDelete</td>
<td>boolean</td>
<td>Required. Indicates whether the object referenced by the <code>object</code> field can be deleted by the users assigned to this permission set (<code>true</code>) or not (<code>false</code>).</td>
</tr>
<tr>
<td>allowEdit</td>
<td>boolean</td>
<td>Required. Indicates whether the object referenced by the <code>object</code> field can be edited by the users assigned to this permission set (<code>true</code>) or not (<code>false</code>).</td>
</tr>
<tr>
<td>allowRead</td>
<td>boolean</td>
<td>Required. Indicates whether the object referenced by the <code>object</code> field can be viewed by the users assigned to this permission set (<code>true</code>) or not (<code>false</code>).</td>
</tr>
<tr>
<td>modifyAllRecords</td>
<td>boolean</td>
<td>Required. Indicates whether the object referenced by the <code>object</code> field can be viewed, edited, or deleted by the users assigned to this permission set (<code>true</code>) or not (<code>false</code>), regardless of the sharing settings for the object. This includes private records (records with no parent object). This is similar to the “Modify All Data” user permission but limited to the individual object level.</td>
</tr>
<tr>
<td>object</td>
<td>string</td>
<td>Required. The API name of the object (such as <code>Warehouse__c</code>).</td>
</tr>
<tr>
<td>viewAllRecords</td>
<td>boolean</td>
<td>Required. Indicates whether the object referenced by the <code>object</code> field can be viewed by the users assigned to this permission set (<code>true</code>) or not (<code>false</code>), regardless of the sharing settings for the object. This includes private records (records with no parent object). The <code>viewAllRecords</code> field is similar to the “View All Data” user permission but limited to the individual object level.</td>
</tr>
</tbody>
</table>
PermissionSetApexPageAccess

PermissionSetApexPageAccess represents the Visualforce page access for users assigned to a permission set.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apexPage</td>
<td>string</td>
<td>Required. The Visualforce page name.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether users assigned to this permission set can execute the Visualforce page (true) or not (false).</td>
</tr>
</tbody>
</table>

PermissionSetRecordTypeVisibility

PermissionSetRecordTypeVisibility represents the visibility of record types for this permission set.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recordType</td>
<td>string</td>
<td>Required. The record type name, for example Account.MyRecordType.</td>
</tr>
<tr>
<td>visible</td>
<td>boolean</td>
<td>Required. Indicates whether the record type is visible to users assigned to this permission set (true) or not (false).</td>
</tr>
</tbody>
</table>

PermissionSetTabSetting

PermissionSetTabSetting represents the tab settings for a permission set.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tab</td>
<td>string</td>
<td>Required. The tab name.</td>
</tr>
</tbody>
</table>
| visibility | PermissionSetTabVisibility (enumeration of type string) | Required. Indicates the visibility settings for the tab. Valid values are:  
  - Available—The tab is available on the All Tabs page. Individual users can customize their display to make the tab visible in any app.  
  - None—The tab isn’t available on the All Tabs page or visible in any apps.  
  - Visible—The tab is available on the All Tabs page and appears in the visible tabs for its associated app. Individual users can customize their display to hide the tab or make it visible in other apps. |

PermissionSetUserPermission

PermissionSetUserPermission represents an app or system permission for a permission set. Use one of these elements for each permission.
Declarative Metadata Sample Definition

When adding or changing a permission set, you don’t need to include all permissions—you only need to include the permissions you’re adding or changing.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PermissionSet xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>Grants all rights needed for an HR administrator to manage employees.</description>
  <label>HR Administration</label>
  <userLicense>Salesforce</userLicense>
  <applicationVisibilities>
    <application>JobApps__Recruiting</application>
    <visible>true</visible>
  </applicationVisibilities>
  <userPermissions>
    <enabled>true</enabled>
    <name>APIEnabled</name>
  </userPermissions>
  <objectPermissions>
    <allowCreate>true</allowCreate>
    <allowDelete>true</allowDelete>
    <allowEdit>true</allowEdit>
    <allowRead>true</allowRead>
    <viewAllRecords>true</viewAllRecords>
    <modifyAllRecords>true</modifyAllRecords>
    <object>Job_Request__c</object>
  </objectPermissions>
  <fieldPermissions>
    <editable>true</editable>
    <field>Job_Request__c.Salary__c</field>
    <readable>true</readable>
  </fieldPermissions>
  <pageAccesses>
    <apexPage>Job_Request_Web_Form</apexPage>
    <enabled>true</enabled>
  </pageAccesses>
  <classAccesses>
    <apexClass>Send_Email_Confirmation</apexClass>
    <enabled>true</enabled>
  </classAccesses>
  <tabSettings>
    <tab>Job_Request__c</tab>
    <visibility>Available</visibility>
  </tabSettings>
</PermissionSet>
```
The following is an example package.xml manifest used to retrieve the PermissionSet metadata for an organization. When you retrieve permission sets, also retrieve the related components with assigned permissions. For example, to retrieve objectPermissions and fieldPermissions for a custom object, you must also retrieve the CustomObject component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Job_Request__c</members>
    <name>CustomTab</name>
  </types>
  <types>
    <members>Job_Request__c</members>
    <name>CustomObject</name>
  </types>
  <types>
    <members>JobApps__Recruiting</members>
    <name>CustomApplication</name>
  </types>
  <types>
    <members>Recruiting.DevManager</members>
    <name>RecordType</name>
  </types>
  <types>
    <members>*</members>
    <name>PermissionSet</name>
  </types>
  <version>46.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**PermissionSetGroup (Pilot)**

Represents a group of permission sets and the permissions within them. Use permission set groups to organize permissions based on job functions or tasks. Then, you can package the groups as needed. This type extends the Metadata metadata type and inherits its fullName field.

**Declarative Metadata File Suffix and Directory Location**

Permission set groups are stored in the permissionsetgroups directory. The file name matches the permission set API name and the extension is .permissionsetgroup. For example, a permission set group with the name Finance_Mgmt_PermSetGroup is stored in permissionsetgroups/Finance_Mgmt_PermSetGroup.permissionsetgroup.
Version

Permission set groups are available in API version 45.0 and later.

Note: We provide this feature to selected customers through a pilot program that requires agreement to specific terms and conditions. To be nominated to participate in the program, contact Salesforce. Because pilot programs are subject to change, we can’t guarantee acceptance. This pilot feature isn’t generally available, as referenced in this document or in press releases or public statements. We can’t guarantee general availability within any particular time frame or at all. Make your purchase decisions only on the basis of generally available features.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The permission set group description provided by the permission set creator.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The permission set group label.</td>
</tr>
<tr>
<td>mutingPermissionSets</td>
<td>string</td>
<td>A permission set containing permissions to disable in the permission set group. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>permissionSets</td>
<td>string</td>
<td>A permission set or permission sets included in the permission set group.</td>
</tr>
<tr>
<td>status</td>
<td>string</td>
<td>Indicates permission set group recalculation status. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Updated—The group is current.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Outdated—The group requires recalculation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Updating—The group is in recalculation mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Failed—The group recalculation failed.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

When adding a permission set group, you can do something like this. Individual permissions are included in the permission set referenced, not in the permission set group.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PermissionSetGroup xmlns="http://soap.sforce.com/2006/04/metadata">
  <fullName>Finance_Mgmt_PermSetGroup</fullName>
  <description>Finance_Mgmt_PermSetGroup desc</description>
  <label>Finance_Mgmt_PermSetGroup</label>
  <permissionSets>Billing_PS</permissionSets>
</PermissionSetGroup>
```

The permission set Billing_PS contains the individual permissions included in Finance_Mgmt_PermSetGroup.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PermissionSet xmlns="http://soap.sforce.com/2006/04/metadata">
</PermissionSet>
```
This example package.xml manifest retrieves the PermissionSetGroup metadata for an org. When you retrieve permission set groups, also retrieve the related components. For example, to retrieve PermissionSetGroup, you must also retrieve PermissionSet.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Billing_PS</members>
    <name>PermissionSet</name>
  </types>
  <version>45.0</version>
</Package>
```

**PlatformCachePartition**

Represents a partition in the Platform Cache. This type extends the `Metadata` metadata type and inherits its `fullName` field.

**File Suffix and Directory Location**

PlatformCachePartition components have the suffix `.cachePartition` and are stored in the `cachePartitions` folder.

**Version**

PlatformCachePartition components are available in API version 35.0 and later.

**Special Access Rules**

The “Author Apex” permission is required to deploy and retrieve PlatformCachePartition components.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Describes the cache partition.</td>
</tr>
<tr>
<td>isDefaultPartition</td>
<td>boolean</td>
<td>Required. Indicates whether this cache partition is the default partition in your organization (true) or not (false).</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The label of the cache partition that appears in the Salesforce user interface.</td>
</tr>
</tbody>
</table>

platformCachePartitionTypes  PlatformCachePartitionType[] An array of cache types that the partition can store.

PlatformCachePartitionType

Contains information about a partition type, including its minimum and allocated capacity.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allocatedCapacity</td>
<td>int</td>
<td>Required. The total storage capacity, in MB, that is allocated for the cache type, including free, purchased, and trial cache. Purchased capacity includes organization-wide cache, which can be used in any partition, and namespace-specific cache, which can be used only in partitions associated with a namespace.</td>
</tr>
<tr>
<td>allocatedPurchasedCapacity</td>
<td>int</td>
<td>Required. The amount of namespace-specific purchased storage capacity, in MB, that is allocated for the cache type.</td>
</tr>
<tr>
<td>allocatedTrialCapacity</td>
<td>int</td>
<td>Required. The amount of trial cache space, in MB, that is allocated for the cache type.</td>
</tr>
</tbody>
</table>
| cacheType | PlatformCacheType (enumeration of type string) | The type of cache. Valid values are:  
  - Session—Session cache  
  - Organization—Org cache |

Declarative Metadata Sample Definition

The following is an example of a PlatformCachePartition component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PlatformCachePartition xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>Custom partition and marked as default.</description>
  <isDefaultPartition>true</isDefaultPartition>
  <masterLabel>myPartition</masterLabel>
  <platformCachePartitionTypes>
    <allocatedCapacity>10</allocatedCapacity>
    <allocatedPurchasedCapacity>5</allocatedPurchasedCapacity>
    <cacheType>Session</cacheType>
  </platformCachePartitionTypes>
</PlatformCachePartition>
```
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>myPartition</members>
    <name>PlatformCachePartition</name>
  </types>
  <version>46.0</version>
</Package>
```

If a namespace is defined in your organization, add the namespace prefix to your partition name. For example:

```xml
<members>Namespace.myPartition</members>
```

To retrieve all cache partitions from your organization, use the wildcard character (*) as follows.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>PlatformCachePartition</name>
  </types>
  <version>46.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**PlatformEventChannel**

Represents a collection of entities selected for Change Data Capture notifications for the standard ChangeEvents channel or a custom channel. The standard channel corresponds to the entity selection in the Change Data Capture page in Setup. A custom channel is a channel that you define for a group of entities using the PlatformEventChannel metadata type. Change Data Capture sends notifications for created, updated, deleted, and undeleted records only for the selected entities. This type extends the Metadata metadata type and inherits its fullName field.

**File Suffix and Directory Location**

PlatformEventChannel components have the suffix .platformEventChannel and are stored in the platformEventChannels folder.
Version

PlatformEventChannel components are available in API version 45.0 and later.

Special Access Rules

You must have the Customize Application permission to deploy and retrieve this type.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channelMembers</td>
<td>PlatformEventChannel SelectedEntity[]</td>
<td>A list of event names of entities, including standard and custom objects, selected for Change Data Capture notifications.</td>
</tr>
<tr>
<td>channelType</td>
<td>PlatformEventChannel Type (enumeration of type string)</td>
<td>Required. The channel type. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• data—Change Data Capture channel corresponding to the selected entities.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• event—Reserved for future use.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The channel label.</td>
</tr>
</tbody>
</table>

PlatformEventChannelSelectedEntity

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selectedEntity</td>
<td>string</td>
<td>Required. The event name of an entity selected for Change Data Capture notifications. For example, for the Account standard object, the name is AccountChangeEvent, or for a custom object MyObject__c, the name is MyObject__ChangeEvent.</td>
</tr>
</tbody>
</table>

Usage

The createMetadata() and deleteMetadata() calls aren't supported with the PlatformEventChannel metadata type. Also, you can't delete all Change Data Capture selections by deploying with destructiveChanges.xml. To update Change Data Capture selections, deploy an updated PlatformEventChannel definition or use the updateMetadata() call.

Declarative Metadata Sample Definition for the Standard Channel

This example of a PlatformEventChannel component for the standard ChangeEvents channel contains the event names of three selected entities: Account, Contact, and the Employee__c custom object.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PlatformEventChannel xmlns="http://soap.sforce.com/2006/04/metadata">
  <channelMembers>
    <selectedEntity>AccountChangeEvent</selectedEntity>
  </channelMembers>
</PlatformEventChannel>
```
Declarative Metadata Sample Definition for a Custom Channel

This example of a PlatformEventChannel component for a custom channel called SalesEvents contains the event names of three related sales objects: Account, Contact, and Order.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>SalesEvents__chn</members>
    <name>PlatformEventChannel</name>
  </types>
  <version>46.0</version>
</Package>
```

This package.xml references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>ChangeEvents</members>
    <name>PlatformEventChannel</name>
  </types>
  <version>46.0</version>
</Package>
```
**Wildcard Support in the Manifest File**

To deploy or retrieve all custom channels, specify the wildcard character \* (asterisk) in the `<members>` field. The wildcard character isn’t supported for the standard channel.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>PlatformEventChannel</name>
  </types>
  <version>46.0</version>
</Package>
```

**Portal**

The Portal metadata type represents a partner portal or Customer Portal. It extends `Metadata` and inherits its `fullName` field. To use this metadata type, you must have a partner portal or Customer Portal enabled for your organization. For more information, see “Partner Portal Overview” and “Enabling Your Customer Portal” in the Salesforce online help.

**Declarative Metadata File Suffix and Directory Location**

Lightning Platform Portal components are stored in the `portals` directory of the corresponding package directory. The file name matches the portal name, and the extension is `.portal`.

**Version**

Lightning Platform Portal components are available in API version 15.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Denotes whether this portal is active.</td>
</tr>
<tr>
<td>admin</td>
<td>string</td>
<td>The full name of the user designated to administer the portal.</td>
</tr>
<tr>
<td>defaultLanguage</td>
<td>string</td>
<td>The default language for HTML messages for the portal. Use the abbreviation for the language, for example, <code>en_US</code> for United States English.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The portal description.</td>
</tr>
<tr>
<td>emailSenderAddress</td>
<td>string</td>
<td>Required. The email address used when sending emails using templates configured from the portal (for example, for resetting the password).</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>emailSenderName</td>
<td>string</td>
<td>Required. The name to display when sending emails using templates configured from the portal (for example, for resetting the password).</td>
</tr>
<tr>
<td>enableSelfCloseCase</td>
<td>boolean</td>
<td>For the Customer Portal, allows portal users to close their own cases.</td>
</tr>
<tr>
<td>footerDocument</td>
<td>string</td>
<td>The file to be used as the footer for this portal.</td>
</tr>
<tr>
<td>forgotPassTemplate</td>
<td>string</td>
<td>The email template to use when a user clicks the Forgot Password link. Lightning email templates aren't packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The name of the portal. Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See createMetadata() to see an example of this field specified for a call.</td>
</tr>
<tr>
<td>headerDocument</td>
<td>string</td>
<td>The file to be used as the header for this portal.</td>
</tr>
<tr>
<td>isSelfRegistrationActivated</td>
<td>boolean</td>
<td>Determines whether self-registration is active or not for this portal.</td>
</tr>
<tr>
<td>loginHeaderDocument</td>
<td>string</td>
<td>The file to be used as the header for this portal's login page.</td>
</tr>
<tr>
<td>logoDocument</td>
<td>string</td>
<td>The file to be used as the logo for this portal.</td>
</tr>
<tr>
<td>logoutUrl</td>
<td>string</td>
<td>The URL that the user should be redirected to on logout.</td>
</tr>
<tr>
<td>newCommentTemplate</td>
<td>string</td>
<td>The email template to be used for auto-notifications on new case comments.</td>
</tr>
<tr>
<td>newPassTemplate</td>
<td>string</td>
<td>The email template to be used for auto-notifications on password reset. Lightning email templates aren't packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>newUserTemplate</td>
<td>string</td>
<td>The email template to be used for auto-notifications on new user creation. Lightning email templates aren't packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>ownerNotifyTemplate</td>
<td>string</td>
<td>The email template to be used for auto-notifications on owner change. Lightning email templates aren't packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>selfRegNewUserUrl</td>
<td>string</td>
<td>The URL of the self-registration page.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>selfRegUserDefaultProfile</td>
<td>string</td>
<td>The default profile for self-registered users.</td>
</tr>
<tr>
<td>selfRegUserDefaultRole</td>
<td>PortalRoles (enumeration of type string)</td>
<td>The default role for self-registered users. The valid values are: Executive, Manager, User, PersonAccount</td>
</tr>
<tr>
<td>selfRegUserTemplate</td>
<td>string</td>
<td>The email template to be used for auto-notifications on self-registration. Lightning email templates aren't packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>showActionConfirmation</td>
<td>boolean</td>
<td>Determines whether or not confirmation messages are displayed for actions in the portal.</td>
</tr>
<tr>
<td>stylesheetDocument</td>
<td>string</td>
<td>The Document object to be used as the CSS stylesheet for this portal.</td>
</tr>
<tr>
<td>type</td>
<td>PortalType (enumeration of type string)</td>
<td>Required. The type for this portal. The valid values are: CustomerSuccess, Partner</td>
</tr>
</tbody>
</table>

## Declarative Metadata Sample Definition

A sample XML definition of a portal is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Portal xmlns="http://soap.sforce.com/2006/04/metadata" >
  <active>true</active>
  <description>Customer Portal</description>
  <emailSenderName>rguest@albany.com</emailSenderName>
  <enableSelfCloseCase>false</enableSelfCloseCase>
  <forgotPassTemplate>unfiled$public/ChangePwdEmail</forgotPassTemplate>
  <isSelfRegistrationActivated>false</isSelfRegistrationActivated>
  <newPassTemplate>unfiled$public/ChangePwdEmail</newPassTemplate>
  <newUserTemplate>unfiled$public/NewUserEmail</newUserTemplate>
  <selfRegUserTemplate>unfiled$public/SelfRegUserEmail</selfRegUserTemplate>
  <showActionConfirmation>false</showActionConfirmation>
  <type>CustomerSuccess</type>
</Portal>
```
**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**SEE ALSO:**
- CustomSite

**PostTemplate**

Represents the metadata associated with an approval post template for Approvals in Chatter. With approval post templates, you can customize the information included in approval request posts that appear in Chatter feeds. This type extends the Metadata metadata type and inherits its fullName field.

**Note:** Review Chatter Post Templates for Approval Requests in the Salesforce Help before you create a post template.

**File Suffix and Directory Location**

PostTemplate components have the suffix `.postTemplate` and are stored in the `postTemplates` folder.

**Version**

PostTemplate components are available in API version 29.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>boolean</td>
<td>Required. Specifies whether this is the default post template for the given object. When set to true, this post template is used by approval processes that are associated with the same object and don’t specify a post template. When an object has no default post template, each of its approval processes uses the system default post template, unless the approval process specifies its own post template.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Optional description of the post template.</td>
</tr>
<tr>
<td>fields</td>
<td>string[]</td>
<td>Required. An array of up to four fields to include in approval request posts. If the approval object is a detail object in a master-detail relationship, Owner isn’t available for approval page layouts or approval post templates.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Name of the post template. This non-unique label is different from the unique name of the post template.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example of a PostTemplate component:

```xml
<PostTemplate xmlns="http://soap.sforce.com/2006/04/metadata">
  <default>false</default>
  <fields>NumberOfEmployees</fields>
  <fields>NumberofLocations__c</fields>
  <fields>PartnerAccount</fields>
  <fields>LeadCustomFieldNumber__c</fields>
  <label>My Lead Post Template</label>
</PostTemplate>
```

The following is an example package manifest that references the previous PostTemplate component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Lead.leadtemplate</members>
    <name>PostTemplate</name>
  </types>
  <version>29.0</version>
</Package>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

PresenceDeclineReason

Represents an Omni-Channel decline reason that agents can select when declining work requests. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

PresenceDeclineReason components have the suffix .presenceDeclineReason and are stored in the presenceDeclineReasons folder.

Version

PresenceDeclineReason components are available in API version 44.0 and later.

Special Access Rules

This type is available only if Omni-Channel is enabled in your org.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>The master label for the decline reason.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a PresenceDeclineReason component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PresenceDeclineReason xmlns="http://soap.sforce.com/2006/04/metadata">
    <label>Incorrect queue</label>
</PresenceDeclineReason>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>*</members>
        <name>PresenceDeclineReason</name>
    </types>
    <version>44.0</version>
</Package>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

PresenceUserConfig

Represents a configuration that determines a presence user's settings. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

PresenceUserConfig components have the suffix .presenceUserConfig and are stored in the presenceUserConfigs folder.

Version

PresenceUserConfig components are available in API version 44.0 and later.

Special Access Rules

This type is available only if Omni-Channel is enabled in your org.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignments</td>
<td>PresenceConfigAssignments</td>
<td>Specifies how presence configurations are assigned to Omni-Channel users. Presence configurations can be assigned to sets of users or to sets of profiles.</td>
</tr>
<tr>
<td>capacity</td>
<td>int</td>
<td>Required. The maximum number of work assignments that can be pushed to an agent at a time.</td>
</tr>
<tr>
<td>declineReasons</td>
<td>string</td>
<td>Specifies the list of decline reasons that an agent can select when they decline a work.</td>
</tr>
<tr>
<td>enableAutoAccept</td>
<td>boolean</td>
<td>Indicates whether work items that are routed to agents are automatically accepted (true) or not (false). Available only if enableDecline is set to false.</td>
</tr>
<tr>
<td>enableDecline</td>
<td>boolean</td>
<td>Indicates whether agents can decline work items that are routed to them (true) or not (false). Available only if enableAutoAccept is set to false.</td>
</tr>
<tr>
<td>enableDeclineReason</td>
<td>boolean</td>
<td>Indicates whether agents can select a reason for declining work requests (true) or not (false). This can be selected only if decline reasons are enabled.</td>
</tr>
<tr>
<td>enableDisconnectSound</td>
<td>boolean</td>
<td>Indicates whether a sound is played when agents are disconnected from Omni-Channel (true) or not (false).</td>
</tr>
<tr>
<td>enableRequestSound</td>
<td>boolean</td>
<td>Indicates whether a sound plays with incoming work requests (true) or not (false). Set to true by default.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label of the presence configuration.</td>
</tr>
<tr>
<td>presenceStatusOnDecline</td>
<td>string</td>
<td>The presence status that’s automatically assigned to the agent when the agent declines a work item. Available only if enableDecline is set to true.</td>
</tr>
<tr>
<td>presenceStatusOnPushTimeout</td>
<td>string</td>
<td>The presence status that’s automatically assigned to the agent when the agent doesn’t respond to a work item before push timeout occurs.</td>
</tr>
</tbody>
</table>

PresenceConfigAssignments

Represents the assignments of an org’s profiles and users to a Presence configuration.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>profiles</td>
<td>PresenceConfigProfileAssignments</td>
<td>Specifies the profiles that are associated with a specific presence configuration.</td>
</tr>
<tr>
<td>users</td>
<td>PresenceConfigUserAssignments</td>
<td>Specifies the users that are associated with a specific presence configuration.</td>
</tr>
</tbody>
</table>
PresenceConfigProfileAssignments

Represents the profiles associated with a specific presence configuration.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>profile</td>
<td>string</td>
<td>Specifies the name of the profile associated with a specific presence configuration.</td>
</tr>
</tbody>
</table>

PresenceConfigUserAssignments

Represents the users associated with a specific presence configuration.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>string</td>
<td>Specifies the username of the user associated with a specific presence configuration.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a PresenceUserConfig component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PresenceUserConfig xmlns="http://soap.sforce.com/2006/04/metadata">
    <assignments>
        <profiles>
            <profile>standard</profile>
        </profiles>
        <users>
            <user>jdoe@example.com</user>
        </users>
    </assignments>
    <capacity>5</capacity>
    <declineReasons>Incorrect_queue</declineReasons>
    <enableAutoAccept>false</enableAutoAccept>
    <enableDecline>true</enableDecline>
    <enableDeclineReason>true</enableDeclineReason>
    <enableDisconnectSound>true</enableDisconnectSound>
    <enableRequestSound>true</enableRequestSound>
    <label>My presence configuration</label>
    <presenceStatusOnDecline>Away</presenceStatusOnDecline>
    <presenceStatusOnPushTimeout>Break</presenceStatusOnPushTimeout>
</PresenceUserConfig>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>*</members>
        <name>PresenceUserConfig</name>
    </types>
</Package>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Profile

Represents a user profile. A profile defines a user’s permission to perform different functions within Salesforce. This type extends the Metadata metadata type and inherits its fullName field.

In API version 29.0 and later, you can retrieve and deploy access settings for the following managed components in profiles and permission sets:

- Apex classes
- Apps
- Custom field permissions
- Custom object permissions
- Custom tab settings
- External data sources
- Record types
- Visualforce pages

For more information, see Managed Component Access in Sample package.xml Manifest Files on page 23.

Declarative Metadata File Suffix and Directory Location

The file suffix is .profile. There is one file for each profile, stored in the profiles folder in the corresponding package directory.

Version

Profiles are available in API version 10.0 and later.

Fields

The content of a profile returned by Metadata API depends on the content requested in the RetrieveRequest message. For example, profiles only include field-level security for fields included in custom objects returned in the same RetrieveRequest as the profiles.

Important: We designed Profile metadata deployment to overlay the existing Profile settings in a target org. For example, if you disable permissions for a profile, the newly disabled permission information isn’t exported. To force all Profile changes to deploy through metadata, including permission disablement, add code that explicitly indicates disabled permissions. For example, add the following code to the Profile metadata .xml file before deploying into a target org: <value>false</value>.
Note: As of API version 38.0, you can change field permissions to make a field editable using the Metadata API for fields that you can't change through the user interface. For example, you can deploy `Asset.ProductCode` as an editable field even though you can't through the user interface.

The profile definition contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationVisibilities</td>
<td>ProfileApplicationVisibility[]</td>
<td>Indicates which apps are visible to users assigned to this profile. In API version 29.0 and earlier, this field supports custom apps only. In API version 30.0 and later, this field supports both standard and custom apps.</td>
</tr>
<tr>
<td>categoryGroupVisibilities</td>
<td>ProfileCategoryGroupVisibility[]</td>
<td>Indicates which data category groups are visible to users assigned to this profile. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>classAccesses</td>
<td>ProfileApexClassAccess[]</td>
<td>Indicates which top-level Apex classes have methods that users assigned to this profile can execute.</td>
</tr>
<tr>
<td>custom</td>
<td>boolean</td>
<td>Indicates whether the profile is a custom (true) or standard (false) profile. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td>customPermissions</td>
<td>ProfileCustomPermissions[]</td>
<td>Indicates which custom permissions are available to users assigned to this profile. Available in API version 31.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The profile description. Limit: 255 characters. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td>externalDataSourceAccesses</td>
<td>ProfileExternalDataSourceAccess[]</td>
<td>Indicates which data sources with identity type of PerUser are available to users assigned to this profile. Available in API version 27.0 and later.</td>
</tr>
<tr>
<td>fieldLevelSecurities</td>
<td>ProfileFieldLevelSecurity[]</td>
<td>Indicates which fields are visible to a user assigned to this profile, and the kind of access available (editable or hidden). This field is available in API version 22.0 and earlier.</td>
</tr>
<tr>
<td>fieldPermissions</td>
<td>ProfileFieldLevelSecurity[]</td>
<td>Indicates which fields are visible to a user assigned to this profile, and the kind of access available (editable or readable). This field is available in API version 23.0 and later.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The name can only contain characters, letters, and the underscore (_) character, must start with a letter, and cannot end with an underscore or contain two consecutive underscore characters. Inherited from the Metadata component, this field is not defined in the WSDL for this component. It must be specified when creating, updating, or deleting. See</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>layoutAssignments</td>
<td>ProfileLayoutAssignments[]</td>
<td>Indicates which layout to use for this profile.</td>
</tr>
<tr>
<td>loginHours</td>
<td>ProfileLoginHours[]</td>
<td>Indicates the hours within which a user with this profile can log in. If not specified, the profile doesn't restrict a user's login hours. This field is available in API version 25.0 and later.</td>
</tr>
<tr>
<td>loginIpRanges</td>
<td>ProfileLoginIpRange[]</td>
<td>The list of IP address ranges from which users with a particular profile can log in. This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>objectPermissions</td>
<td>ProfileObjectPermissions[]</td>
<td>Indicates which objects are accessible to a user assigned to this profile, and the kind of access available (create, read, edit, delete, and so on). In API version 28.0 and later, this field is only retrieved when allowRead is true.</td>
</tr>
<tr>
<td>pageAccesses</td>
<td>ProfileApexPageAccess[]</td>
<td>Indicates which Visualforce pages that users assigned to this profile can execute.</td>
</tr>
<tr>
<td>profileActionOverrides</td>
<td>ProfileActionOverride[]</td>
<td>A list of the Lightning Experience Home page action overrides that are assigned to this profile. When a user logs in with a profile, a matching ProfileActionOverride assignment takes precedence over existing overrides for the Home tab specified in ActionOverride. This field is available in API versions 37.0 to 44.0.</td>
</tr>
<tr>
<td>recordTypeVisibilities</td>
<td>ProfileRecordTypeVisibility[]</td>
<td>Indicates the visibility of record types for users assigned to this profile. In API version 29.0 and later, this field is not retrieved or deployed for inactive record types.</td>
</tr>
<tr>
<td>tabVisibilities</td>
<td>ProfileTabVisibility[]</td>
<td>Indicates which record types are visible to a user assigned to this profile, and therefore which tabs within an app are visible.</td>
</tr>
<tr>
<td>userLicense</td>
<td>string</td>
<td>The User License for the profile. A user license determines the baseline of features that the user can access. Every user must have exactly one user license. This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>userPermissions</td>
<td>ProfileUserPermission[]</td>
<td>Specifies a user permission (such as “API Enabled”) and whether it's enabled for this profile. This field retrieves only enabled user permissions. Available in API version 29.0 and later.</td>
</tr>
</tbody>
</table>
ProfileActionOverride

ProfileActionOverride represents a user profile-based override of an ActionOverride on a standard Home tab in Lightning Experience.

Note:

- ProfileActionOverride can be defined only on Profile for API version 39.0 to 44.0. In API version 45.0 and later, ProfileActionOverride must be defined for CustomApplication instead. Beginning with API version 45.0, Home page assignments related to user profile must also have a corresponding app assignment because more granular Home page assignments are supported. As a result, ProfileActionOverride is defined for CustomApplication rather than Profile.
- ProfileActionOverride settings aren’t retrieved in the .profile file unless a Lightning page is referenced in the package.xml file.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>string</td>
<td>Required. The possible values are the same as the actions you can override: engage, clone, delete, edit, list, new, tab, view</td>
</tr>
<tr>
<td>content</td>
<td>string</td>
<td>Set this field if type is set to flexipage, lightningcomponent, scontrol, or visualforce. It refers to the name of the Lightning page, Lightning component, s-control, or Visualforce page to use as the override. To reference installed components, use this format: Component_namespace__Component_name.</td>
</tr>
<tr>
<td>formFactor</td>
<td>FormFactor (enumeration of type string)</td>
<td>The size of the page being overridden. The Large value represents the Lightning Experience desktop environment and is valid only for the flexipage and lightningcomponent types. The Small value represents the Salesforce app on a phone or tablet. The Medium value is reserved for future use. The null value (which is the same as specifying no value) represents Salesforce Classic.</td>
</tr>
<tr>
<td>pageOrObjectType</td>
<td>string</td>
<td>The name of the sObject type being overridden. Valid values are standard and custom. This value must be standard-home when actionName is tab.</td>
</tr>
<tr>
<td>recordType</td>
<td>string</td>
<td>The record type assigned to the ProfileActionOverride. If the PageOrObjectType is standard-home, this field is null.</td>
</tr>
</tbody>
</table>
### ProfileApplicationVisibility

ProfileApplicationVisibility determines whether an app is visible to a user assigned to this profile.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>string</td>
<td>Required. The name of the app.</td>
</tr>
<tr>
<td>default</td>
<td>boolean</td>
<td>Required. Indicates whether the app is the default app (true) or not (false). Only one app per profile can be set to true.</td>
</tr>
<tr>
<td>visible</td>
<td>boolean</td>
<td>Required. Indicates whether this app is visible to users assigned to this profile (true) or not (false).</td>
</tr>
</tbody>
</table>

### ProfileCategoryGroupVisibility

ProfileCategoryGroupVisibility determines whether a data category group is visible to a user assigned to this profile. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataCategories</td>
<td>string[]</td>
<td>Array of one or more data category names.</td>
</tr>
<tr>
<td>dataCategoryGroup</td>
<td>string</td>
<td>Required. The name of the data category group.</td>
</tr>
</tbody>
</table>
| visibility | CategoryGroupVisibility (enumeration of type string) | Required. Indicates the visibility of the data category. Valid values are:  
  - ALL  
  - CUSTOM  
  - NONE |

### ProfileApexClassAccess

ProfileApexClassAccess determines which top-level Apex classes have methods that users assigned to this profile can execute.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apexClass</td>
<td>string</td>
<td>Required. The Apex class name.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether users assigned to this profile can execute methods in the top-level class (true) or not (false).</td>
</tr>
</tbody>
</table>
**ProfileCustomPermissions**

ProfileCustomPermissions represents the custom permissions access for users assigned to a profile. Only enabled custom permissions are retrieved.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether the custom permission is enabled (true) or not (false).</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The custom permission name.</td>
</tr>
</tbody>
</table>

**ProfileExternalDataSourceAccess**

ProfileExternalDataSourceAccess represents the data source access for users with identity type of Per User. Available in API version 27.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether the data source is enabled (true) or not (false).</td>
</tr>
<tr>
<td>externalDataSource</td>
<td>string</td>
<td>The name of the external data source.</td>
</tr>
</tbody>
</table>

**ProfileFieldLevelSecurity**

ProfileFieldLevelSecurity represents the field level security for users assigned to a profile. In API version 30.0 and later, permissions for required fields can’t be retrieved or deployed.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>editable</td>
<td>boolean</td>
<td>Required. Indicates whether this field is editable (true) or not (false). In API version 30.0 and later, when deploying a new custom field, this field is false by default.</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>Required. Indicates the name of the field.</td>
</tr>
<tr>
<td>hidden</td>
<td>boolean</td>
<td>Indicates whether this field is hidden (true) or not (false). This field is available in API version 22.0 and earlier. For portal profiles, this field is set to true by default in API version 19.0 and later.</td>
</tr>
<tr>
<td>readable</td>
<td>boolean</td>
<td>Indicates whether this field is readable (true) or not (false). This field is available in API version 23.0 and later. It replaces the hidden field. In API version 30.0 and later, when deploying a new custom field, this field is false by default. For portal profiles, this field is set to false by default.</td>
</tr>
</tbody>
</table>
ProfileLayoutAssignments

ProfileLayoutAssignments determines which layout to use for a profile and a given entity.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layout</td>
<td>string</td>
<td>Required. Indicates the layout for this particular entity.</td>
</tr>
<tr>
<td>recordType</td>
<td>string</td>
<td>This field is optional. If the recordType of the record matches a layout assignment rule, it will use the specified layout.</td>
</tr>
</tbody>
</table>

ProfileLoginHours

ProfileLoginHours restricts the days and times within which users with a particular profile can log in.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>weekdayStart</td>
<td>string</td>
<td>Specifies the earliest time on that day that a user with this profile can log in. If a start time for a particular day is specified, an end time for that day must be specified as well. Start can’t be greater than end for a particular day.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Valid values for weekday: monday, tuesday, wednesday, thursday, friday, saturday, or sunday. For example, mondayStart indicates the beginning of the login period for Monday.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Valid values for Start: the number of minutes since midnight. Must be evenly divisible by 60 (full hours). For example, 300 is 5:00 a.m.</td>
</tr>
<tr>
<td>weekdayEnd</td>
<td>string</td>
<td>Specifies the time on that day by which a user with this profile must log out.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Valid values for weekday: monday, tuesday, wednesday, thursday, friday, saturday, or sunday. For example, mondayEnd indicates the close of the login period for Monday.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Valid values for End: the number of minutes since midnight. Must be evenly divisible by 60 (full hours). For example, 1020 is 5:00 p.m.</td>
</tr>
</tbody>
</table>

To delete login hour restrictions from a profile that previously had them, you must explicitly include an empty loginHours tag without any start or end times.

ProfileLoginIpRange

ProfileLoginIpRange IP defines an IP address range from which users with a particular profile can log in.
### Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Use this field to identify the purpose of the range, such as which part of a network corresponds to this range. This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>endAddress</td>
<td>string</td>
<td>Required. The end IP address for the range.</td>
</tr>
<tr>
<td>startAddress</td>
<td>string</td>
<td>Required. The start IP address for the range.</td>
</tr>
</tbody>
</table>

---

### ProfileObjectPermissions

ProfileObjectPermissions represents a user’s access to objects.

**Note:** In API version 18.0 and later, these permissions are disabled in new custom objects for any profiles in which “View All Data” or “Modify All Data” is disabled.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowCreate</td>
<td>boolean</td>
<td>Indicates whether the object referenced by the object field can be created by the users assigned to this profile (true) or not (false).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is named revokeCreate before version 14.0 and the logic is reversed. The field name change and the update from true to false and the reverse is automatically handled between versions and does not require any manual editing of existing XML component files.</td>
</tr>
<tr>
<td>allowDelete</td>
<td>boolean</td>
<td>Indicates whether the object referenced by the object field can be deleted by the users assigned to this profile (true) or not (false).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is named revokeDelete before version 14.0 and the logic is reversed. The field name change and the update from true to false and the reverse is automatically handled between versions and does not require any manual editing of existing XML component files.</td>
</tr>
<tr>
<td>allowEdit</td>
<td>boolean</td>
<td>Indicates whether the object referenced by the object field can be edited by the users assigned to this profile (true) or not (false).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is named revokeEdit before version 14.0 and the logic is reversed. The field name change and the update from true to false and the reverse is automatically handled between versions and does not require any manual editing of existing XML component files.</td>
</tr>
<tr>
<td>allowRead</td>
<td>boolean</td>
<td>Indicates whether the object referenced by the object field can be seen by the users assigned to this profile (true) or not (false).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is named revokeRead before version 14.0 and the logic is reversed. The field name change and the update from true to false and the reverse is automatically handled between versions and does not require any manual editing of existing XML component files.</td>
</tr>
<tr>
<td>modifyAllRecords</td>
<td>boolean</td>
<td>Indicates whether the object referenced by the object field can be read, edited, or deleted by the users assigned to this profile (true) or</td>
</tr>
</tbody>
</table>
**Profile**

ProfileApexPageAccess determines which Visualforce pages that users assigned to this profile can execute.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apexPage</td>
<td>string</td>
<td>Required. The Visualforce page name.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether users assigned to this profile can execute the Visualforce page (true) or not (false).</td>
</tr>
</tbody>
</table>

ProfileRecordTypeVisibility represents the visibility of record types for this profile. Record types let you offer different business processes, picklist values, and page layouts to different users.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>boolean</td>
<td>Required. Indicates whether the record type is the default for this pair of profile and object (true) or not (false). Only one default is allowed per object.</td>
</tr>
</tbody>
</table>
Metadata Types

Profile

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>personAccountDefault</td>
<td>boolean</td>
<td>Indicates whether the record type is the default person account record type for this pair of profile and object (true) or not (false). Only one person account record type default is allowed per object. This field is only relevant for record types for account or contact objects. For more information about person accounts, see “Person Accounts” in the Salesforce online help. Person accounts are not enabled by default in Salesforce. To request person accounts, contact Salesforce.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recordType</td>
<td>string</td>
<td>Required. The record type name, for example <code>Account.MyRecordType</code></td>
</tr>
<tr>
<td>visible</td>
<td>boolean</td>
<td>Required. Indicates whether this record type is visible to users assigned to this profile (true) or not (false).</td>
</tr>
</tbody>
</table>

ProfileTabVisibility

ProfileTabVisibility represents the visibility of tabs for this profile. For version 17.0 and later, ProfileTabVisibility supports visibility of tabs for standard objects. The manifest file must include the standard object corresponding to a standard tab to retrieve the tab visibility in a profile.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tab</td>
<td>string</td>
<td>Required. The name of the tab.</td>
</tr>
</tbody>
</table>
| visibility | TabVisibility (enumeration of type string) | Required. Indicates the visibility of the tab. Valid values are:  
- **DefaultOff**—The tab is available on the All Tabs page. Users can individually customize their display to make the tab visible in any app.  
- **DefaultOn**—The tab is available on the All Tabs page and appears in the visible tabs for its associated app. Users can individually customize their display to hide the tab or make it visible in other apps.  
- **Hidden**—The tab isn’t available on the All Tabs page or visible in any apps.  

**Note:** In version 36.0 and earlier, Hidden is returned only if visibility was set using the API. If it was set to Hidden from the profile in Salesforce, the API doesn’t return a visibility value. For version 37.0 and later, when tab visibility is set to hidden, the API returns Hidden, regardless of how the value was set.

ProfileUserPermission

ProfileUserPermission represents an app or system permission for a profile. Use one of these elements for each permission.
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required. Indicates whether the permission is enabled (true) or disabled (false).</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The permission name.</td>
</tr>
</tbody>
</table>

### Java Sample

The following sample uses picklists, profiles, record types, and a custom app:

```java
public void profileSample() {
    try {
        // Create an expense report record, tab and app...
        CustomObject expenseRecord = new CustomObject();
        expenseRecord.setFullName("ExpenseReport__c");
        expenseRecord.setLabel("Expense Report");
        expenseRecord.setPluralLabel("Expense Reports");
        expenseRecord.setDeploymentStatus(DeploymentStatus.Deployed);
        expenseRecord.setSharingModel(SharingModel.ReadWrite);

        CustomField nameField = new CustomField();
        nameField.setType(FieldType.AutoNumber);
        nameField.setLabel("Expense Report Number");
        nameField.setDisplayFormat("ER-{0000}");
        expenseRecord.setNameField(nameField);

        AsyncResult[] arsExpenseRecord =
            metadataConnection.create(new Metadata[] {expenseRecord});

        Picklist expenseStatus = new Picklist();
        PicklistValue unsubmitted = new PicklistValue();
        unsubmitted.setFullName("Unsubmitted");
        PicklistValue submitted = new PicklistValue();
        submitted.setFullName("Submitted");
        PicklistValue approved = new PicklistValue();
        approved.setFullName("Approved");
        PicklistValue rejected = new PicklistValue();
        rejected.setFullName("Rejected");
        expenseStatus.setPicklistValues(new PicklistValue[] {
            unsubmitted, submitted, approved, rejected
        });

        CustomField expenseStatusField = new CustomField();
        expenseStatusField.setFullName("ExpenseReport__c.ExpenseStatus__c");
        expenseStatusField.setLabel("Expense Report Status");
        expenseStatusField.setType(FieldType.Picklist);
        expenseStatusField.setPicklist(expenseStatus);
        AsyncResult[] arsStatusField =
            metadataConnection.create(new Metadata[]
```

---

667
CustomTab expenseTab = new CustomTab();
expenseTab.setFullName("ExpenseReport__c");
expenseTab.setMotif("Custom70: Handsaw");
expenseTab.setCustomObject(true);
AsyncResult[] arsTab =
    metadataConnection.create(new Metadata[] {expenseTab});

CustomApplication application = new CustomApplication();
application.setFullName("ExpenseForce");
application.setTab(new String[] {expenseTab.getFullName()});
AsyncResult[] arsApp =
    metadataConnection.create(new Metadata[] {application});

// Employees and managers have the same app visibility...
ProfileApplicationVisibility appVisibility =
    new ProfileApplicationVisibility();
appVisibility.setApplication("ExpenseForce");
appVisibility.setVisible(true);

Profile employee = new Profile();
employee.setFullName("Employee");
employee.setApplicationVisibilities(new ProfileApplicationVisibility[] {appVisibility});
AsyncResult[] arsProfileEmp =
    metadataConnection.create(new Metadata[] {employee});

Profile manager = new Profile();
manager.setFullName("Manager");
manager.setApplicationVisibilities(new ProfileApplicationVisibility[] {appVisibility});
AsyncResult[] arsProfileMgr =
    metadataConnection.create(new Metadata[] {manager});

// But employees and managers have different access to the state of the expense sheet
RecordType edit = new RecordType();
eidt.setFullName("ExpenseReport__c.Edit");
RecordTypePicklistValue editStatuses =
    new RecordTypePicklistValue();
editStatuses.setPicklist("ExpenseStatus__c");
editStatuses.setValues(new PicklistValue[] {unsubmitted, submitted});
edit.setPicklistValues(new RecordTypePicklistValue[] {editStatuses});
AsyncResult[] arsRecTypeEdit =
    metadataConnection.create(new Metadata[] {edit});

RecordType approve = new RecordType();
approve.setFullName("ExpenseReport__c.Approve");
RecordTypePicklistValue approveStatuses =
new RecordTypePicklistValue();
approveStatuses.setPicklist("ExpenseStatus__c");
approveStatuses.setValues(new PicklistValue[]{
approved, rejected});
approve.setPicklistValues(new RecordTypePicklistValue[]{
approveStatuses});
AsyncResult[] arsRecTypeApp =
    metadataConnection.create(new Metadata[]{approve});
} catch (ConnectionException ce) {
    ce.printStackTrace();
}

Declarative Metadata Sample Definition

The following is the definition of a profile in an organization with a custom app, custom object, record type, tab, and user permission:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Profile xmlns="http://soap.sforce.com/2006/04/metadata">
    <applicationVisibilities>
        <application>PubApps__Myriad_Publishing</application>
        <default>false</default>
        <visible>true</visible>
    </applicationVisibilities>
    <custom>true</custom>
    <objectPermissions>
        <object>TestWeblinks__c</object>
    </objectPermissions>
    <recordTypeVisibilities>
        <default>true</default>
        <recordType>TestWeblinks__c.My First Recordtype</recordType>
        <visible>true</visible>
    </recordTypeVisibilities>
    <tabVisibilities>
        <tab>Myriad Publications</tab>
        <visibility>DefaultOn</visibility>
    </tabVisibilities>
    <userPermissions>
        <enabled>true</enabled>
        <name>APIEnabled</name>
    </userPermissions>
</Profile>
```

Usage

When you use the `retrieve()` call to get information about profiles in your organization, the returned `.profile` files only include security settings for the other metadata types referenced in the retrieve request (except for user permissions, IP address ranges, and login hours, which are always retrieved). For example, the following `package.xml` file contains a `<types>` element that matches all
Metadata Types

custom objects, so the returned profiles contain object and field permissions for all custom objects in your organization, but do not include permissions for standard objects, such as Account, and standard fields.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>CustomObject</name>
  </types>
  <types>
    <members>*</members>
    <name>Profile</name>
  </types>
  <version>46.0</version>
</Package>
```

The wildcard "*" on CustomObject does not match standard objects and this helps to avoid making unintended, high-impact profile changes. If you create a few custom objects in a Developer Edition organization, retrieve() the information, and subsequently deploy() the custom objects to your production organization, the profile and field-level security for all your standard objects, such as Account, and standard fields are not overwritten unless you explicitly create separate types elements for the standard objects or fields.

Metadata API intentionally makes it difficult to include standard fields in retrieve() calls to prevent unexpected profile changes. However, you can still retrieve and deploy profile permissions for custom and standard fields in standard objects, such as Account.

The next package.xml file allows you to return profile permissions for Account standard and custom fields. Note how the standard Account object is defined in a types element by specifying it as a member of a CustomObject type.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Account</members>
    <name>CustomObject</name>
  </types>
  <types>
    <members>*</members>
    <name>Profile</name>
  </types>
  <version>46.0</version>
</Package>
```

The final package.xml file allows you to return profile permissions for the MyCustomField__c custom field in the Account object.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Account.MyCustomField__c</members>
    <name>CustomField</name>
  </types>
  <types>
    <members>*</members>
    <name>Profile</name>
  </types>
  <version>46.0</version>
</Package>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

ProfileActionOverride

Represents an override of an ActionOverride by a user profile. You can use it to override an ActionOverride on a standard Home tab or object record page in Lightning Experience. When a user logs in with a profile, a matching ProfileActionOverride assignment takes precedence over existing overrides for the Home tab or record page specified in ActionOverride. In API versions 39.0 to 44.0, you can access ProfileActionOverride by accessing its encompassing CustomApplication or Profile metadata types. In API version 45.0 and later, you can access ProfileActionOverride only by accessing its encompassing CustomApplication.

Note: ProfileActionOverrides aren’t supported in packaging. They are supported in change sets, but you have to add them manually.

File Suffix and Directory Location

Profile-based action overrides are defined as part of a custom application or profile.

Version

ProfileActionOverrides are available in API version 39.0 and later.

ProfileActionOverride can be defined on Profile or CustomApplication for API version 39.0 to 44.0. In API version 45.0 and later, ProfileActionOverride must be defined for CustomApplication instead. Beginning with API version 45.0, Home page assignments related to user profile must also have a corresponding app assignment because more granular Home page assignments are supported. As a result, ProfileActionOverride is defined for CustomApplication rather than Profile.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>string</td>
<td>The name of the action. The only valid values are Tab and View. If pageOrSobjectType is standard-home, this field must be Tab. The Tab action is supported only when ProfileActionOverride is being specified as part of a Profile in API version 39.0 to 44.0. In API version 45.0 and later, this action is supported only when ProfileActionOverride is being specified as part of a CustomApplication, pageOrSobjectType is standard-home, and this field is Tab. If pageOrSobjectType is record-home, this field must be View. The View action is supported only when ProfileActionOverride is being specified as part of a CustomApplication.</td>
</tr>
<tr>
<td>content</td>
<td>string</td>
<td>Read-only. Represents the name of the Lightning page being used as the override.</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
---|---|---
formFactor | FormFactor (enumeration of type string) | The size of the page being overridden. The Large value represents the Lightning Experience desktop environment.
pageOrSobjectType | string | The name of the page being overridden. The only valid values are record-home and standard-home. If the actionName is Tab, this field must be standard-home.
recordType | string | The record type associated with the override. If pageOrSobjectType is standard-home, this field must be null. This field is required when actionName is set to View.
type | ActionOverrideType (enumeration of type string) | Read-only. The type of action override. The only valid value is flexipage.

### Declarative Metadata Sample Definition

You can define a ProfileActionOverride like this.

```xml
<CustomApplication xmlns="http://soap.sforce.com/2006/04/metadata">
  <profileActionOverrides>
    <actionName>View</actionName>
    <content>CustomObjectFlexiPage</content>
    <formFactor>Large</formFactor>
    <pageOrSobjectType>TestObj__c</pageOrSobjectType>
    <type>Flexipage</type>
    <profile>standard</profile>
    <recordType>TestObj__c.TestRecordType</recordType>
  </profileActionOverrides>
  <defaultLandingTab>standard-home</defaultLandingTab>
  <formFactors>Large</formFactors>
  <label>My Custom App</label>
  <tab>standard-Account</tab>
  <tab>standard-Opportunity</tab>
  <uiType>Lightning</uiType>
  <navType>Standard</navType>
</CustomApplication>
```

Here is an example `package.xml`.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>MyCustomApp</members>
    <name>CustomApplication</name>
  </types>
  <version>39.0</version>
</Package>
```
Wildcard Support in the Manifest File

This metadata type doesn't support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

ProfilePasswordPolicy

Represents a profile's password policies. Profile password policies override org-wide password policies for that profile's users. Use ProfilePasswordPolicy to retrieve password policies for a given profile. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

ProfilePasswordPolicy components have the suffix .profilePasswordPolicy and are stored in the profilePasswordPolicies folder.

Version

ProfilePasswordPolicy components are available in API version 40.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>forgotPasswordRedirect</td>
<td>boolean</td>
<td>If true, reset password links in forgot password emails don't immediately expire the first time they're clicked. Instead, the links stay active until a user confirms the password reset request on an interstitial page. The default value is false. This field is available in API version 43.0 and later.</td>
</tr>
<tr>
<td>lockoutInterval</td>
<td>int</td>
<td>Required. The duration of the login lockout, in minutes. If users are locked out, they must wait until the lockout period expires. Valid values: 0, 15, 30, 60.</td>
</tr>
<tr>
<td>maxLoginAttempts</td>
<td>int</td>
<td>Required. The number of times a user can enter a wrong password before getting locked out. Valid values: 0, 3, 5, 10.</td>
</tr>
<tr>
<td>minimumPasswordLength</td>
<td>int</td>
<td>Required. Minimum number of characters required for a password. Valid values: 5–50.</td>
</tr>
<tr>
<td>minimumPasswordLifetime</td>
<td>boolean</td>
<td>If true, a user cannot change a password more than once in a 24-hour period.</td>
</tr>
<tr>
<td>obscure</td>
<td>boolean</td>
<td>If true, answers to security questions are hidden as the user types.</td>
</tr>
<tr>
<td>passwordComplexity</td>
<td>int</td>
<td>Required. Level of complexity required for the character types in a user's password.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If 0, the password can contain any type of character.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>passwordExpiration</td>
<td>int</td>
<td>Required. Number of days until user passwords expire and must be changed. If set to 0, the password never expires. Valid values: 0, 30, 60, 90, 365.</td>
</tr>
<tr>
<td>passwordHistory</td>
<td>int</td>
<td>Required. Number of previous passwords to save. Saving passwords is required to ensure that users reset their password to a new, unique password. This value must be set before a password reset succeeds. If 0, passwordExpiration must be set to 0.</td>
</tr>
<tr>
<td>passwordQuestion</td>
<td>int</td>
<td>Required. If set to 1, the answer to the password hint cannot contain the password itself. If 0, the answer has no restrictions.</td>
</tr>
<tr>
<td>profile</td>
<td>string</td>
<td>Required. Name of the user profile.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example of a ProfilePasswordPolicy component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ProfilePasswordPolicy xmlns="http://soap.sforce.com/2006/04/metadata">
  <forgotPasswordRedirect>true</forgotPasswordRedirect>
  <lockoutInterval>30</lockoutInterval>
  <maxLoginAttempts>0</maxLoginAttempts>
  <minimumPasswordLength>7</minimumPasswordLength>
  <minimumPasswordLifetime>false</minimumPasswordLifetime>
  <obscure>false</obscure>
  <passwordComplexity>1</passwordComplexity>
  <passwordExpiration>0</passwordExpiration>
  <passwordHistory>0</passwordHistory>
  <passwordQuestion>1</passwordQuestion>
  <profile>platformportal</profile>
</ProfilePasswordPolicy>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
Prompt (Beta)

Represents the metadata associated with in-app guidance prompts. Use prompts to display announcements, training, or news to users within the app. Choose to add action button that links to a URL of your choice. Track view and button click completes. This type extends the Metadata metadata type and inherits its fullName field.

As a beta feature, Prompts is a preview and isn’t part of the “Services” under your master subscription agreement with Salesforce. Use this feature at your sole discretion, and make your purchase decisions only on the basis of generally available products and features. Salesforce doesn’t guarantee general availability of this feature within any particular time frame or at all, and we can discontinue it at any time. This feature is for evaluation purposes only, not for production use. It’s offered as is and isn’t supported, and Salesforce has no liability for any harm or damage arising out of or in connection with it. All restrictions, Salesforce reservation of rights, obligations concerning the Services, and terms for related Non-Salesforce Applications and Content apply equally to your use of this feature.

File Suffix and Directory Location

Prompt components have the suffix prompt and are stored in the prompts folder.

Version

Prompt components are available in API version 46.0 and later.

Special Access Rules

Admins have access to add or edit prompts. For non-admin users, assign the Manage Prompts and Modify Metadata user permission. Everyone can see the In-App Guidance setup page.

Packaging Prompts

See Creating Managed Packages in Salesforce Help for more information.

To create a managed package in a Developer edition org, set an org namespace prefix first. You aren’t able to create a prompt without one.

Unmanaged packages for prompts isn’t supported.

When orgs install prompts from packages, the prompts are inactive until the admin activates them.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The master label for the prompt. Maximum of 80 characters.</td>
</tr>
<tr>
<td>promptVersions</td>
<td>PromptVersion</td>
<td>A list of prompt entries. Each entry represent a different prompt.</td>
</tr>
</tbody>
</table>

PromptVersion

A list of prompt entries. Each entry represents a different prompt.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionButtonLabel</td>
<td>string</td>
<td>Label for the action button. Maximum of 25 characters.</td>
</tr>
<tr>
<td>actionButtonLink</td>
<td>string</td>
<td>URL for the action button. Maximum of 255 characters.</td>
</tr>
<tr>
<td>body</td>
<td>string</td>
<td>Required. Body content of the prompt. For floating prompts, there’s a maximum of 120 characters. For docked prompts, there’s a maximum of 4000 characters.</td>
</tr>
<tr>
<td>delayDays</td>
<td>int</td>
<td>Required if recurrences are scheduled. Number of days in between occurrences of the prompt.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the prompt. Maximum of 255 characters.</td>
</tr>
<tr>
<td>dismissButtonLabel</td>
<td>string</td>
<td>Label for the dismiss button. Maximum of 15 characters.</td>
</tr>
<tr>
<td>displayPosition</td>
<td>PromptDisplayPosition (enumeration of type string)</td>
<td>Indicates the position of the floating prompt on the page. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BottomCenter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BottomLeft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BottomRight</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TopCenter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TopLeft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TopRight</td>
</tr>
<tr>
<td>displayType</td>
<td>PromptDisplayType (enumeration of type string)</td>
<td>Required. Indicates the type of prompt. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DockedComposer, which is the docked prompt</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FloatingPanel, which is the floating prompt</td>
</tr>
<tr>
<td>endDate</td>
<td>date</td>
<td>Indicates the date to stop showing the prompt.</td>
</tr>
<tr>
<td>header</td>
<td>string</td>
<td>Label for the header of the docked prompt. This is the label contained in the window’s browser bar. Maximum of 36 characters.</td>
</tr>
<tr>
<td>indexWithIsPublished</td>
<td>string</td>
<td>Used by Salesforce for efficient querying.</td>
</tr>
<tr>
<td>indexWithoutIsPublished</td>
<td>string</td>
<td>Used by Salesforce for efficient querying.</td>
</tr>
<tr>
<td>isPublished</td>
<td>boolean</td>
<td>Indicates if a prompt is active true or not false.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The master label for the prompt.</td>
</tr>
<tr>
<td>publishedByUser</td>
<td>string</td>
<td>The ID of the user who activated the prompt.</td>
</tr>
<tr>
<td>publishedDate</td>
<td>date</td>
<td>Indicates the date the prompt was activated.</td>
</tr>
<tr>
<td>shouldDisplayActionButton</td>
<td>boolean</td>
<td>Required. Indicates if an action button is included in the prompt true or not false.</td>
</tr>
<tr>
<td>startDate</td>
<td>date</td>
<td>Required. Indicates the date to start showing the prompt.</td>
</tr>
<tr>
<td>targetAppDeveloperName</td>
<td>string</td>
<td>Required. The app’s developer name where the prompt appears.</td>
</tr>
<tr>
<td>targetAppNamespacePrefix</td>
<td>string</td>
<td>The app’s namespace prefix where the prompt appears.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>targetPageKey1</td>
<td>string</td>
<td>Required. Used by Salesforce to identify the prompt page location along with targetPageKey2 and targetPageType.</td>
</tr>
<tr>
<td>targetPageKey2</td>
<td>string</td>
<td>Required. Used by Salesforce to identify the prompt page location along with targetPageKey1 and targetPageType.</td>
</tr>
<tr>
<td>targetPageType</td>
<td>string</td>
<td>Required. Used by Salesforce to identify the prompt page location along with targetPageKey1 and targetPageKey2.</td>
</tr>
<tr>
<td>timesToDisplay</td>
<td>int</td>
<td>Required if recurrences are scheduled. Maximum number of times to display the prompt (that is, the number of occurrences). Salesforce detects if the user interacts with (or ignores) the prompt to determine if we should show the prompt again or cancel scheduled recurrences. This might run counter to the number of occurrences scheduled. Maximum value of 30.</td>
</tr>
<tr>
<td>title</td>
<td>string</td>
<td>Required. The label for the title of the prompt. Maximum of 36 characters.</td>
</tr>
<tr>
<td>uiFormulaRule</td>
<td>UiFormulaRule</td>
<td>A set of one or more permission filters that define the conditions under which the prompt displays on the page. If the rule evaluates to <code>true</code>, the prompt displays on the page. If <code>false</code>, it doesn’t display. If this field is <code>null</code>, the prompt displays by default.</td>
</tr>
</tbody>
</table>
| userAccess          | PromptUserAccess (enumeration of type string) | Required. Indicates who can see the prompt. Valid values are:  
- `Everyone`, which indicates that there’s no restrictions on who can see the prompt  
- `SpecificPermissions`, which indicates that only users with all the specific user permissions specified can see the prompt  
Users must have the View Setup and Configuration permission to see prompts only visible with specific permissions. They can see prompts visible to everyone without this user permission. |
| versionNumber       | int        | Required. The number remains 1 since multiple versions aren’t saved in the org.                                                             |

**UiFormulaRule**

A set of one or more filters that define the conditions under which a prompt displays on a Lightning page.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Specifies the AND filter condition.</td>
</tr>
<tr>
<td>criteria</td>
<td>UiFormulaCriterion</td>
<td>List of one or more filters that, when evaluated, determine prompt visibility.</td>
</tr>
</tbody>
</table>

**UiFormulaCriterion**

A single filter that, when evaluated, helps define prompt visibility on a Lightning page.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>leftValue</td>
<td>string</td>
<td>Required. The field upon which the filter is based. Only standard and custom permissions can be included. You can use these expressions in the leftValue field when setting filters for prompt visibility.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>{!$Permission.CustomPermission.permissionName}</code> — Use this expression to control prompt visibility based on the custom permissions of the user viewing the Lightning page. Supported for app, Home, and record pages only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>{!$Permission.StandardPermission.permissionName}</code> — Use this expression to control prompt visibility based on the standard permissions of the user viewing the Lightning page. Supported for app, Home, and record pages only.</td>
</tr>
<tr>
<td>operator</td>
<td>string</td>
<td>Required. Defines the operator used to filter the data. Valid value is EQUAL.</td>
</tr>
<tr>
<td>rightValue</td>
<td>boolean</td>
<td>Specifies if you want to evaluate the prompt's visibility. Use true.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of a Prompt component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Prompt xmlns="http://soap.sforce.com/2006/04/metadata">
  <masterLabel>Prompt Master Label</masterLabel>
  <promptVersions>
    <actionButtonLabel>Learn How</actionButtonLabel>
    <body>Explore how the Path and the Kanban view can help you track, manage, and update your records.</body>
    <delayDays>1</delayDays>
    <description>Kanban floating prompt</description>
    <dismissButtonLabel>OK</dismissButtonLabel>
    <displayPosition>TopLeft</displayPosition>
    <displayType>FloatingPanel</displayType>
    <endDate>2019-03-11</endDate>
    <isPublished>true</isPublished>
    <masterLabel>Prompt Master Label</masterLabel>
    <publishedByUserName>mar3Test mar3User</publishedByUserName>
    <publishedDate>2019-03-11</publishedDate>
    <shouldDisplayActionButton>false</shouldDisplayActionButton>
    <startDate>2019-03-11</startDate>
    <targetAppDeveloperName>LightningSales</targetAppDeveloperName>
    <targetAppNamespacePrefix>standard</targetAppNamespacePrefix>
</promptVersions>
</Prompt>
```
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
  </types>
</Package>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Queue

Represents a holding area for items before they are processed.

Declarative Metadata File Suffix and Directory Location

The file suffix for queue components is .queue and components are stored in the queues directory of the corresponding package directory. This component supports cases, leads, service contracts (if Entitlements are enabled), and custom objects.

Version

Queue components are available in API version 24.0 and later.

Fields

This metadata type represents the valid values that define a queue:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doesSendEmailToMembers</td>
<td>boolean</td>
<td>Indicates whether emails are sent to queue members (true) or not (false) when a new record is added to the queue.</td>
</tr>
<tr>
<td>email</td>
<td>string</td>
<td>The email address of the queue owner.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the queue. Corresponds to Label in the user interface.</td>
</tr>
<tr>
<td>queueMembers</td>
<td>QueueMembers[]</td>
<td>Represents queue members added to the queue. Members can be added directly or selected by roles and public groups they belong to. Available in API version 42.0 and later.</td>
</tr>
<tr>
<td>queueRoutingConfig</td>
<td>string</td>
<td>Routing configuration name. Applies to orgs that use Omni-Channel with a routing configuration. Available in API version 42.0 and later.</td>
</tr>
<tr>
<td>queueObject</td>
<td>QueueObject[]</td>
<td>Indicates the supported entity types.</td>
</tr>
</tbody>
</table>
QueueMembers

Represents queue members added to the queue. Members can be added directly as users or selected by the roles and public groups they belong to. Available in API version 42.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>publicGroups</td>
<td>PublicGroups[]</td>
<td>Represents public groups in the org. Public groups are optionally used to add queue members.</td>
</tr>
<tr>
<td>roleAndSubordinates</td>
<td>RoleAndSubordinates[]</td>
<td>Represents roles and their subordinates in the org’s role hierarchy, including customer and partner roles. Roles and their subordinate hierarchy are optionally used to add queue members.</td>
</tr>
<tr>
<td>roleAndSubordinatesInternal</td>
<td>RoleAndSubordinatesInternal[]</td>
<td>Represents internal roles and their subordinates in the org’s role hierarchy, excluding customer and partner roles. Roles and their subordinate hierarchy are optionally used to add queue members.</td>
</tr>
<tr>
<td>roles</td>
<td>Roles[]</td>
<td>Represents roles in the org. Roles are optionally used to add queue members.</td>
</tr>
<tr>
<td>users</td>
<td>Users[]</td>
<td>Represents users in the org. Users can be added directly as queue members.</td>
</tr>
</tbody>
</table>

PublicGroups

Represents public groups in the org. Public groups are optionally used to add queue members. Available in API version 42.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>publicGroup</td>
<td>string</td>
<td>Represents a public group.</td>
</tr>
</tbody>
</table>

RoleAndSubordinates

Represents roles and their subordinates in the org’s role hierarchy, including customer and partner roles. Roles and their subordinate hierarchy can be used to add queue members. Available in API version 42.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>roleAndSubordinate</td>
<td>string</td>
<td>Represents a role and its subordinates, including customer and partner roles.</td>
</tr>
</tbody>
</table>

RoleAndSubordinatesInternal

Represents internal roles and their subordinates in the org’s role hierarchy, excluding customer and partner roles. Roles and their subordinate hierarchy can be used to add queue members. Available in API version 42.0 and later.
Roles
Represents roles in the org. Roles can be used to add queue members. Available in API version 42.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>roleAndSubordinateInternal</td>
<td>string</td>
<td>Represents a role and its subordinates, excluding customer and partner roles.</td>
</tr>
</tbody>
</table>

Users
Represents users in the org. Users can be added directly as queue members. Available in API version 42.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>string</td>
<td>Represents a user.</td>
</tr>
</tbody>
</table>

QueueSobject
QueueSobject represents an entity type that the queue supports.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| sobjectType  | string     | Valid values are:  
|              |            | • Case  
|              |            | • ContactRequest  
|              |            | • Lead  
|              |            | • ServiceContract  
|              |            | • Custom objects (such as ObjA_c) |

Declarative Metadata Sample Definition
The following is the definition of a queue, which supports Case, Lead, and a custom object named ObjA.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Queue xmlns="http://soap.sforce.com/2006/04/metadata">
  <doesSendEmailToMembers>true</doesSendEmailToMembers>
  <email>member@company.com</email>
  <fullName>Your Name</fullName>
  <name>memberQueue</name>
  <queueSobject>
    <sobjectType>Case</sobjectType>
  </queueSobject>
</Queue>
```
Here's another definition of a queue containing queue members added directly or via public groups and roles. Queries retrieve values using the `DeveloperName` field, not the `Name` field, so that the returned names are unique. The query also appends letters to the end of duplicate names, so these groups and roles can be referred to independently.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Queue xmlns="http://soap.sforce.com/2006/04/metadata">
    <doesSendEmailToMembers>false</doesSendEmailToMembers>
    <name>queue1</name>
    <members>
        <publicGroups>
            <publicGroup>All Internal Users</publicGroup>
        </publicGroups>
        <queueRoleAndSubordinates>
            <queueRoleAndSubordinate>role1</queueRoleAndSubordinate>
            <queueRoleAndSubordinate>role2</queueRoleAndSubordinate>
            <queueRoleAndSubordinate>role3</queueRoleAndSubordinate>
        </queueRoleAndSubordinates>
        <roles>
            <role>role1</role>
        </roles>
        <users>
            <user>s@sm.com</user>
            <user>std@sm.com</user>
        </users>
    </members>
    <queueRoutingConfig>my_omni_routing_config</queueRoutingConfig>
</Queue>
```

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**QueueRoutingConfig**

Represents the settings that determine how work items are routed to agents. This type extends the `Metadata` metadata type and inherits its `fullName` field.
File Suffix and Directory Location
ServicePresenceStatus components have the suffix .queueRoutingConfig and are stored in the queueRoutingConfigs folder.

Version
QueueRoutingConfig components are available in API version 44.0 and later.

Special Access Rules
This type is available only if Omni-Channel is enabled in your org.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>capacityPercentage</td>
<td>double</td>
<td>The percentage of an agent’s capacity for work items that’s consumed by a specific type of work item from this service channel. For example, you might give phone calls a capacity percentage of 100. If an agent receives a phone call, the agent won’t receive new work items until the call ends, because at that point the agent’s capacity will have reached 100%.</td>
</tr>
<tr>
<td>capacityWeight</td>
<td>double</td>
<td>The amount of an agent’s capacity for work items that’s consumed by a work item from this service channel. For example, if an agent has a capacity of 6, and cases are assigned a capacity weight of 2, an agent can be assigned up to 3 cases before the agent is at capacity and can’t receive new work items.</td>
</tr>
<tr>
<td>dropAdditionalSkillsTimeout</td>
<td>int</td>
<td>The number of seconds to elapse before additional skills are dropped from Omni-Channel routing. In skills-based routing, you can set some skills to Additional Skill. After the timeout elapses, a skill marked as Additional Skill is dropped from Omni-Channel routing and the case is routed to the best-matched agent, even if the agent doesn’t have all the skills.</td>
</tr>
<tr>
<td>isAttributeBased</td>
<td>boolean</td>
<td>Indicates whether this routing configuration is used with attribute setup for skills-based routing (true) or not (false).</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The label of the presence status.</td>
</tr>
<tr>
<td>pushTimeout</td>
<td>int</td>
<td>The number of seconds set for push timeout. 0 is returned when push timeout isn’t enabled.</td>
</tr>
<tr>
<td>queueOverflowAssignee</td>
<td>string</td>
<td>The ID of the queue that’s set as the Overflow Assignee.</td>
</tr>
<tr>
<td>routingModel</td>
<td>RoutingModel (enumeration of type string)</td>
<td>Required. The routing type that determines how work items are routed (pushed) to agents. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LEAST_ACTIVE</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
| | | • MOST_AVAILABLE
| | | • EXTERNAL_ROUTING

**routingPriority** int Required. The priority in which work items from the service channels that are related to this routing configuration are routed to agents. Work items from routing configurations that have lower priority values (for example, 0) are routed to agents first.

**userOverflowAssignee** string The ID of the user that’s set as the Overflow Assignee.

---

**Declarative Metadata Sample Definition**

The following is an example of a QueueRoutingConfig component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<QueueRoutingConfig xmlns="http://soap.sforce.com/2006/04/metadata">
  <capacityWeight>1.0</capacityWeight>
  <label>Case Routing</label>
  <pushTimeout>120</pushTimeout>
  <queueOverflowAssignee>queueOverflow</queueOverflowAssignee>
  <routingModel>LEAST_ACTIVE</routingModel>
  <routingPriority>1</routingPriority>
</QueueRoutingConfig>
```

The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>QueueRoutingConfig</name>
  </types>
  <version>44.0</version>
</Package>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](#).

### QuickAction

Represents a specified create or update quick action for an object that then becomes available in the Chatter publisher. For example, you can create an action that, on the detail page of an account, allows a user to create a contact related to that account from the Chatter feed on that page. QuickAction can be created on objects that allow custom fields. The parent objects supported include:

- Account
- Campaign
Case  
Contact  
ContentNote  
Custom objects  
Group  
Lead  
Opportunity

File Suffix and Directory Location

QuickAction components have the suffix quickAction and are stored in the quickActions folder.

Version

QuickAction components are available in API version 28.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>canvas</td>
<td>string</td>
<td>If the custom action invokes a Canvas app, the app name. Returns the fully qualified name of the Canvas app in the format &lt;namespace&gt;__&lt;dev_name&gt;, if the quick action type is Canvas; otherwise, returns null. This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the action.</td>
</tr>
<tr>
<td>fieldOverrides</td>
<td>FieldOverride on page 688</td>
<td>The specific field that may be overridden within a QuickAction.</td>
</tr>
<tr>
<td>flowDefinition</td>
<td>string</td>
<td>If the custom action invokes a flow, this field represents the API name of the flow. Otherwise, this field is null.</td>
</tr>
<tr>
<td>height</td>
<td>int</td>
<td>If a custom action is created, this field represents the height in pixels of the action pane.</td>
</tr>
<tr>
<td>icon</td>
<td>string</td>
<td>The icon used to identify the action. API version 32.0 and later returns different icons than in earlier API versions.</td>
</tr>
<tr>
<td>isProtected</td>
<td>boolean</td>
<td>Indicates whether this component is protected (true) or not (false). Protected components cannot be linked to or referenced by components created in the installing organization.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Identifies the action and displays to users. This is also the default identifier used for the API and managed packages.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>lightningComponent</td>
<td>string</td>
<td>If the custom action invokes a Lightning component, this field represents the fully qualified name of the component. Otherwise, this field is null. Available in API version 38.0 and later.</td>
</tr>
<tr>
<td>optionsCreateFeedItem</td>
<td>boolean</td>
<td>Required. Indicates whether successful completion of the action creates a feed item (true) or not (false). Applies only to Create Record, Update Record, and Log a Call quick action types. Available in API version 36.0 and later.</td>
</tr>
<tr>
<td>page</td>
<td>string</td>
<td>If the custom action invokes a Visualforce page, this field identifies the page.</td>
</tr>
<tr>
<td>quickActionLayout</td>
<td>QuickActionLayout</td>
<td>The layout of fields on the action.</td>
</tr>
<tr>
<td>standardLabel</td>
<td>QuickActionLabel (enumeration of type string)</td>
<td>Specifies the standard label to use for the action. The valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- AddRecord</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- AddMember</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ChangeDueDate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ChangePriority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ChangeStatus</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- CreateNew</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- CreateNewRecordType (For example, a label with something like “Create New Idea”)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Defer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- EditDescription</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Escalate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- EscalateToRecord</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Forward (Available in API version 42.0 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- LogACall</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- LogANote</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- New (A new record)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NewChild (A new child record)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NewChildRecordType</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NewRecordType (For example, a label with something like “New Idea”)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- OfferFeedback</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Quick (A quick record)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- QuickRecordType</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Reply (Available in API version 42.0 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ReplyAll (Available in API version 42.0 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- RequestFeedback</td>
</tr>
</tbody>
</table>
### Metadata Types

#### QuickAction

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|                    |            | • SendEmail (This value is available in API version 31.0 and later.)  
|                    |            | • Update |
| successMessage     | string     | The message that displays to the user upon successful completion of the action.  
|                    |            | Available in API version 36.0 and later. |
| targetObject       | string     | The object for which the action is created and performed.  
|                    |            | For example, you can create an action that, on the detail page of an account, allows a user to create a contact related to that account from the Chatter feed on that page. In this case, Contact is the targetObject. |
| targetParentField  | string     | The parent object type of the action. Links the target object to the parent object. For example, use Account if the target object is Contact and the parent object is Account. |
| targetRecordType   | string     | Specifies which record type to create. Valid values are:  
|                    |            | • Business Account  
|                    |            | • Person Account  
|                    |            | • Master |
| type               | QuickActionType (enumeration of type string) | Required. The type of quick action. Valid values are:  
|                    |            | • Canvas  
|                    |            | • Create  
|                    |            | • Flow (This value is available as a Beta in API version 41.0 and later)  
|                    |            | • LightningComponent (This value is available in API version 38.0 and later.)  
|                    |            | • LogACall  
|                    |            | • Post  
|                    |            | • SendEmail (This value is available in API version 31.0 and later.)  
|                    |            | • SocialPost  
|                    |            | • Update  
|                    |            | • VisualforcePage |
| width              | int        | If a custom action is created, this field represents the width in pixels of the action pane. |

### FieldOverride

Represents the field names and their respective formulas and literal values that comprise predefined value settings for a QuickAction. If a field on an action has both a predefined value and a default value set, the action uses the predefined value, not the default value. A formula value takes precedence over a literal value if both are defined.
### Field Name | Field Type | Description
---|---|---
field | string | Required. The name of the field to allow predefined values on.
formula | string | Specifies the formula to use when setting a field’s predefined value. Supported for single-select picklists as of API version 43.0.
literalValue | string | Supported for picklists only. Specifies the literal value of the field defined from values in the picklist. Corresponds to the Specific Value field in the predefined value UI.

**QuickActionLayout**

The layout of fields on the action. There is no hard limit to the number of fields you can add to an action layout. However, for optimum usability, we recommend a maximum of eight fields. Adding more than 20 fields can severely affect user efficiency.

### Field Name | Field Type | Description
---|---|---
layoutSectionStyle | LayoutSectionStyle | Required. The type of layout structure used. The valid values are:
- TwoColumnsTopToBottom
- TwoColumnsLeftToRight
- OneColumn
- CustomLinks

**quickActionLayoutColumns** | QuickActionLayoutColumn[] | Specifies columns in a QuickActionLayout.

**QuickActionLayoutColumn**

A column defined for a QuickActionLayout.

### Field Name | Field Type | Description
---|---|---
quickActionLayoutItems | QuickActionLayoutItem[] | Specifies row items in a QuickActionLayoutColumn.

**QuickActionLayoutItem**

A row item comprised of fields and defined for a QuickActionLayoutColumn.

### Field Name | Field Type | Description
---|---|---
emptySpace | boolean | Controls if this layout item is a blank space (true) or not (false).
field | string | Represents a specific field in QuickActionLayoutItem. There is no hard limit to the number of fields you can add to an action layout. However, for optimum usability, we recommend a maximum of eight fields. Adding more than 20 fields can severely affect user efficiency.
### Field Type: Field Name

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uiBehavior</td>
<td>UiBehavior (enumeration of type string)</td>
<td>Specifies user input behavior for specific fields in QuickActionLayoutItem. The valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Edit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Required</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Readonly</td>
</tr>
</tbody>
</table>

#### Declarative Metadata Sample Definition

The following is an example of a QuickAction component:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<QuickAction xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>testActionDefinitionTypesCreateTask</description>
  <label>testActionDefinitionTypesCreateTask</label>
  <optionsCreateFeedItem>true</optionsCreateFeedItem>
  <quickActionLayout>
    <layoutSectionStyle>TwoColumnsLeftToRight</layoutSectionStyle>
    <quickActionLayoutColumns>
      <quickActionLayoutItems>
        <emptySpace>false</emptySpace>
        <field>OwnerId</field>
        <uiBehavior>Required</uiBehavior>
      </quickActionLayoutItems>
      <quickActionLayoutItems>
        <emptySpace>false</emptySpace>
        <field>WhoId</field>
        <uiBehavior>Edit</uiBehavior>
      </quickActionLayoutItems>
      <quickActionLayoutItems>
        <emptySpace>false</emptySpace>
        <field>WhatId</field>
        <uiBehavior>Edit</uiBehavior>
      </quickActionLayoutItems>
      <quickActionLayoutItems>
        <emptySpace>false</emptySpace>
        <field>ActivityDate</field>
        <uiBehavior>Edit</uiBehavior>
      </quickActionLayoutItems>
      <quickActionLayoutItems>
        <emptySpace>false</emptySpace>
        <field>Subject</field>
        <uiBehavior>Edit</uiBehavior>
      </quickActionLayoutItems>
      <quickActionLayoutItems>
        <emptySpace>false</emptySpace>
        <field>Status</field>
        <uiBehavior>Required</uiBehavior>
      </quickActionLayoutItems>
    </quickActionLayoutColumns>
  </quickActionLayout>
</QuickAction>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

RecommendationStrategy

Represents a recommendation strategy. Recommendation strategies are applications, similar to data flows, that determine a set of recommendations to be delivered to the client through data retrieval, branching, and logic operations.

File Suffix and Directory Location

RecommendationStrategy components have the suffix .recommendationStrategy and are stored in the recommendationStrategies folder.

Version

RecommendationStrategy components are available in API version 45.0 and later.

Special Access Rules

Metadata access for the RecommendationStrategy type is backed by the ManageRecommendationStrategies user permission.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionContext</td>
<td>StrategyAction[]</td>
<td>An array of action contexts used by the strategy.</td>
</tr>
<tr>
<td>contextRecordType</td>
<td>string</td>
<td>The sObject type of the $record used by the flow.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the recommendation strategy.</td>
</tr>
<tr>
<td>filter</td>
<td>StrategyNodeFilter[]</td>
<td>An array of filter nodes.</td>
</tr>
</tbody>
</table>
### StrategyNodeBase

Base class for all strategy nodes. This is an abstract class.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>childNode</td>
<td>string</td>
<td>Array of child node names, in order of execution.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the node.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Label of the node.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name of the node.</td>
</tr>
</tbody>
</table>

### StrategyAction

Defines a call to an invocable action from the strategy. Results are used by decision elements in the strategy.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>string</td>
<td>Required. The name or id of the InvocableAction to execute.</td>
</tr>
<tr>
<td>argument</td>
<td>StrategyActionArg[]</td>
<td>List of strategy action arguments.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the strategy.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Label for the strategy action.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name of the strategy action, which is referenced by decisioning elements in the strategy.</td>
</tr>
</tbody>
</table>
### StrategyActionArg

Defines arguments passed to invocable actions associated with a strategy action.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the parameter to pass to the invocable action.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Required. A Salesforce formula expression that is evaluated with the result being used as the parameter value for the Strategy Action.</td>
</tr>
</tbody>
</table>

### StrategyNodeUnionBase

Base class for nodes that perform a union of their children. Union nodes combine the outputs of their children to form the input to themselves. StrategyNodeUnionBase extends StrategyNodeBase and inherits all of its fields. This is an abstract class.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>limit</td>
<td>int</td>
<td>Maximum number of results to output.</td>
</tr>
</tbody>
</table>

### StrategyNodeFilter

Defines a filter element that filters recommendations. It extends StrategyNodeUnionBase and inherits all of its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>expression</td>
<td>string</td>
<td>Required. A formula expression that results in a boolean value when executed on each recommendation in the node’s input. Inputs that result in true form the output, and inputs that result in false are excluded.</td>
</tr>
</tbody>
</table>

### StrategyNodeIf

Selects specific children to execute and combines their results. Executes and returns results of children based on the array of child node expressions. Extends StrategyNodeUnionBase and inherits all of its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>childNodeExpression</td>
<td>IfExpression[]</td>
<td>Array of if expressions.</td>
</tr>
<tr>
<td>onlyFirstMatch</td>
<td>boolean</td>
<td>If true, selects only the results from the matching child. If false, selects and combines results from all matching children. The default value is false.</td>
</tr>
</tbody>
</table>
IfExpression

Expression used by StrategyNodeInvocableAction.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>childName</td>
<td>string</td>
<td>Required. Name of child to match.</td>
</tr>
<tr>
<td>expression</td>
<td>string</td>
<td>Required. Formula expression returning true or false.</td>
</tr>
</tbody>
</table>

StrategyNodeInvocableAction

Defines an element that calls an Apex invocable action to generate or enhance a list of recommendations. It extends StrategyNodeUnionBase and inherits all its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>string</td>
<td>Required. The name of the invocable action to execute.</td>
</tr>
<tr>
<td>argument</td>
<td>StrategyNodeInvocableActionArg</td>
<td>List of arguments that are passed to the invocable action.</td>
</tr>
<tr>
<td>isGenerator</td>
<td>boolean</td>
<td>Required. If true, the UI displays the Generate element. If false, the UI displays the Enhance element. Defaults to false.</td>
</tr>
<tr>
<td>type</td>
<td>InvocableActionType (enumeration of type string)</td>
<td>Required. Type of the invocable action. See the enum InvocableActionType. Valid value is apex.</td>
</tr>
</tbody>
</table>

StrategyNodeInvocableActionArg

Defines arguments passed to an Apex invocable action that generates or enhances a list of recommendations.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the parameter to pass to the invocable action. The name must match a parameter that's defined in the invocable action.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Required. A Salesforce formula expression that is evaluated with the result used as the parameter value for the action.</td>
</tr>
</tbody>
</table>

StrategyNodeRecommendationLimit

Filters out recommendations that have already been accepted or rejected. Extends StrategyNodeUnionBase and inherits all of its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterMode</td>
<td>StrategyReactionType (enumeration of type string)</td>
<td>Available reactions to filter out. The valid values are: • Accepted • Rejected</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
lookbackDuration | int | Number of days to search back.  
maxRecommendationCount | int | Maximum number of times recommendation has been accepted or rejected.

---

**StrategyNodeRecommendationLoad**

Retrieves Recommendation objects. Extends StrategyNodeUnionBase and inherits all of its fields.

### Field Name | Field Type | Description
--- | --- | ---
condition | RecommendationLoadCondition[] | Array of conditions specifying which recommendations to load.
conditionLogic | string | Logic to combine conditions, either AND or OR. All conditions are combined (not mixed). For example: Cond1 AND Cond2 AND Cond3.

---

**RecommendationLoadCondition**

Represents a condition used as part of the query constructed by StrategyNodeRecommendationLoad.

### Field Name | Field Type | Description
--- | --- | ---
field | string | Required. Any field from Recommendation BPO (SOAP) object.
operator | RecommendationConditionOperator (enumeration of type string) | Required. Valid values are:
  * EQUALS
  * GREATER_THAN
  * GREATER_THAN_OR_EQUAL_TO
  * LESS_THAN
  * LESS_THAN_OR_EQUAL_TO
  * NOT_EQUALS
  * LIKE
  * STARTS_WITH
  * ENDS_WITH=
  * CONTAINS

value | RecommendationConditionValue | Required. Constant value to use in query.

---

**RecommendationConditionValue**

Represents a value used as part of a RecommendationCondition.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>RecommendationOutcomeValue</td>
<td>Required. Valid values are: TEXT, NUMBER, BOOLEAN, DATE, DATE_TIME, TIME</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Required. The constant value.</td>
</tr>
</tbody>
</table>

**StrategyNodeSort**

Sorts the recommendations. Extends StrategyNodeUnionBase and inherits all of its fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>StrategyNodeSortField</td>
<td>Required. Fields to sort on.</td>
</tr>
</tbody>
</table>

**StrategyNodeSortField**

Defines the fields for sorting by StrategyNodeSort.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Name of the field to sort.</td>
</tr>
<tr>
<td>nullsFirst</td>
<td>boolean</td>
<td>If true, null values are sorted to the beginning of the list. Defaults to false.</td>
</tr>
<tr>
<td>order</td>
<td>SortOrder</td>
<td>Order in which the list is sorted. Defaults to Asc. Valid values are: Asc (ascending) Desc (descending)</td>
</tr>
</tbody>
</table>

**StrategyNodeUnion**

StrategyNodeUnion combines the output of all its child nodes. StrategyNodeUnion is a concrete implementation of StrategyNodeUnionBase and inherits all its fields.

**StrategyNodeMap**

Set recommendation fields with values. Extends StrategyNodeUnionBase and inherits all of its fields.
StrategyNodeExclusive

Returns results from the first child node that has results and no other. Extends StrategyNodeUnionBase and inherits all its fields.

MapExpression

Sets the value for a recommendation field used by the strategy.

### Field Name | Field Type | Description
--- | --- | ---
expression | string | Required. A formula expression that results in a valid value supported by the data type specified in the type field.
name | string | Required. Recommendation field name that the expression sets the value for.
type | string | Required. The data type of the value resulting from the value in the expression field.

Valid values are:
- BOOLEAN
- CURRENCY
- DATE
- DOUBLE
- DATE_TIME
- INTEGER
- LONG
- PERCENT
- TEXT
- TIME

### Declarative Metadata Sample Definition

The following is an example of a RecommendationStrategy component that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<RecommendationStrategy xmlns="http://soap.sforce.com/2006/04/metadata">
  <contextRecordType>Asset</contextRecordType>
  <description>Hills Brothers Coffee strategy to handle machine down incidents</description>
  <if>
    <childNodes>IfNoEscaladeOrBetterSupport</childNodes>
  </if>
  <childNodes>IfModel</childNodes>
</RecommendationStrategy>
```
<description>Id Machine Down</description>
<label>RootNode</label>
<name>RootNode</name>
<childNodeExpression>
    <childName>IfModel</childName>
    <expression>ISPICKVAL($Record.Status, &quot;OutOfOrder&quot;)</expression>
</childNodeExpression>
<childNodeExpression>
    <childName>IfNoEscaladeOrBetterSupport</childName>
    <expression>ISPICKVAL($Record.Status, &quot;OutOfOrder&quot;)</expression>
</childNodeExpression>
<onlyFirstMatch>false</onlyFirstMatch>

<if>
    <childNode>LoadEscalade</childNode>
    <childNode>LoadMiniDiagnostic</childNode>
    <childNode>LoadMaxiDiagnostic</childNode>
    <description>If Machine Model switch node</description>
    <label>IfModel</label>
    <name>IfModel</name>
    <childNodeExpression>
        <childName>LoadMiniDiagnostic</childName>
        <expression>$Record.Product2.Name== &quot;Mini Coffee Roaster&quot;</expression>
    </childNodeExpression>
    <childNodeExpression>
        <childName>LoadMaxiDiagnostic</childName>
        <expression>$Record.Product2.Name== &quot;Maxi Coffee Roaster&quot;</expression>
    </childNodeExpression>
    <onlyFirstMatch>false</onlyFirstMatch>
</if>

<if>
    <childNode>LoadEscalade</childNode>
    <description>If Customer does not have escalade support plan</description>
    <label>IfNoEscaladeOrBetterSupport</label>
    <name>IfNoEscaladeOrBetterSupport</name>
    <childNodeExpression>
        <childName>LoadEscalade</childName>
        <expression>NOT(ISPICKVAL($Record.Account.SLA__c, &quot;Gold&quot;) || ISPICKVAL($Record.Account.SLA__c, &quot;Platinum&quot;))</expression>
    </childNodeExpression>
    <onlyFirstMatch>false</onlyFirstMatch>
</if>

<recommendationLoad>
    <description>Load upgrade to escalade support plan</description>
    <label>LoadEscalade</label>
    <name>LoadEscalade</name>
    <condition>
        <field>Name</field>
        <operator>EQUALS</operator>
        <value>
            <type>TEXT</type>
            <value>Upgrade your Maintenance Package</value>
        </value>
    </condition>
</recommendationLoad>
<value/>
</condition>
<conditionLogic>and</conditionLogic>
</recommendationLoad>
<recommendationLoad>
<description>Load Mini Coffee Roaster Diagnostic Troubleshooting proposition</description>
<label>LoadMiniDiagnostic</label>
<name>LoadMiniDiagnostic</name>
<condition>
    <field>Name</field>
    <operator>EQUALS</operator>
    <value>
        <type>TEXT</type>
        <value>Mini Coffee Roaster Diagnostic Troubleshooting</value>
    </value>
</condition>
<conditionLogic>and</conditionLogic>
</recommendationLoad>
<recommendationLoad>
<description>Load Maxi Coffee Roaster Diagnostic Troubleshooting proposition</description>
<label>LoadMaxiDiagnostic</label>
<name>LoadMaxiDiagnostic</name>
<condition>
    <field>Name</field>
    <operator>EQUALS</operator>
    <value>
        <type>TEXT</type>
        <value>Maxi Coffee Roaster Diagnostic Troubleshooting</value>
    </value>
</condition>
<conditionLogic>and</conditionLogic>
</recommendationLoad>
<union>
    <childNode>RootNode</childNode>
    <label>Output</label>
    <name>Output</name>
</union>
<invocableAction>
    <action>MyInvocableApexClass</action>
    <isGenerator>true</isGenerator>
    <type>apex</type>
    <argument>
        <name>MyNameParam</name>
        <value>$User.FirstName</value>
    </argument>
    <argument>
        <name>MyIdParam</name>
        <value>$Record.Id</value>
    </argument>
</invocableAction>
<map>
    <expression>
**RecordActionDeployment**

Represents configuration settings for the Actions & Recommendations component. For example, you can have a deployment that specifies which types of actions to display, default actions for channels, and the actions that users can add at runtime. If the component shows Next Best Action recommendations, the deployment configures which strategies to use and how recommendations appear. This type extends the `Metadata` metadata type and inherits its `fullName` field.

**File Suffix and Directory Location**

RecordActionDeployment values are stored in the `developer_name.deployment` file in the `recordActionDeployments` directory.

**Note:** We don’t recommend programmatically changing the API name of a RecordActionDeployment.

**Version**

RecordActionDeployment is available in API version 45.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channelConfigurations</td>
<td><code>RecordActionDeploymentChannel</code></td>
<td>Specifies configuration settings for different channels in an Actions &amp; Recommendations deployment.</td>
</tr>
<tr>
<td>deploymentContexts</td>
<td><code>RecordActionDeploymentContext</code></td>
<td>Specifies the object context for quick actions and Next Best Action strategies. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>hasGuidedActions</td>
<td>boolean</td>
<td>Specifies that the component shows standard actions; for example, flows and quick actions. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>hasRecommendations</td>
<td>boolean</td>
<td>Specifies that the component shows recommendations from a Next Best Action strategy. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. Specifies the name of the deployment.</td>
</tr>
<tr>
<td>recommendation</td>
<td><code>RecordActionRecommendation</code></td>
<td>Specifies settings for how Next Best Action recommendations appear in the component. Available in API version 46.0 and later.</td>
</tr>
</tbody>
</table>
### RecordActionSelectableItem

Specifies the actions that users can add at runtime.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selectableItems</td>
<td>RecordActionSelectableItem</td>
<td>Specifies the actions that users can add at runtime.</td>
</tr>
</tbody>
</table>

### RecordActionDefaultItem

Represents actions and attributes specified as channel defaults in a deployment.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>string</td>
<td>Required. Specifies the API name of an action. For example, the API name of a flow, such as Verify_Information.</td>
</tr>
<tr>
<td>isMandatory</td>
<td>boolean</td>
<td>Specifies whether the action is marked as mandatory. The default value is false.</td>
</tr>
<tr>
<td>isUiRemoveHidden</td>
<td>boolean</td>
<td>Specifies whether the remove option is hidden in the UI. The default value is false. If true, the UI hides the ability to remove the action from the list.</td>
</tr>
<tr>
<td>pinned</td>
<td>PinnedAction (enumeration of type string)</td>
<td>Required. Indicates whether the action is pinned to the Top or Bottom, or unpinned (None). The default value is None.</td>
</tr>
<tr>
<td>position</td>
<td>int</td>
<td>Required. Indicates the order of the action among all actions associated with this record.</td>
</tr>
</tbody>
</table>
| type               | RecordActionType (enumeration of type string) | Required. The type of action that’s associated with the record. Valid values are:  
  - Flow  
  - QuickAction (Available in API version 46.0 and later.) |

### RecordActionDeploymentChannel

Specifies channel-specific defaults to show in the Actions & Recommendations component. The component displays the channel defaults when the list is otherwise empty.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>ChannelSource (enumeration of type string)</td>
<td>Required. Specifies the channel. Valid values are Phone, Chat, or Default.</td>
</tr>
<tr>
<td>channelItems</td>
<td>RecordActionDefaultItem</td>
<td>Specifies default actions for a channel and attributes for each action, such as whether the action is pinned to the list top or bottom or whether an action is considered mandatory.</td>
</tr>
<tr>
<td>isAutopopEnabled</td>
<td>boolean</td>
<td>Specifies whether the first action in the list is launched when the record page opens. If true, the first action is launched. The default value is false.</td>
</tr>
</tbody>
</table>
**RecordActionDeploymentContext**

Specifies an object that provides context for quick actions and Next Best Action strategies. When the component appears on this type of page, it includes object-specific quick actions and uses an object-specific strategy to filter recommendations. Available in API version 46.0 and later.

> **Note:** We support a maximum of 10 objects that provide context within a deployment.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityName</td>
<td>string</td>
<td>Required. Specifies the API name of an object to use as context.</td>
</tr>
<tr>
<td>recommendationStrategy</td>
<td>string</td>
<td>Specifies the API name of a Next Best Action strategy that overrides the default strategy on this page. A strategy is a metadata type RecommendationStrategy.</td>
</tr>
</tbody>
</table>

**RecordActionRecommendation**

Specifies settings to display Next Best Action recommendations in the component. Available in API version 46.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultStrategy</td>
<td>string</td>
<td>Specifies the API name of the default Next Best Action strategy, which is a metadata type RecommendationStrategy.</td>
</tr>
<tr>
<td>hasDescription</td>
<td>boolean</td>
<td>Required. If true, display the description for the recommendation.</td>
</tr>
<tr>
<td>hasImage</td>
<td>boolean</td>
<td>Required. If true, display the image for the recommendation.</td>
</tr>
<tr>
<td>hasRejectAction</td>
<td>boolean</td>
<td>Required. If true, display the label that the user clicks to reject the recommendation.</td>
</tr>
<tr>
<td>hasTitle</td>
<td>boolean</td>
<td>Required. If true, display the title for the recommendation.</td>
</tr>
<tr>
<td>maxDisplayRecommendations</td>
<td>int</td>
<td>Required. Specifies the maximum number of recommendations to display. Valid values are 1–4.</td>
</tr>
</tbody>
</table>

**RecordActionSelectableItem**

Represents the set of actions available for users to add to the component at runtime.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>string</td>
<td>Required. Specifies the API name of an action. For example, the API name of a flow, such as Verify_Information.</td>
</tr>
<tr>
<td>type</td>
<td>RecordActionType</td>
<td>Required. The type of action that's associated with the record. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• Flow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• QuickAction (Available in API version 46.0 and later.)</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is a sample of a `recordActionDeployment` file.

```xml
<RecordActionDeployment xmlns="http://soap.sforce.com/2006/04/metadata">
  <channelConfigurations>
    <channel>Phone</channel>
    <channelItems>
      <action>Sample_Flow</action>
      <isMandatory>false</isMandatory>
      <isUiRemoveHidden>false</isUiRemoveHidden>
      <position>1</position>
      <pinned>Top</pinned>
      <type>Flow</type>
    </channelItems>
    <channelItems>
      <action>Another_Sample_Flow</action>
      <isMandatory>false</isMandatory>
      <isUiRemoveHidden>true</isUiRemoveHidden>
      <position>2</position>
      <pinned>Top</pinned>
      <type>Flow</type>
    </channelItems>
    <isAutopopEnabled>true</isAutopopEnabled>
  </channelConfigurations>
  <masterLabel>Sample Deployment</masterLabel>
  <selectableItems>
    <action>Sample_Flow</action>
    <type>Flow</type>
  </selectableItems>
  <selectableItems>
    <action>Sample_Flow_2</action>
    <type>Flow</type>
  </selectableItems>
  <hasGuidedActions>true</hasGuidedActions>
  <hasRecommendations>true</hasRecommendations>
  <recommendation>
    <defaultStrategy>Sample_Global_Strategy</defaultStrategy>
    <maxDisplayRecommendations>4</maxDisplayRecommendations>
    <hasImage>true</hasImage>
    <hasDescription>true</hasDescription>
    <hasRejectAction>true</hasRejectAction>
    <hasTitle>true</hasTitle>
  </recommendation>
  <deploymentContexts>
    <entityName>Case</entityName>
    <recommendationStrategy>Sample_Case_Strategy</recommendationStrategy>
  </deploymentContexts>
  <deploymentContexts>
    <entityName>Account</entityName>
    <recommendationStrategy>Sample_Acc_Strategy</recommendationStrategy>
  </deploymentContexts>
</RecordActionDeployment>
```
The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <fullName>SecondTest</fullName>
  <types>
    <members>Sample_Flow</members>
    <members>Another_Sample_Flow</members>
    <members>Sample_Flow_2</members>
    <name>Flow</name>
  </types>
  <types>
    <members>SampleDeployment</members>
    <name>RecordActionDeployment</name>
  </types>
  <version>45.0</version>
</Package>
```

SEE ALSO:
- RecommendationStrategy

RemoteSiteSetting

Represents a remote site setting. Before any Visualforce page, Apex callout, or JavaScript code using XmlHttpRequest in an s-control or custom button can call an external site, that site must be registered in the Remote Site Settings page, or the call fails.

RemoteSiteSetting extends the Metadata metadata type and inherits its fullName field.

Declarative Metadata File Suffix and Directory Location

RemoteSiteSetting components are stored in the `remoteSiteSettings` directory of the corresponding package directory. The file name matches the unique name of the remote site setting, and the extension is `.remoteSite`.

Version

RemoteSiteSetting components are available in API version 19.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The description explaining what this remote site setting is used for.</td>
</tr>
<tr>
<td>disableProtocolSecurity</td>
<td>boolean</td>
<td>Required. Indicates whether code within Salesforce can access the remote site regardless of whether the user’s connection is over HTTP or HTTPS (true) or not (false). When true, code within Salesforce can pass data from an HTTPS session to an HTTP session, and vice versa.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

A sample XML definition of a remote site setting is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<RemoteSiteSetting xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>Used for Apex callout to mapping web service</description>
  <disableProtocolSecurity>false</disableProtocolSecurity>
  <isActive>true</isActive>
  <url>https://www.maptestsite.net/mapping1</url>
</RemoteSiteSetting>
```

Report

Represents a custom report. This metadata type only supports custom reports; standard reports are not supported.

This type extends the Metadata metadata type and inherits its fullName field.

Declarative Metadata File Suffix and Directory Location

Reports are stored in the reports directory of the corresponding package directory. The file name matches the report title and the extension is .report.

Retrieving Reports

You can’t use the wildcard (*) symbol with reports in package.xml. To retrieve the list of reports for populating package.xml with explicit names, call listMetadata() and pass in ReportFolder as the type. Note that ReportFolder is not returned as a type in describeMetadata(). Report is returned from describeMetadata() with an associated attribute of inFolder.
set to true. If that attribute is set to true, you can construct the type by using the component name with the word Folder, such as ReportFolder.

The following example shows folders in package.xml:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>MyDBFolder/MyDBName</members>
    <name>Dashboard</name>
  </types>
  <types>
    <members>MyDocumentFolder/MyDocumentName</members>
    <name>Document</name>
  </types>
  <types>
    <members>unfiled$public/MarketingProductInquiryResponse</members>
    <members>unfiled$public/SalesNewCustomerEmail</members>
    <name>EmailTemplate</name>
  </types>
  <types>
    <members>MyReportFolder/MyReportName</members>
    <name>Report</name>
  </types>
  <version>46.0</version>
</Package>
```

**Version**

Report components are available in API version 14.0 and later.

**Fields**

The following information assumes that you are familiar with creating and running reports. For more information on these fields, see “Create a Report” in the Salesforce online help.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregates</td>
<td>ReportAggregate[]</td>
<td>List that defines custom summary formulas for summary, matrix, and joined reports.</td>
</tr>
<tr>
<td>block</td>
<td>Report[]</td>
<td>Represents each block in a joined report where every block can be of a different report type.</td>
</tr>
<tr>
<td>blockInfo</td>
<td>ReportBlockInfo</td>
<td>Defines attributes for each block in a joined report.</td>
</tr>
<tr>
<td>buckets</td>
<td>ReportBucketField[]</td>
<td>Defines a bucket field to be used in the report. This field is available in API version 24.0 and later.</td>
</tr>
<tr>
<td>chart</td>
<td>ReportChart</td>
<td>Defines a chart for summary and matrix reports.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>colorRanges</td>
<td>ReportColorRange[]</td>
<td>List that specifies conditional highlighting for report summary data.</td>
</tr>
<tr>
<td>columns</td>
<td>ReportColumn[]</td>
<td>List that specifies the fields displayed in the report. Fields appear in the report in the same order as they appear in the Metadata API file.</td>
</tr>
<tr>
<td>crossFilters</td>
<td>ReportCrossFilter[]</td>
<td>Defines a cross filter’s object, related object, and condition (WITH or WITHOUT). This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>currency</td>
<td>CurrencyIsoCode (enumeration of type string)</td>
<td>When using multiple currencies, some reports allow you to display converted amounts by selecting the appropriate column to display. For example, in opportunity reports, you can include the Amount (converted) column on the report. This field is an enumeration of type string that defines the currency in which to display converted amounts. Valid values: Must be one of the valid alphabetic, three-letter currency ISO codes defined by the ISO 4217 standard, such as USD, GBP, or JPY.</td>
</tr>
<tr>
<td>dataCategoryFilters</td>
<td>string</td>
<td>Specifies a filter according to data category.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Specifies a general description, which is displayed with the report name. Maximum characters: 255 characters.</td>
</tr>
<tr>
<td>division</td>
<td>string</td>
<td>If your organization uses divisions to segment data and you have the “Affected by Divisions” permission, records in the report must match this division. This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>filter</td>
<td>ReportFilter</td>
<td>Limits report results to records with specific data. For example, you can limit report results to opportunities for which the amount is greater than $1,000:</td>
</tr>
</tbody>
</table>

```xml
<filter>
  <criteriaItems>
    <column>AMOUNT</column>
    <operator>greaterThan</operator>
    <value>1000</value>
  </criteriaItems>
</filter>
```
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>folderName</td>
<td>string</td>
<td>Name of the folder that houses the report.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 35.0 and later.</td>
</tr>
<tr>
<td>format</td>
<td>ReportFormat (enumeration of type string)</td>
<td>Defines the report format. For example, Tabular for a simple data list without subtotals.</td>
</tr>
<tr>
<td>groupingsAcross</td>
<td>ReportGrouping[]</td>
<td>List that defines the fields by which you want to group and subtotal data across a matrix report (row headings). When grouping by a date field, you can further group the data by a specific time period such as days, weeks, or months. Maximum: 2 fields.</td>
</tr>
<tr>
<td>groupingsDown</td>
<td>ReportGrouping[]</td>
<td>For Summary and Matrix reports: List that defines the fields by which you want to group and subtotal. For summary reports, choosing more than one sort field allows you to subsort your data. For matrix reports, specifies summary fields for column headings. When grouping by a date field, you can further group the data by a specific time period such as days, weeks, or months. Maximum for matrix reports: 2. Maximum for summary reports: 3</td>
</tr>
<tr>
<td>historicalSelector</td>
<td>ReportHistoricalSelector</td>
<td>Defines a date range for which historical trend reporting data is to be captured. Default is “Any Historical Date.”</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The report name. For example, Opportunity Pipeline</td>
</tr>
<tr>
<td>numSubscriptions</td>
<td>int</td>
<td>Indicates whether a user has subscribed to this report Lightning Experience (1) or not (0). Tied to user context.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>params</td>
<td>ReportParam[]</td>
<td>List that specifies settings specific to each report type, in particular options that let you filter a report to obtain useful subsets. For example, the Activities report type lets you specify whether you want to see open or closed activities or both and whether you want to see tasks or events or both. Valid values depend on the report type.</td>
</tr>
<tr>
<td>reportType</td>
<td>string</td>
<td>Required. Defines the type of data in the report. For example, Opportunity to create a report of opportunities data.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>roleHierarchyFilter</td>
<td>string</td>
<td>The role name for a report drill down. Some reports, such as opportunity and activity reports, display Hierarchy links that allow you to drill down to different data sets based on the role hierarchy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 17.0 and later.</td>
</tr>
<tr>
<td>rowLimit</td>
<td>int</td>
<td>Defines the maximum number of rows that can be returned for the report.</td>
</tr>
<tr>
<td>scope</td>
<td>string</td>
<td>Defines the scope of data on which you run the report. For example, whether you want to run the report against all opportunities, opportunities you own, or opportunities your team owns. Valid values depend on the reportType. For example, for Accounts reports:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MyAccounts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MyTeamsAccounts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AllAccounts</td>
</tr>
<tr>
<td>showCurrentDate</td>
<td>boolean</td>
<td>Can be set to true for historical trending reports in matrix format. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>showDetails</td>
<td>boolean</td>
<td>false shows a collapsed view of the report with only the headings, subtotals, and total. Default: true</td>
</tr>
<tr>
<td>showGrandTotal</td>
<td>boolean</td>
<td>true displays the calculated total for the full report.</td>
</tr>
<tr>
<td>showSubTotals</td>
<td>boolean</td>
<td>true displays the calculated subtotals for sections of the report.</td>
</tr>
<tr>
<td>sortColumn</td>
<td>string</td>
<td>Specifies the field on which to sort data in the report. Use sortOrder to specify the sort order.</td>
</tr>
<tr>
<td>sortOrder</td>
<td>SortOrder</td>
<td>Specifies the sort order. Use sortOrder to specify the field on which to sort.</td>
</tr>
<tr>
<td>territoryHierarchyFilter</td>
<td>string</td>
<td>The territory name for a report drill down. If your organization uses territory management, some reports display Hierarchy links that allow you to drill down to different data sets based on the territory hierarchy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 17.0 and later.</td>
</tr>
</tbody>
</table>
### Field Types

**Limits report results to records within a specified time frame.**

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeFrameFilter</td>
<td>ReportTimeFrameFilter</td>
<td>Limits report results to records within a specified time frame.</td>
</tr>
</tbody>
</table>

**The user name for a report drill down. Some reports, such as opportunity and activity reports, display Hierarchy links that allow you to drill down to different data sets based on the user hierarchy.**

This field is available in API version 17.0 and later.

### ReportAggregate

ReportAggregate defines custom summary formulas on summary, matrix, and joined reports. For more information on these fields, see “Add a Summary Formula Column to a Report” in the Salesforce online help.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acrossGroupingContext</td>
<td>string</td>
<td>Defines the row grouping level at which you want your custom summary formula to be displayed. This is a new field in API version 15.0.</td>
</tr>
<tr>
<td>calculatedFormula</td>
<td>string</td>
<td>Required. The custom summary formula. For example, <code>AMOUNT:SUM + OPP_QUANTITY:SUM</code></td>
</tr>
<tr>
<td>datatype</td>
<td>ReportAggregateDatatype</td>
<td>Required. Specifies the data type for formatting and display of the custom summary formula results.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. The internal development name of the custom summary formula, for example, <code>FORMULA1</code>. This is used to reference custom summary formulas from other report components, including conditional highlighting.</td>
</tr>
<tr>
<td>downGroupingContext</td>
<td>string</td>
<td>Defines the column grouping level at which you want your custom summary formula to be displayed. This field is available in API version 15.0 and later.</td>
</tr>
<tr>
<td>isActive</td>
<td>boolean</td>
<td>Required. <code>true</code> displays the formula result in the report. <code>false</code> does not display the result in the report.</td>
</tr>
<tr>
<td>isCrossBlock</td>
<td>boolean</td>
<td>Determines whether the custom summary formula is a cross-block formula, which is available with joined reports. <code>true</code> indicates a cross-block custom summary formula. <code>false</code> indicates a standard custom summary formula. This field is available in API version 25.0 and later.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The custom summary formula label (name).</td>
</tr>
</tbody>
</table>
ReportBlockInfo

ReportBlockInfo defines blocks in a joined report.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregateReferences</td>
<td>ReportAggregateReference[]</td>
<td>Lists the aggregates that represent the custom summary formulas used in a joined report block.</td>
</tr>
<tr>
<td>blockId</td>
<td>string</td>
<td>Required. blockId is used in cross-block custom summary formulas and joined report charts to identify the block containing each summary field. blockId is assigned automatically. Valid values are B1 through B5. This field is available in API version 25.0 and later.</td>
</tr>
<tr>
<td>joinTable</td>
<td>string</td>
<td>Required. Refers to the entity used to join blocks in a joined report. The entity provides a list of fields that are available for globally grouping across the blocks.</td>
</tr>
</tbody>
</table>

ReportAggregateReference

ReportAggregateReference defines the developer name used for custom summary formulas in joined reports.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregate</td>
<td>string</td>
<td>Required. The developerName of the ReportAggregate, which specifies the custom summary formula used in a block of a joined report.</td>
</tr>
</tbody>
</table>

ReportBucketField

ReportBucketField defines a bucket to be used in the report.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bucketType</td>
<td>ReportBucketFieldType (enumeration of type string)</td>
<td>Required. Specifies the type of bucket. Valid values: text, number, picklist</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. A unique name used as the <code>&lt;field&gt;</code> value to display a bucket field in the column list and other report components, including sort, filter, list, group, and chart. Must be of the format BucketField_name. For example, BucketField_BusinessSize.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The bucket field label. Maximum 40 characters. Any line breaks, tabs, or multiple spaces at the beginning or end of the label are removed. Any of these characters within the label are reduced to a single space.</td>
</tr>
<tr>
<td>nullTreatment</td>
<td>ReportBucketFieldNullTreatment (enumeration of type string)</td>
<td>For numeric bucket fields only. Specifies whether empty values are treated as zeros (z) or not (n).</td>
</tr>
<tr>
<td>otherBucketLabel</td>
<td>string</td>
<td>The label of the container for unbucketed values.</td>
</tr>
<tr>
<td>sourceColumnName</td>
<td>string</td>
<td>Required. The source field that the bucket is applied to. For example, SALES or INDUSTRY.</td>
</tr>
<tr>
<td>values</td>
<td>ReportBucketFieldValue (enumeration of type string)</td>
<td>Defines one bucket value used in the bucket field.</td>
</tr>
</tbody>
</table>

**Note:** While this name is plural, it represents a single bucket. In typical use, a bucket field contains multiple buckets.

---

### ReportBucketFieldValue

ReportBucketFieldValue defines a bucket value used in the bucket field.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceValues</td>
<td>ReportBucketFieldSourceValue (enumeration of type string)</td>
<td>The value of a bucket in the bucket field. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• sourceValue—Used for picklist and text bucket fields. For picklists, describes the picklist item in the bucket. For example, the sourceValue of a bucket on TYPE could be Customer. For text, the full string for the item in the bucket. For example, the sourceValue of a bucket on ADDRESS_STATE1 could be NY.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• from—Used only on numeric bucket fields. A non-inclusive lower bound for a numeric bucket range. This value must be a number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• to—Used only on numeric bucket fields. The inclusive upper bound for a numeric bucket range. This value must be a number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In numeric buckets, the first value must only have to and last value must only have from. All other values must have both to and from.</td>
</tr>
</tbody>
</table>
### Field Types and Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>string</td>
<td>Required. The name of a specific bucket value within the bucket field.</td>
</tr>
</tbody>
</table>

### ReportGrouping

ReportGrouping defines how to group, subtotal, and sort data for summary, matrix, and joined reports.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregateType</td>
<td>ReportAggrType (enumeration of type string)</td>
<td>The type of aggregate value to sort by. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Sum</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Average</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Maximum</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Minimum</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RowCount</td>
</tr>
<tr>
<td>dateGranularity</td>
<td>UserDateGranularity (enumeration of type string)</td>
<td>When grouping by a date field, the time period by which to group.</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>Required. The field by which you want to summarize data. For example, CAMPAIGN_SOURCE</td>
</tr>
<tr>
<td>sortByName</td>
<td>string</td>
<td>The API name of the column, aggregate or custom summary field used to order the grouping.</td>
</tr>
<tr>
<td>sortOrder</td>
<td>SortOrder</td>
<td>Required. Whether to sort data in ascending or descending alphabetical and numerical order.</td>
</tr>
<tr>
<td>sortType</td>
<td>ReportSortType (enumeration of type string)</td>
<td>Indicates if the grouping is sorted by a column, aggregate or custom summary field. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Column</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Aggregate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CustomSummaryFormula</td>
</tr>
</tbody>
</table>

### ReportHistoricalSelector

ReportHistoricalSelector defines a date range for historical data.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>snapshot</td>
<td>string</td>
<td>Represents the date value to apply a historical filter, either relative (in the format N_DAYS_AGO:2) or absolute (in the format yyyy-MM-dd). If unspecified, it’s assumed that the filter will be applied to all the columns the user sees. Available in API version 29.0 and later.</td>
</tr>
</tbody>
</table>
SortOrder
An enumeration of type string that defines the order in which data is sorted in the report fields. Valid values:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asc</td>
<td>Sorts data in ascending alphabetical and numerical order.</td>
</tr>
<tr>
<td>Desc</td>
<td>Sorts data in descending alphabetical and numerical order.</td>
</tr>
</tbody>
</table>

UserDateGranularity
An enumeration of type string that defines the time period by which to group data. Valid values:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>No grouping by date</td>
</tr>
<tr>
<td>Day</td>
<td>By day</td>
</tr>
<tr>
<td>Week</td>
<td>By week</td>
</tr>
<tr>
<td>Month</td>
<td>By month</td>
</tr>
<tr>
<td>Quarter</td>
<td>By quarter</td>
</tr>
<tr>
<td>Year</td>
<td>By year</td>
</tr>
<tr>
<td>FiscalQuarter</td>
<td>By fiscal quarter. You can set the fiscal year for your organization. See “Set the Fiscal Year” in the Salesforce online help.</td>
</tr>
<tr>
<td>FiscalYear</td>
<td>By fiscal year</td>
</tr>
<tr>
<td>MonthInYear</td>
<td>By calendar month in year</td>
</tr>
<tr>
<td>DayInMonth</td>
<td>By calendar day in month</td>
</tr>
<tr>
<td>FiscalPeriod</td>
<td>When custom fiscal years are enabled: By fiscal period</td>
</tr>
<tr>
<td>FiscalWeek</td>
<td>When custom fiscal years are enabled: By fiscal week</td>
</tr>
</tbody>
</table>

ReportSummaryType
An enumeration of type string that defines how report fields are summarized. Valid values:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum</td>
<td>Total</td>
</tr>
<tr>
<td>Average</td>
<td>Average</td>
</tr>
<tr>
<td>Maximum</td>
<td>Largest value</td>
</tr>
<tr>
<td>Minimum</td>
<td>Smallest value</td>
</tr>
</tbody>
</table>
ReportColorRange

ReportColorRange defines conditional highlighting for report summary data.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregate</td>
<td>ReportSummaryType (enumeration of type string)</td>
<td>Required. Defines how the field specified in columnName is summarized. For example, Sum.</td>
</tr>
<tr>
<td>columnName</td>
<td>string</td>
<td>Required. Specifies the field whose value ranges are represented by colors.</td>
</tr>
<tr>
<td>highBreakpoint</td>
<td>double</td>
<td>Required. Specifies the number that separates the mid color from the high color.</td>
</tr>
<tr>
<td>highColor</td>
<td>string</td>
<td>Required. Specifies the color (in HTML format) to represent data that falls into the high number range. This color spans from the highBreakpoint value.</td>
</tr>
<tr>
<td>lowBreakpoint</td>
<td>double</td>
<td>Required. Specifies the number that separates the low color from the mid color.</td>
</tr>
<tr>
<td>lowColor</td>
<td>string</td>
<td>Required. Specifies a color (in HTML format) to represent data that falls into the low value range, below the lowBreakpoint value.</td>
</tr>
<tr>
<td>midColor</td>
<td>string</td>
<td>Required. Specifies a color (in HTML format) to represent data that falls into the mid value range.</td>
</tr>
</tbody>
</table>

ReportColumn

ReportColumn defines how fields (columns) are displayed in the report.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregateTypes</td>
<td>ReportSummaryType[] (enumeration of type string)</td>
<td>List that defines if and how each report field is summarized.</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>Required. The field name. For example, AGE or OPPORTUNITY_NAME</td>
</tr>
<tr>
<td>reverseColors</td>
<td>boolean</td>
<td>In historical trend reports, displays greater Date values as green and greater Amount values as red, reversing the default colors. Available in API version 29.0 and later.</td>
</tr>
</tbody>
</table>
### showChanges

**Field Type**: boolean

In historical trend reports, adds a column displaying the difference between current and historical Date and Amount values.

Available in API version 29.0 and later.

### ReportFilter

ReportFilter limits the report results by filtering data on specified fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Specifies filter logic conditions.</td>
</tr>
<tr>
<td>criteriaItems</td>
<td>ReportFilterItem</td>
<td>The criteria by which you want to filter report data, either by comparing historical values or by applying a date range.</td>
</tr>
</tbody>
</table>

```xml
<criteriaItems>
  criteriaItems ReportFilterItem
  <column>Opportunity.Opportunity__hdsAmount__hst</column>
  <columnToColumn>false</columnToColumn>
  <operator>equals</operator>
  <snapshot>N_DAYS_AGO:90</snapshot>
  <value>100</value>
</criteriaItems>
```

<table>
<thead>
<tr>
<th>language</th>
<th>Language (enumeration of type string)</th>
<th>The language used when a report filters against a picklist value using the operators contains or startsWith. For a list of valid language values, see Language.</th>
</tr>
</thead>
</table>

### ReportFilterItem

ReportFilterItem limits the report results by filtering data on specified fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>column</td>
<td>string</td>
<td>Required. The field on which you want to filter data. For example, AMOUNT</td>
</tr>
<tr>
<td>columnToColumn</td>
<td>boolean</td>
<td>Indicates that the field contains data from a historical snapshot. Available in API version 29.0 and later.</td>
</tr>
<tr>
<td>isUnlocked</td>
<td>boolean</td>
<td>Optional. Indicates whether the report filter is unlocked (true) or locked (false). You can edit unlocked filters on the report</td>
</tr>
</tbody>
</table>
run page in Lightning Experience. If unspecified, the default value is false.
Available in API version 38.0 and later.

**operator**

FilterOperation (enumeration of type string)

Required. An enumeration of type string that defines the operator used to filter the data, for example, `greaterThan`. For valid values, see FilterOperation.

**snapshot**

string

Represents the date value, either relative (in the format `N_DAYS_AGO:2`) or absolute (in the format `yyyy-MM-dd`). Available in API version 29.0 and later.

**value**

string

The value by which you want to filter the data, for example, `1000`. Note that the Metadata API filter condition values do not always match those that you enter in the report wizard. For example, in the Metadata API dates are always converted to the US date format and values entered in a non-US English language may be converted to a standard US English equivalent.

### ReportFormat

An enumeration of type string that defines the report format. Valid values:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix</td>
<td>Summarizes data in a grid. Use to compare related totals.</td>
</tr>
<tr>
<td>Summary</td>
<td>Lists, sorts, and subtotals data.</td>
</tr>
<tr>
<td>Tabular</td>
<td>Lists data with no sorting or subtotals.</td>
</tr>
<tr>
<td>Joined</td>
<td>Joins data from different report types storing each report’s data in its own block.</td>
</tr>
</tbody>
</table>

### ReportParam

ReportParam represents settings specific to a report type, especially options that let you filter a report to certain useful subsets.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Specifies a specific <code>reportType</code> setting.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Required. The setting value.</td>
</tr>
</tbody>
</table>

### ReportAggregateDatatype

An enumeration of type string that specifies the data type for formatting and display of custom summary formula results. Valid values:
ReportChart
ReportChart represents charts on summary, matrix, and joined reports.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>backgroundColor1</td>
<td>string</td>
<td>Specifies the beginning color (in HTML format) for a gradient color background.</td>
</tr>
<tr>
<td>backgroundColor2</td>
<td>string</td>
<td>Specifies the end color (in HTML format) for a gradient color background.</td>
</tr>
</tbody>
</table>
| backgroundFadeDir  | ChartBackgroundDirection (enumeration of type string) | Specifies the direction for a gradient color background. Use with backgroundColor1 to specify the beginning color and backgroundColor2 to specify the end color for the gradient design. Use white for both if you do not want a background design. Valid values:  
  • Diagonal  
  • LeftToRight  
  • TopToBottom |
<p>| chartSummaries     | ChartSummary[]                    | Specifies the summaries you want to use for the chart. Invalid summaries are ignored without notification. If there are no valid summaries, RowCount is used by default for the axis value. This field is available in API version 17.0 and later. |
| chartType          | ChartType (enumeration of type string) | Required. Specifies the chart type. Available chart types depend on the report type. |
| enableHoverLabels  | boolean                           | Specifies whether to display values, labels, and percentages when hovering over charts. Hover details depend on chart type. Percentages apply to pie, donut, and funnel charts only. This field is available in API version 17.0 and later. |
| expandOthers       | boolean                           | Specifies whether to combine all groups less than or equal to 3% of the total into a single 'Others' wedge or segment. This only applies to pie, donut, and funnel charts. Set to true to show all values individually on the chart; set to false to combine small groups into 'Others.' This field is available in API version 17.0 and later. |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupingColumn</td>
<td>string</td>
<td>Specifies the field by which to group data. This data is displayed on the X-axis for vertical column charts and on the Y-axis for horizontal bar charts.</td>
</tr>
</tbody>
</table>
| legendPosition           | ChartLegendPosition (enumeration of type string) | Required. The location of the legend with respect to the chart. The valid values are:  
  • Bottom  
  • OnChart  
  • Right |
<p>| location                 | ChartPosition (enumeration of type string) | Required. Specifies whether the chart is displayed at the top or bottom of the report.                |
| secondaryGroupingColumn  | string                     | For grouped chart types: Specifies the field by which to group the data.                              |
| showAxisLabels           | boolean                    | For bar and line charts: Specifies whether the chart displays names for each axis.                    |
| showPercentage           | boolean                    | Indicates if percentages are displayed for wedges and segments of pie, donut, and funnel charts, as well as for gauges (true), or not (false). |
| showTotal                | boolean                    | Indicates if the total is displayed for donut charts and gauges (true), or not (false).              |
| showValues               | boolean                    | Indicates if the values of individual records or groups are displayed for charts (true), or not (false). |
| size                     | ReportChartSize (enumeration of type string) | Required. Specifies the chart size.                                                                  |
| summaryAggregate         | ReportSummaryType (enumeration of type string) | Defines how to summarize the chart data. For example, Sum. No longer supported in version API 17.0 and later. See chartSummaries. |
| summaryAxisManualRangeEnd| double                     | When specifying the axis range manually: Defines the ending value.                                   |
| summaryAxisManualRangeStart| double                   | When specifying the axis range manually: Defines the starting value.                                 |
| summaryAxisRange         | ChartRangeType (enumeration of type string) | Required. For bar, line, and column charts: Defines whether to specify the axis range manually or automatically. |
| summaryColumn            | string                     | Required. Specifies the field by which to summarize the chart data. Typically this field is displayed on the Y-axis. No longer supported in version API 17.0 and later. See chartSummaries. |
| textColor                | string                     | The color (in HTML format) of the chart text and labels.                                             |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>textSize</td>
<td>int</td>
<td>The size of the chart text and labels. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 36</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The maximum size is 18. Larger values are shown at 18 points.</td>
</tr>
<tr>
<td>title</td>
<td>string</td>
<td>The chart title. Max 255 characters.</td>
</tr>
<tr>
<td>titleColor</td>
<td>string</td>
<td>The color (in HTML format) of the title text.</td>
</tr>
<tr>
<td>titleSize</td>
<td>int</td>
<td>The size of the title text. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 36</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The maximum size is 18. Larger values are shown at 18 points.</td>
</tr>
</tbody>
</table>

**ChartType**

An enumeration of type string that defines the chart type. For information on each of these chart types, see “Chart Types” in the Salesforce online help. Valid values:

**Enumeration Value**

- None
- HorizontalBar
- HorizontalBarGrouped
- HorizontalBarStacked
- HorizontalBarStackedTo100
- VerticalColumn
**Enumeration Value**

VerticalColumnGrouped  
VerticalColumnStacked  
VerticalColumnStackedTo100  
Line  
LineGrouped  
LineCumulative  
LineCumulativeGrouped  
Pie  
Donut  
Funnel  
Scatter  
ScatterGrouped  
VerticalColumnLine  
VerticalColumnGroupedLine  
VerticalColumnStackedLine  
Plugin  
Reserved for future use. This value is available in API version 31.0 and later.

**ChartPosition**

An enumeration of type string that specifies the position of the chart in the report. Valid values:

**Enumeration Value**

CHART_TOP  
CHART_BOTTOM

**ChartSummary**

ChartSummary defines how data in the chart is summarized. Valid values:

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregate</td>
<td>ReportSummaryType</td>
<td>Specifies the aggregation method—such as Sum, Average, Min, and Max—for the summary value. Use the column field to specify the summary value to use for the aggregation. You</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>axisBinding</td>
<td>ChartAxis</td>
<td>Specifies the axis or axes to use on the chart. Use the column field to specify the summary value to use for the axis.</td>
</tr>
<tr>
<td>column</td>
<td>string</td>
<td>Required. Specifies the summary field for the chart data. If all columns are invalid, RowCount is used by default for the axis. For vertical column and horizontal bar combination charts, you can specify up to four values.</td>
</tr>
</tbody>
</table>

**ChartAxis**

An enumeration of type string that specifies the axis or axes to be used in charts. Valid values:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>The summary value to use for the X-axis of a scatter chart.</td>
</tr>
<tr>
<td>y</td>
<td>The Y-axis for the chart.</td>
</tr>
<tr>
<td>y2</td>
<td>The secondary Y-axis for vertical column combination charts with a line added.</td>
</tr>
</tbody>
</table>

**ReportChartSize**

An enumeration of type string that specifies the chart size. Valid values:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Tiny</td>
<td></td>
</tr>
<tr>
<td>Small</td>
<td></td>
</tr>
<tr>
<td>Medium</td>
<td></td>
</tr>
<tr>
<td>Large</td>
<td></td>
</tr>
<tr>
<td>Huge</td>
<td></td>
</tr>
</tbody>
</table>

**ChartRangeType**

An enumeration of type string that defines the report format. Valid values:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto</td>
<td></td>
</tr>
<tr>
<td>Manual</td>
<td></td>
</tr>
</tbody>
</table>
**ReportTimeFrameFilter**

ReportTimeFrameFilter represents the report time period.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateColumn</td>
<td>string</td>
<td>Required. The date field on which to filter data. For example, CLOSE_DATE</td>
</tr>
<tr>
<td>endDate</td>
<td>date</td>
<td>When interval is INTERVAL_CUSTOM, specifies the end of the custom time period.</td>
</tr>
<tr>
<td>interval</td>
<td>UserDateInterval (enumeration of type string)</td>
<td>Required. Specifies the period of time.</td>
</tr>
<tr>
<td>startDate</td>
<td>date</td>
<td>When interval is INTERVAL_CUSTOM, specifies the start of the custom time period.</td>
</tr>
</tbody>
</table>

**ReportCrossFilter**

ReportCrossFilter represents the cross filter functionality in reports.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteriaItems</td>
<td>ReportFilterItem</td>
<td>Represents the subfilters of a cross filter. There can be up to five subfilters. This field requires the following attributes.</td>
</tr>
<tr>
<td>operation</td>
<td>ObjectFilterOperator. Enumeration of type string</td>
<td>The action indicating whether to include or exclude an object. Valid values: with and without.</td>
</tr>
<tr>
<td>primaryTableColumn</td>
<td>string</td>
<td>The parent object used for the cross filter.</td>
</tr>
<tr>
<td>relatedTable</td>
<td>string</td>
<td>The child object used for the cross filter.</td>
</tr>
<tr>
<td>relatedTableJoinColumn</td>
<td>string</td>
<td>The field from the child object that is used to join the parent.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

A sample XML snippet using cross filters to build an Accounts report for cases where case status is not closed:

```xml
<crossFilters>
    <criteriaItems>
        <column>Status</column>
        <operator>notequal</operator>
        <value>Closed</value>
    </criteriaItems>
    <operation>with</operation>
    <primaryTableColumn>ACCOUNT_ID</primaryTableColumn>
</crossFilters>
```
UserDateInterval

An enumeration of type string that defines the period of time. Valid values:

<table>
<thead>
<tr>
<th>Enumeration Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTERVAL_CURRENT</td>
<td>Current fiscal quarter</td>
</tr>
<tr>
<td>INTERVAL_CURNEXT1</td>
<td>Current and next fiscal quarters</td>
</tr>
<tr>
<td>INTERVAL_CURPREV1</td>
<td>Current and previous fiscal quarters</td>
</tr>
<tr>
<td>INTERVAL_NEXT1</td>
<td>Next fiscal quarter</td>
</tr>
<tr>
<td>INTERVAL_PREV1</td>
<td>Previous fiscal quarter</td>
</tr>
<tr>
<td>INTERVAL_CURNEXT3</td>
<td>Current and next three fiscal quarters</td>
</tr>
<tr>
<td>INTERVAL_CURFY</td>
<td>Current fiscal year</td>
</tr>
<tr>
<td>INTERVAL_PREVFY</td>
<td>Previous fiscal year</td>
</tr>
<tr>
<td>INTERVAL_PREV2FY</td>
<td>Previous two fiscal years</td>
</tr>
<tr>
<td>INTERVAL_AGO2FY</td>
<td>Two fiscal years ago</td>
</tr>
<tr>
<td>INTERVAL_NEXTFY</td>
<td>Next fiscal year</td>
</tr>
<tr>
<td>INTERVAL_PREVCURFY</td>
<td>Current and previous fiscal years</td>
</tr>
<tr>
<td>INTERVAL_PREVCUR2FY</td>
<td>Current and previous two fiscal years</td>
</tr>
<tr>
<td>INTERVAL_CURNEXTFY</td>
<td>Current and next fiscal year</td>
</tr>
<tr>
<td>INTERVAL_CUSTOM</td>
<td>A custom time period. Use startDate and endDate fields to specify the time period's start date and end date.</td>
</tr>
<tr>
<td>INTERVAL_YESTERDAY</td>
<td>Yesterday</td>
</tr>
<tr>
<td>INTERVAL_TODAY</td>
<td>Today</td>
</tr>
<tr>
<td>INTERVAL_TOMORROW</td>
<td>Tomorrow</td>
</tr>
<tr>
<td>INTERVAL_LASTWEEK</td>
<td>Last calendar week</td>
</tr>
<tr>
<td>INTERVAL_THISWEEK</td>
<td>This calendar week</td>
</tr>
<tr>
<td>INTERVAL_NEXTWEEK</td>
<td>Next calendar week</td>
</tr>
<tr>
<td>INTERVAL_LASTMONTH</td>
<td>Last calendar month</td>
</tr>
<tr>
<td>INTERVAL_THISMONTH</td>
<td>This calendar month</td>
</tr>
<tr>
<td>Enumeration Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>INTERVAL_NEXTMONTH</td>
<td>Next calendar month</td>
</tr>
<tr>
<td>INTERVAL_LASTTHISMONTH</td>
<td>Current and previous calendar months</td>
</tr>
<tr>
<td>INTERVAL_THISNEXTMONTH</td>
<td>Current and next calendar months</td>
</tr>
<tr>
<td>INTERVAL_CURRENTQ</td>
<td>Current calendar quarter</td>
</tr>
<tr>
<td>INTERVAL_CURNEXTQ</td>
<td>Current and next calendar quarters</td>
</tr>
<tr>
<td>INTERVAL_CURPREVQ</td>
<td>Current and previous calendar quarters</td>
</tr>
<tr>
<td>INTERVAL_NEXTQ</td>
<td>Next calendar quarter</td>
</tr>
<tr>
<td>INTERVAL_PREVQ</td>
<td>Previous calendar quarter</td>
</tr>
<tr>
<td>INTERVAL_CURNEXT3Q</td>
<td>Current and next three calendar quarters</td>
</tr>
<tr>
<td>INTERVAL_CURY</td>
<td>Current calendar year</td>
</tr>
<tr>
<td>INTERVAL_PREVY</td>
<td>Previous calendar year</td>
</tr>
<tr>
<td>INTERVAL_PREV2Y</td>
<td>Previous two calendar years</td>
</tr>
<tr>
<td>INTERVAL_AGO2Y</td>
<td>Two calendar years ago</td>
</tr>
<tr>
<td>INTERVAL_NEXTY</td>
<td>Next calendar year</td>
</tr>
<tr>
<td>INTERVAL_PREVCURY</td>
<td>Current and previous calendar years</td>
</tr>
<tr>
<td>INTERVAL_PREVCUR2Y</td>
<td>Current and previous two calendar years</td>
</tr>
<tr>
<td>INTERVAL_CURNEXTY</td>
<td>Current and next calendar years</td>
</tr>
<tr>
<td>INTERVAL_LAST7</td>
<td>Last 7 days</td>
</tr>
<tr>
<td>INTERVAL_LAST30</td>
<td>Last 30 days</td>
</tr>
<tr>
<td>INTERVAL_LAST60</td>
<td>Last 60 days</td>
</tr>
<tr>
<td>INTERVAL_LAST90</td>
<td>Last 90 days</td>
</tr>
<tr>
<td>INTERVAL_LAST120</td>
<td>Last 120 days</td>
</tr>
<tr>
<td>INTERVAL_NEXT7</td>
<td>Next 7 days</td>
</tr>
<tr>
<td>INTERVAL_NEXT30</td>
<td>Next 30 days</td>
</tr>
<tr>
<td>INTERVAL_NEXT60</td>
<td>Next 60 days</td>
</tr>
<tr>
<td>INTERVAL_NEXT90</td>
<td>Next 90 days</td>
</tr>
<tr>
<td>INTERVAL_NEXT120</td>
<td>Next 120 days</td>
</tr>
<tr>
<td>LAST_FISCALWEEK</td>
<td>When custom fiscal years are enabled: Last fiscal week</td>
</tr>
<tr>
<td>THIS_FISCALWEEK</td>
<td>When custom fiscal years are enabled: This fiscal week</td>
</tr>
<tr>
<td>NEXT_FISCALWEEK</td>
<td>When custom fiscal years are enabled: Next fiscal week</td>
</tr>
<tr>
<td>Enumeration Value</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>LAST_FISCALPERIOD</td>
<td>When custom fiscal years are enabled: Last fiscal period</td>
</tr>
<tr>
<td>THIS_FISCALPERIOD</td>
<td>When custom fiscal years are enabled: This fiscal period</td>
</tr>
<tr>
<td>NEXT_FISCALPERIOD</td>
<td>When custom fiscal years are enabled: Next fiscal period</td>
</tr>
<tr>
<td>LASTTHIS_FISCALPERIOD</td>
<td>When custom fiscal years are enabled: This fiscal period and last fiscal period</td>
</tr>
<tr>
<td>THISNEXT_FISCALPERIOD</td>
<td>When custom fiscal years are enabled: This fiscal period and next fiscal period</td>
</tr>
<tr>
<td>CURRENT_ENTITLEMENT_PERIOD</td>
<td>Current entitlement period</td>
</tr>
<tr>
<td>PREVIOUS_ENTITLEMENT_PERIOD</td>
<td>Previous entitlement period</td>
</tr>
<tr>
<td>PREVIOUS_TWO_ENTITLEMENT_PERIODS</td>
<td>Previous two entitlement periods</td>
</tr>
<tr>
<td>TWO_ENTITLEMENT_PERIODS_AGO</td>
<td>Two entitlement periods ago</td>
</tr>
<tr>
<td>CURRENT_AND_PREVIOUS_ENTITLEMENT_PERIOD</td>
<td>Current and previous entitlement period</td>
</tr>
<tr>
<td>CURRENT_AND_PREVIOUS_TWO_ENTITLEMENT_PERIODS</td>
<td>Current and previous two entitlement periods</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

A sample XML report definition:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Report xmlns="http://soap.sforce.com/2006/04/metadata">
  <aggregates>
    <acrossGroupingContext>CRT_Object__c$Id</acrossGroupingContext>
    <calculatedFormula>PREVGROUPVAL(CRT_Object__c.Currency__c:AVG, CRT_Object__c.Id) *
    PARENTGROUPVAL(CRT_Object__c.Number__c:MAX, CRT_Object__c.CreatedBy.Name,
    COLUMN_GRAND_SUMMARY)/RowCount</calculatedFormula>
    <datatype>number</datatype>
    <developerName>FORMULA1</developerName>
    <downGroupingContext>CRT_Object__c$CreatedBy</downGroupingContext>
    <isActive>true</isActive>
    <masterLabel>CurrCSF</masterLabel>
    <scale>2</scale>
  </aggregates>
  <aggregates>
    <acrossGroupingContext>CRT_Object__c$LastModifiedDate</acrossGroupingContext>
    <calculatedFormula>IF(RowCount>10,
    BLANKVALUE(ROUND(PREVGROUPVAL(CRT_Object__c.Currency__c:SUM,
    CRT_Object__c.LastModifiedDate),3),3),
    PARENTGROUPVAL(CRT_Object__c.Number__c:SUM, ROW_GRAND_SUMMARY,
    CRT_Object__c.Id)), 1000)</calculatedFormula>
    <datatype>number</datatype>
    <developerName>FORMULA2</developerName>
    <downGroupingContext>GRAND_SUMMARY</downGroupingContext>
    <isActive>true</isActive>
    <masterLabel>numCSF</masterLabel>
</Report>
```
<aggregates>
<buckets>
<bucketType>number</bucketType>
<developerName>BucketField_BusinessSize</developerName>
<masterLabel>NumericBucket</masterLabel>
<nullTreatment>z</nullTreatment>
<sourceColumnName>SALES</sourceColumnName>
<values>
<sourceValues>
<to>10000</to>
</sourceValues>
<value>low</value>
</values>
<values>
<sourceValues>
<from>10000</from>
<to>25000</to>
</sourceValues>
<value>mid</value>
</values>
<values>
<sourceValues>
<from>25000</from>
</sourceValues>
<value>high</value>
</values>
</buckets>
<brackets>
<bucketType>text</bucketType>
<developerName>BucketField_Region</developerName>
<masterLabel>TextBucket</masterLabel>
<nullTreatment>n</nullTreatment>
<otherBucketLabel>Other</otherBucketLabel>
<sourceColumnName>ADDRESS1_STATE</sourceColumnName>
<values>
<sourceValues>
<sourceValue>CA</sourceValue>
</sourceValues>
<value>west</value>
</values>
<values>
<sourceValues>
<sourceValue>NY</sourceValue>
</sourceValues>
<sourceValues>
<sourceValue>Ontario</sourceValue>
</sourceValues>
<value>east</value>
</values>
</buckets>
<chart>
<backgroundColor1>#FFFFFF</backgroundColor1>
<backgroundColor2>#FFFFFF</backgroundColor2>
Declarative Metadata Sample Definition for a Joined Report

A sample XML report definition:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Report xmlns="http://soap.sforce.com/2006/04/metadata">
    <!-- This is a cross-block custom summary formula. Note that the calculated formula reference for a blocks reference uses the BlockId#Aggregate. -->
    <aggregates>
        <calculatedFormula>B1#AMOUNT:SUM+B2#EMPLOYEES:SUM</calculatedFormula>
        <datatype>number</datatype>
        <developerName>FORMULA</developerName>
        <isActive>true</isActive>
        <isCrossBlock>true</isCrossBlock>
        <masterLabel>Cross-Block CSF Example</masterLabel>
        <scale>2</scale>
    </aggregates>

    <!-- This is a standard custom summary formula. Note that the calculated formula reference does not have block reference but just the aggregate name of the report type associated (Opportunity). -->
    <aggregates>
        <calculatedFormula>AMOUNT:SUM</calculatedFormula>
        <developerName>FORMULA2</developerName>
        <isActive>true</isActive>
        <isCrossBlock>false</isCrossBlock>
        <masterLabel>Standard CSF Example</masterLabel>
        <reportType>Opportunity</reportType>
        <scale>2</scale>
    </aggregates>

    <block>
        <blockInfo>
            <!-- This is how the block defines that the custom summary formula should be referenced. In this example, it’s the in standard FORMULA 2 defined above. This block report has blockID B1. -->
        </blockInfo>
    </block>
</Report>
```
<aggregateReferences>
  <aggregate>FORMULA2</aggregate>
</aggregateReference>

<blockId>B1</blockId>
<joinTable>a</joinTable>
</blockInfo>
<columns>
  <field>TYPE</field>
</columns>
<format>Summary</format>
<name>Opportunities BLock 3</name>
<params>
  <name>role_territory</name>
  <value role"/>
</params>
<params>
  <name>terr</name>
  <value>all</value>
</params>
<params>
  <name>open</name>
  <value>all</value>
</params>
<params>
  <name>probability</name>
  <value>0</value>
</params>
<params>
  <name>co</name>
  <value>1</value>
</params>
<reportType>Opportunity</reportType>
<scope>organization</scope>
<timeFrameFilter>
  <dateColumn>CLOSE_DATE</dateColumn>
  <interval>INTERVAL_CUSTOM</interval>
</timeFrameFilter>
</block>
</block>
<!-- This is how the block defines that the custom summary formula should be referenced. In this example, it’s the cross-block custom summary formula FORMULA 1 defined above. This block report has blockId B2.-->
<aggregateReferences>
  <aggregate>FORMULA1</aggregate>
</aggregateReferences>
<blockId>B2</blockId>
<joinTable>a</joinTable>
</blockInfo>
<columns>
  <field>USERS_NAME</field>
</columns>
<columns>
  <field>TYPE</field>
</columns>
Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
   Dashboard

ReportType

Represents the metadata associated with a custom report type. Custom report types allow you to build a framework from which users can create and customize reports. For more information, see “Set Up a Custom Report Type” in the Salesforce online help.

This type extends the Metadata metadata type and inherits its fullName field.

Declarative Metadata File Suffix and Directory Location

The file suffix is .reportType for the custom report type definition. There is one file per custom report type. Report types are stored in the reportTypes directory of the corresponding package directory.

Version

Custom report types are available in API version 14.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>autogenerated</td>
<td>boolean</td>
<td>Indicates that the report type was automatically generated when historical trending was enabled for an entity. Available in API version 29 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>baseObject</td>
<td>string</td>
<td>Required. The primary object for the custom report type, for example, Account. All objects, including custom and external objects, are supported. You cannot edit this field after initial creation. Support for external objects is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>category</td>
<td>ReportTypeCategory (enumeration of type string)</td>
<td>Required. This field controls the category for the report. The valid values are: accounts opportunities forecasts cases leads campaigns activities busop products admin territory territory2 (This value is available in API version 31.0 and later.) usage_entitlement wdc (This value is available in API version 29.0 and later.) calibration (This value is available in API version 29.0 and later.) other content quotes individual (This value is available in API version 45.0 and later.)</td>
</tr>
<tr>
<td>deployed</td>
<td>boolean</td>
<td>Required. Indicates whether the report type is available to users (true) or whether it’s still in development (false).</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the custom report type.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The report type developer name used as a unique identifier for API access. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.</td>
</tr>
<tr>
<td>join</td>
<td>ObjectRelationship</td>
<td>The object joined to the baseObject. For example, Contacts may be joined to the primary Accounts object.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The report type label.</td>
</tr>
<tr>
<td>sections</td>
<td>ReportLayoutSection[]</td>
<td>The groups of columns available for the report type. Though columns are not strictly required, a report without columns is not very useful.</td>
</tr>
</tbody>
</table>
**ObjectRelationship**

ObjectRelationship represents a join to another object. For more information, see "Add Child Objects To Your Custom Report Type" in the Salesforce online help.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>join</td>
<td>ObjectRelationship</td>
<td>This field is a recursive reference that allows you to join more than two objects. A maximum of four objects can be joined in a custom report type. When more than two objects are joined, an inner join is not allowed if there has been an outer join earlier in the join sequence. The baseObject is first joined to the object specified in relationship; the resulting data set is then joined with any objects specified in this field.</td>
</tr>
<tr>
<td>outerJoin</td>
<td>boolean</td>
<td>Required. Indicates whether this is an outer join (true) or not (false). An outer join returns a row even if the joined table does not contain a matching value in the join column.</td>
</tr>
<tr>
<td>relationship</td>
<td>string</td>
<td>Required. The object joined to the primary object; for example, Contacts.</td>
</tr>
</tbody>
</table>

**ReportLayoutSection**

ReportLayoutSection represents a group of columns used in the custom report type.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns</td>
<td>ReportTypeColumn</td>
<td>The list of columns projected from the query, defined by this custom report type.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The label for this group of columns in the report wizard.</td>
</tr>
</tbody>
</table>

**ReportTypeColumn**

ReportTypeColumn represents a column in the custom report type.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkedByDefault</td>
<td>boolean</td>
<td>Required. Indicates whether this column is selected be default (true) or not (false).</td>
</tr>
<tr>
<td>displayNameOverride</td>
<td>string</td>
<td>A customized column name, if desired.</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>Required. The field name associated with the report column.</td>
</tr>
<tr>
<td>table</td>
<td>string</td>
<td>Required. The table associated with the field; for example, Account.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The definition of a custom report type is shown below. Account is joined to Contacts and the resulting data set is joined with Assets.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ReportType xmlns="http://soap.sforce.com/2006/04/metadata">
    <baseObject>Account</baseObject>
    <category>accounts</category>
    <deployed>true</deployed>
    <description>Account linked to Contacts and Assets</description>
    <join>
        <join>
            <outerJoin>false</outerJoin>
            <relationship>Assets</relationship>
        </join>
        <outerJoin>false</outerJoin>
        <relationship>Contacts</relationship>
    </join>
    <label>Account Contacts and Assets</label>
    <sections>
        <columns>
            <checkedByDefault>true</checkedByDefault>
            <field>obj_lookup__c.Id</field>
            <table>Account</table>
        </columns>
        <columns>
            <checkedByDefault>false</checkedByDefault>
            <field>obj_lookup__c.Name</field>
            <table>Account</table>
        </columns>
        <columns>
            <checkedByDefault>false</checkedByDefault>
            <field>Opportunity__c.Amount</field>
            <table>Account</table>
        </columns>
        <columns>
            <checkedByDefault>false</checkedByDefault>
            <field>Owner.IsActive</field>
            <table>Account</table>
        </columns>
        <masterLabel>Accounts</masterLabel>
    </sections>
    <sections>
        <columns>
            <checkedByDefault>false</checkedByDefault>
            <field>Owner.Email</field>
            <table>Account.Contacts</table>
        </columns>
        <columns>
            <checkedByDefault>false</checkedByDefault>
            <field>byr__c</field>
            <table>Account.Contacts</table>
        </columns>
    </sections>
</ReportType>
```
Usage

The custom report type refers to fields by using their API names. For a historical field (one that has `trackTrending` set to `true`) the API name includes `hst`, such as `Field2__c_hst`.

For more information, see `trackTrending` on page 318.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Role

Represents a role in your organization.

Declarative Metadata File Suffix and Directory Location

The file suffix for role components is `.role` and components are stored in the `roles` directory of the corresponding package directory.

Version

Role components are available in API version 24.0 and later.

Fields

This metadata type extends to subtype `RoleOrTerritory` on page 737.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fullName</td>
<td>string</td>
<td>The unique identifier for API access. The <code>fullName</code> can contain only underscores and alphanumeric characters. It must be unique, begin with</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is the definition of a role.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Role xmlns="http://soap.sforce.com/2006/04/metadata">
  <caseAccessLevel>Edit</caseAccessLevel>
  <contactAccessLevel>Edit</contactAccessLevel>
  <description>Sample Role</description>
  <mayForecastManagerShare>false</mayForecastManagerShare>
  <name>R22</name>
  <opportunityAccessLevel>Read</opportunityAccessLevel>
</Role>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

RoleOrTerritory

Represents the common base type and valid values for role or territory.

Version

RoleOrTerritory components are available in API version 24.0 and later.

Note: You can’t create a RoleOrTerritory component directly. Use the Role or Territory metadata types instead.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| caseAccessLevel | string     | Specifies whether a user can access other users’ cases that are associated with accounts the user owns. Valid values are:  
|                 |            | • Read  
|                 |            | • Edit  
|                 |            | • None  |
**Field Name** | **Field Type** | **Description**
--- | --- | ---
| | | This field is not visible if your organization’s sharing model for cases is Public Read/Write. If no value is set for this field, this field value uses the default access level that is specified in the Manage Territory page in Setup.

| contactAccessLevel | string | Specifies whether a user can access other users’ contacts that are associated with accounts the user owns. Valid values are:
| | | - Read
| | | - Edit
| | | - None

This field is not visible if your organization’s sharing model for contacts is Public Read/Write or Controlled by Parent. If no value is set for this field, this field value uses the default access level that is specified in the Manage Territory page in Setup.

| description | string | The description of the role or territory.
| | | The unique identifier for API access. The `fullName` can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the **Metadata** component.

| mayForecastManagerShare | boolean | Indicates whether the forecast manager can manually share their own forecast.
| | | Required. The name of the role or territory.

| opportunityAccessLevel | string | Specifies whether a user can access other users’ opportunities that are associated with accounts the user owns. Valid values are:
| | | - Read
| | | - Edit
| | | - None

This field is not visible if your organization’s sharing model for opportunities is Public Read/Write. If no value is set for this field, this field value uses the default access level that is specified in the Manage Territory page in Setup.

---

**Declarative Metadata Sample Definition**

The following is the definition of a role.

```
<?xml version="1.0" encoding="UTF-8"?>
<Role xmlns="http://soap.sforce.com/2006/04/metadata">
    <caseAccessLevel>Edit</caseAccessLevel>
</Role>
```
The following is the definition of a territory.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Territory xmlns="http://soap.sforce.com/2006/04/metadata">
  <accountAccessLevel>Edit</accountAccessLevel>
  <caseAccessLevel>Edit</caseAccessLevel>
  <contactAccessLevel>Edit</contactAccessLevel>
  <description>Sample Territory</description>
  <mayForecastManagerShare>false</mayForecastManagerShare>
  <name>T22name</name>
  <opportunityAccessLevel>Read</opportunityAccessLevel>
</Territory>
```

SEE ALSO:
- Role
- Territory

**SamlSsoConfig**

Represents a SAML Single Sign-On configuration. This type extends the Metadata metadata type and inherits its fullName field. Single sign-on (SSO) lets users access authorized network resources with one login. You validate usernames and passwords against your corporate user database or other client app rather than Salesforce managing separate passwords for each resource.

**File Suffix and Directory Location**

SamlSsoConfig components have the suffix .samlssoconfig and are stored in the samlssoconfigs folder.

**Version**

SamlSsoConfig components are available in API version 28.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributeName</td>
<td>string</td>
<td>The name of the identity provider's application. Get this from your identity provider.</td>
</tr>
<tr>
<td>attributeNameIdFormat</td>
<td>string</td>
<td>For SAML 2.0 only and when identityLocation is set to Attribute. Possible values include unspecified, emailAddress or persistent. All legal values can be found in</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>decryptionCertificate</td>
<td>string</td>
<td>The name of the certificate to use for decrypting incoming SAML assertions. This certificate is saved in the organization's Certificate and Key Management list. Available in API version 30.0 and later.</td>
</tr>
<tr>
<td>errorUrl</td>
<td>string</td>
<td>The URL of the page users should be directed to if there's an error during SAML login. It must be a publicly accessible page, such as a public site Visualforce page. The URL can be absolute or relative.</td>
</tr>
<tr>
<td>executionUserId</td>
<td>string</td>
<td>The user that runs the Apex handler class. The user must have the &quot;Manage Users&quot; permission. A user is required if you specify a SAML JIT handler class.</td>
</tr>
</tbody>
</table>
| identityLocation      | SamlIdentityLocationType (enumeration of type string) | The location in the assertion where a user should be identified. Valid values are:  
- **SubjectNameId** — The identity is in the `<Subject>` statement of the assertion.  
- **Attribute** — The identity is specified in an `<AttributeValue>`, located in the `<Attribute>` of the assertion. |
| identityMapping       | SamlIdentityType (enumeration of type string) | The identifier the service provider uses for the user during Just-in-Time user provisioning. Valid values are:  
- **Username** — The user’s Salesforce username.  
- **FederationId** — The federation ID from the user object; the identifier used by the service provider for the user.  
- **UserId** — The user ID from the user’s Salesforce organization. |
| issuer                | string              | The identification string for the Identity Provider.                                                                                                                                                 |
| loginUrl              | string              | For SAML 2.0 only: The URL where Salesforce sends a SAML request to start the login sequence.                                                                                                           |
| logoutUrl             | string              | For SAML 2.0 only: The URL to direct the user to when they click the Logout link. The default is [http://www.salesforce.com](http://www.salesforce.com).                                                      |
| name                  | string              | The unique name used by the API and managed packages. The name must begin with a letter and use only alphanumeric characters and underscores. The name cannot end with an underscore or have two consecutive underscores. |
| oauthTokenEndpoint    | string              | For SAML 2.0 only: The ACS URL used with enabling Salesforce as an identity provider in the Web single sign-on OAuth assertion flow.                                                                  |
| redirectBinding       | boolean             | If you’re using My Domain, chose the binding mechanism your identity provider requests for your SAML messages. Values are:  
- **HTTP POST** — HTTP POST binding sends SAML messages using base64-encoded HTML forms. |

For SAML 2.0 only: The ACS URL used with enabling Salesforce as an identity provider in the Web single sign-on OAuth assertion flow.

If you’re using My Domain, chose the binding mechanism your identity provider requests for your SAML messages. Values are:

- **HTTP POST** — HTTP POST binding sends SAML messages using base64-encoded HTML forms.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestSignatureMethod</td>
<td>string</td>
<td>The method that's used to sign the SAML request. Valid values are RSA-SHA1 and RSA-SHA256.</td>
</tr>
<tr>
<td>requestSigningCertId</td>
<td>string</td>
<td>The 18 digit ID for the certificate used to generate the signature on a SAML request to the identity provider. The certificate is saved in the Certificate and Key Management page in Setup.</td>
</tr>
<tr>
<td>salesforceLoginUrl</td>
<td>string</td>
<td>The URL associated with login for the Web single sign-on flow.</td>
</tr>
<tr>
<td>samlEntityId</td>
<td>string</td>
<td>The issuer in SAML requests generated by Salesforce, and is also the expected audience of any inbound SAML Responses. If you don’t have domains deployed, this value is always <a href="https://saml.salesforce.com">https://saml.salesforce.com</a>. If you have domains deployed, Salesforce recommends that you use your custom domain name.</td>
</tr>
<tr>
<td>samlJitHandlerId</td>
<td>string</td>
<td>The name of an existing Apex class that implements the Auth.SamlJitHandler interface.</td>
</tr>
<tr>
<td>samlVersion</td>
<td>SamlType (enumeration of type string)</td>
<td>The SAML version in use. Valid values are:</td>
</tr>
<tr>
<td>singleLogoutBinding</td>
<td>SamlSpSLOBinding (enumeration of type string)</td>
<td>The HTTP binding type. This value determines where to put the LogoutRequest or LogoutResponse in the SAML request during single logout (SLO). The value is base64 encoded. Valid values are:</td>
</tr>
<tr>
<td>singleLogoutUrl</td>
<td>string</td>
<td>The SAML single logout endpoint. This URL is the endpoint where Salesforce sends LogoutRequests (when Salesforce initiates a logout), or LogoutResponses (when the identity provider initiates a logout).</td>
</tr>
<tr>
<td>userProvisioning</td>
<td>boolean</td>
<td>If true, Just-in-Time user provisioning is enabled, which creates users on the fly the first time they try to log in. Specify Federation ID for the identityMapping value to use this feature.</td>
</tr>
<tr>
<td>validationCert</td>
<td>string</td>
<td>The certificate used to validate the request. Get this from your identity provider.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example of a SamlSsoConfig component. The validation certificate string has been truncated for readability.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SamlSsoConfig xmlns="http://soap.sforce.com/2006/04/metadata">
  <identityLocation>SubjectNameId</identityLocation>
</SamlSsoConfig>```
<identityMapping>FederationId</identityMapping>
<issuer>https://my-idp.my.salesforce.com</issuer>
<loginUrl>
  https://my-idp.my.salesforce.com/idp/endpoint/HttpRedirect
</loginUrl>
<logoutUrl>https://www.salesforce.com</logoutUrl>
<name>SomeCompany</name>
<oauthTokenEndpoint>
  https://login.salesforce.com/services/oauth2/token?so=00DD0000000JxeI
</oauthTokenEndpoint>
<redirectBinding>true</redirectBinding>
$requestSignatureMethod>RSA-SHA1</requestSignatureMethod>
<salesforceLoginUrl>
  https://login.salesforce.com?so=00DD0000000JxeI
</salesforceLoginUrl>
<samlEntityId>
  https://saml.salesforce.com/customPath
</samlEntityId>
<samlVersion>SAML2_0</samlVersion>
$userProvisioning>false</userProvisioning>
/validationCert>
  MIIEojCCA4gAwIBAgIIOATtxsoBFAAAAAD4...
</validationCert>
</SamlSsoConfig>

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Scontrol

Deprecated. Represents an Scontrol component, corresponding to an s-control in the Salesforce user interface.

⚠️ Important: Visualforce pages supersede s-controls. Organizations that haven't previously used s-controls can't create them. Existing s-controls are unaffected, and can still be edited.

For more information, see "About S-Controls" in the Salesforce online help. This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.

Declarative Metadata File Suffix and Directory Location

The file suffix is .scf for the s-control file. The accompanying metadata file is named ScontrolName-meta.xml.

Scontrol components are stored in the scontrols folder in the corresponding package directory.

Version

Scontrols are available in API version 10.0 and later.
## Fields

This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
<td>base64Binary</td>
<td>Content of the s-control. Base 64-encoded binary data. Prior to making an API call, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to binary. This conversion is usually handled for you by a SOAP client. This field is inherited from the MetadataWithContent component.</td>
</tr>
</tbody>
</table>
| contentSource  | SControlContentSource (enumeration of type string) | Required. Determines how you plan to use the s-control:  
  - **HTML**: Select this option if you want to enter the content for your s-control in content.  
  - **URL**: Select this option if you want to enter the link or URL of an external website in content.  
  - **Snippet**: Snippets are s-controls that are designed to be included in other s-controls. Select this option if you want to enter the content for your s-control snippet in content. |
| description    | string                          | Optional text that describes the s-control. This only displays to users with “View All Data” permission (administrator).                                                                                       |
| encodingKey    | Encoding (enumeration of type string) | Required. The default encoding setting is Unicode: UTF-8. Change it if you are passing information to a URL that requires data in a different format. This option is available when you select URL as the value for contentSource. |
| fileContent    | base64                           | File contents displayed if you add this s-control to a custom link. The file can contain a Java applet, Active-X control, or any other type of content you want. This option only applies to s-controls with a value of HTML for contentSource. |
| fileName       | string                          | The unique name for the s-control. This name can contain only underscores and alphanumeric characters, and must be unique in your org. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field cannot be changed for components installed by a managed package. It is only relevant if the fileContent field also has a value. This is a new field in API version 14.0. |
| fullName       | string                          | The s-control developer name used as a unique identifier for API access. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. If this field contained characters before version 14.0 that are no longer allowed, the characters were stripped out of this field, and the previous value of the field was saved in the name field. This field is inherited from the Metadata component. |
### Declaration Metadata Sample Definition

The following sample creates the `Myriad_Publishing.scf` s-control, which creates a link to the website specified in the s-control. The corresponding `Myriad_Publishing.scf-meta.xml` metadata file follows the s-control file.

**Myriad_Publishing.scf file:**

```
http://www.myriadpubs.com
```

**Myriad_Publishing.scf-meta.xml:**

```
<?xml version="1.0" encoding="UTF-8"?>
<Scontrol xmlns="http://soap.sforce.com/2006/04/metadata">
  <contentSource>URL</contentSource>
  <description>s-control to open Myriad Publishing website.</description>
  <encodingKey>UTF-8</encodingKey>
  <name>Myriad Publishing</name>
  <supportsCaching>true</supportsCaching>
</Scontrol>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### ServiceChannel

Represents a channel of work items that are received from your organization—for example, cases, chats, or leads. This type extends the Metadata metadata type and inherits its fullName field.

### File Suffix and Directory Location

ServiceChannel components have the suffix .serviceChannel and are stored in the serviceChannels folder.

### Version

ServiceChannel components are available in API version 44.0 and later.
Special Access Rules
This type is available only if Omni-Channel is enabled in your org.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interactionComponent</td>
<td>string</td>
<td>The custom console component to open in the footer when an agent accepts a work item from this service channel.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The label of the service channel.</td>
</tr>
<tr>
<td>relatedEntityType</td>
<td>string</td>
<td>Required. The type of object that’s associated with this service channel.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a ServiceChannel component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceChannel xmlns="http://soap.sforce.com/2006/04/metadata">
  <interactionComponent>component</interactionComponent>
  <label>Case</label>
  <relatedEntityTypes>Case</relatedEntityTypes>
</ServiceChannel>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>ServiceChannel</name>
  </types>
  <version>44.0</version>
</Package>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

ServicePresenceStatus

Represents a presence status that can be assigned to a service channel. This type extends the Metadata metadata type and inherits its fullName field.
File Suffix and Directory Location

ServicePresenceStatus components have the suffix `.servicePresenceStatus` and are stored in the `servicePresenceStatuses` folder.

Version

ServicePresenceStatus components are available in API version 44.0 and later.

Special Access Rules

This type is available only if Omni-Channel is enabled in your org.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channels</td>
<td>ServiceChannelStatus</td>
<td>Represents the status that’s associated with a specific service channel.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label of the presence status.</td>
</tr>
</tbody>
</table>

ServiceChannelStatus

Represents the status that’s associated with a specific service channel.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>string</td>
<td>Represents the channels assigned to the presence status.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a ServicePresenceStatus component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServicePresenceStatus xmlns="http://soap.sforce.com/2006/04/metadata">
    <channels>
        <channel>Case</channel>
    </channels>
    <label>Available for Cases</label>
</Service PresenceStatus>
```

The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
.Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>*</members>
        <name>ServicePresenceStatus</name>
    </types>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Settings

Represents the organization settings related to a feature. For example, your password policies, session settings and network access controls are all available in the SecuritySettings component type.

Not all feature settings are available in the Metadata API. See Unsupported Metadata Types on page 122 for information on which feature settings are not available.

Settings can be accessed using the specific component member or via wildcard. For example, in the package manifest file you would use the following section to access SecuritySettings:

```xml
<types>
    <members>Security</members>
    <name>Settings</name>
</types>
```

The member format when used in the package manifest is the component metadata type name without the “Settings” suffix, so in the preceding example “Security” is used instead of “SecuritySettings”.

File Suffix and Directory Location

Each settings component gets stored in a single file in the settings directory of the corresponding package directory. The filename uses the format Setting feature.settings. For example, the SecuritySettings file would be Security.settings. See “File Suffix and Directory Location” information for the individual settings components to determine the exact filename.

Version

Settings is available in API version 27.0 and later. See the version information for the individual setting component to determine which API version the settings component became available.

Declarative Metadata Sample Definition

The following is an example package manifest used to deploy or retrieve only the MobileSettings for an organization:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>Mobile</members>
        <name>Settings</name>
    </types>
    <version>27.0</version>
</Package>
```
The following is an example package manifest used to deploy or retrieve all the available settings metadata for an organization, using a wildcard:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>*</members>
        <name>Settings</name>
    </types>
    <version>27.0</version>
</Package>
```

**SEE ALSO:**
- AccountSettings
- ActivitiesSettings
- AddressSettings
- CaseSettings
- ChatterAnswersSettings
- CompanySettings
- ContractSettings
- EntitlementSettings
- ForecastingSettings
- IdeasSettings
- KnowledgeSettings
- MobileSettings
- SecuritySettings

**AccountSettings**

Represents an organization’s account settings for account teams, account owner report, and the View Hierarchy link. This type extends the Metadata metadata type and inherits its fullName field.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

**File Suffix and Directory Location**

AccountSettings values are stored in the Account.settings file in the settings folder. The .settings files are different from other named components because there is only one settings file for each settings component.

**Version**

AccountSettings is available in API versions 29.0 and later.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableAccountOwnerReport</td>
<td>boolean</td>
<td>Indicates whether Account Owner Report may (true) or may not (false) be run by all users.</td>
</tr>
<tr>
<td>enableAccountTeams</td>
<td>boolean</td>
<td>Indicates whether Account Teams are enabled (true) or not (false). The Metadata API can’t be used to disable Account Teams.</td>
</tr>
<tr>
<td>enableContactHistoryTracking</td>
<td>boolean</td>
<td>Indicates whether Enable Contact History for Contact objects is enabled (true) or not (false). Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>showViewHierarchyLink</td>
<td>boolean</td>
<td>Indicates whether the default View Hierarchy link on all business account detail pages is visible (true) or hidden (false).</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of the Account.settings file:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<AccountSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableAccountOwnerReport>true</enableAccountOwnerReport>
  <enableAccountTeams>true</enableAccountTeams>
  <enableContactHistoryTracking>true</enableContactHistoryTracking>
  <showViewHierarchyLink>true</showViewHierarchyLink>
</AccountSettings>
```

Example Package Manifest

The following is an example package manifest used to deploy or retrieve the Account settings metadata for an organization:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Account</members>
    <name>Settings</name>
  </types>
  <version>29.0</version>
</Package>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:

- Settings
### ActivitiesSettings

Represents an organization's activity settings, and its user interface settings for the calendar. Use the `ActivitiesSettings` component type to control the following activity settings:

- Configure group and recurring tasks, recurring and multiday events, and email tracking
- Relate multiple contacts to tasks and events (shared activities)
- Display custom logos in meeting requests

Also use the `ActivitiesSettings` component type to control user interface settings for the calendar, including hover links and drag-and-drop editing.

In the package manifest, all organization settings metadata types are accessed using the "Settings" name. See Settings for more details.

### File Suffix and Directory Location

ActivitiesSettings values are stored in the `Activities.settings` file in the `settings` directory. The `.settings` files are different from other named components because there is only one settings file for each settings component.

### Version

ActivitiesSettings is available in API versions 28.0 and later.

### Fields

Settings for all types listed below are controlled on the Activity settings page or the User Interface settings page as noted.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowUsersToRelateMultipleContactsToTasksAndEvents</td>
<td>boolean</td>
<td>This read only field indicates whether Shared Activities is enabled. When the value is true, allows users to relate multiple contacts to a task or event.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Important</strong>: Beginning with API v36.0, this field is read-only in all versions of the API. You can’t change the value of this field. Even though this field was updateable before Spring '16, changing this field’s value wasn’t supported and could have resulted in an incorrect integration. If you have code in older API versions that changes the value of this field, ensure you update that code to prevent any errors.</td>
</tr>
<tr>
<td>autoRelateEventAttendees</td>
<td>boolean</td>
<td>When users add attendees to events, events are automatically related to up to 50 contacts or one lead. An attendee is matched by their email address to a contact or lead. Admins control this field on the Activity Settings page. Available in API version 42.0 and later.</td>
</tr>
<tr>
<td>enableActivityReminders</td>
<td>boolean</td>
<td>Enables popup activity reminders for an organization. Admins control this field on the Activity Settings page.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>enableClickCreateEvents</td>
<td>boolean</td>
<td>Lets users create events in day and weekly calendar views by double-clicking a specific time slot and entering the details of the event in an overlay. Hovering over an event displays an overlay where users can view the event details or delete the event without leaving the page. Admins use a mini page layout to configure the fields shown in the overlays. Does not support recurring events or multi-person events. Admins control this field on the User Interface settings page.</td>
</tr>
<tr>
<td>enableDragAndDropScheduling</td>
<td>boolean</td>
<td>Lets users create events associated with records by dragging a record from a list view onto a calendar view and entering the details of the event in an overlay. Hovering over an event displays an overlay where users can view the event details or delete the event without leaving the page. Admins use a mini page layout to configure the fields shown in the overlays. Admins control this field on the User Interface settings page.</td>
</tr>
<tr>
<td>enableEmailTracking</td>
<td>boolean</td>
<td>Enables tracking of outbound HTML emails if an organization uses HTML email templates. Admins control this field on the Activity settings page.</td>
</tr>
<tr>
<td>enableGroupTasks</td>
<td>boolean</td>
<td>Lets users assign independent copies of a new task to multiple users. Admins control this field on the Activity settings page.</td>
</tr>
<tr>
<td>enableListViewScheduling</td>
<td>boolean</td>
<td>Extends the functionality of enableDragAndDropScheduling and enableClickCreateEvents to list view calendars. Admins control this field on the User Interface settings page.</td>
</tr>
<tr>
<td>enableMultidayEvents</td>
<td>boolean</td>
<td>Enables creation of events that end more than 24 hours after they start. Admins control this field on the Activity settings page.</td>
</tr>
<tr>
<td>enableRecurringEvents</td>
<td>boolean</td>
<td>Enables creation of events that repeat at specified intervals. Admins control this field on the Activity settings page.</td>
</tr>
<tr>
<td>enableRecurringTasks</td>
<td>boolean</td>
<td>Enables creation of tasks that repeat at specified intervals. Admins control this field on the Activity settings page.</td>
</tr>
<tr>
<td>enableSidebarCalendarShortcut</td>
<td>boolean</td>
<td>In the sidebar, displays a shortcut link to a user’s last-used calendar view. Admins control this field on the Activity settings page.</td>
</tr>
<tr>
<td>enableSimpleTaskCreateUI</td>
<td>boolean</td>
<td>Allows Admins to specify whether tapping New Task in Salesforce opens a regular task record edit page or a page that displays key task fields first. Admins control this field on the Activity settings page.</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableUndelegatedTaskNotifications</td>
<td>boolean</td>
<td>On the Activity settings page, exposes a setting for Admins to hide or show a user setting that lets individual users enable or disable email notifications when tasks are assigned to them.</td>
</tr>
<tr>
<td>meetingRequestsLogo</td>
<td>string</td>
<td>Available when <code>showCustomLogoMeetingRequests</code> is enabled. Uploads a custom logo. An administrator can select only a logo that has been uploaded to certain folders in the Documents tab. Admins control this field on the Activity settings page.</td>
</tr>
<tr>
<td>showCustomLogoMeetingRequests</td>
<td>boolean</td>
<td>Displays a custom logo in meeting request emails and on a meeting’s Web page. Invitees see the logo when a user either invites them to an event or requests a meeting. Admins control this field on the Activity settings page.</td>
</tr>
<tr>
<td>showEventDetailsMultiUserCalendar</td>
<td>boolean</td>
<td>Displays event details on-screen rather than in hover text. Admins control this field on the Activity settings page.</td>
</tr>
<tr>
<td>showHomePageHoverLinksForEvents</td>
<td>boolean</td>
<td>In the calendar section of the Home tab:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• When a user hovers over the subject of an event, a hover link displays an overlay with selected event details. (Hover links are always available in other calendar views.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• When a user clicks the subject of an event, displays the event detail page. Admins use a mini page layout to configure the fields shown in the overlay.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Admins control this field on the User Interface settings page.</td>
</tr>
<tr>
<td>showMyTasksHoverLinks</td>
<td>boolean</td>
<td>In the My Tasks section of the Home tab and on the calendar day view:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• When a user hovers over the subject of a task, a hover link displays an overlay with selected task details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• When a user clicks the subject of a task, displays the task detail page. Admins use a mini page layout to configure the fields shown in the overlay.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Admins control this field on the User Interface settings page.</td>
</tr>
</tbody>
</table>

**Example Package Manifest**

The following is an example package manifest used to deploy or retrieve the Activity settings metadata for an organization:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Activities</members>
    <name>Settings</name>
</types>
```
Declarative Metadata Sample Definition

The following is an example of an activity settings file:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ActivitiesSettings xmlns="http://soap.sforce.com/2006/04/metadata">
    <enableActivityReminders>true</enableActivityReminders>
    <autoRelateEventAttendees>true</autoRelateEventAttendees>
    <enableClickCreateEvents>true</enableClickCreateEvents>
    <enableDragAndDropScheduling>true</enableDragAndDropScheduling>
    <enableEmailTracking>true</enableEmailTracking>
    <enableGroupTasks>true</enableGroupTasks>
    <enableListViewScheduling>true</enableListViewScheduling>
    <enableMultidayEvents>true</enableMultidayEvents>
    <enableRecurringEvents>true</enableRecurringEvents>
    <enableRecurringTasks>true</enableRecurringTasks>
    <enableSidebarCalendarShortcut>true</enableSidebarCalendarShortcut>
    <meetingRequestsLogo>Folder02/logo03.png</meetingRequestsLogo>
    <showCustomLogoMeetingRequests>true</showCustomLogoMeetingRequests>
    <showEventDetailsMultiUserCalendar>true</showEventDetailsMultiUserCalendar>
    <showHomePageHoverLinksForEvents>true</showHomePageHoverLinksForEvents>
    <showMyTasksHoverLinks>true</showMyTasksHoverLinks>
</ActivitiesSettings>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO: AddressSettings

AddressSettings

Represents the configuration of country and state picklists. Use the AddressSettings component type to configure state and country data in your organization so that you can convert text-based values into standard picklist values. To convert your state and country values, from Setup, enter State and Country Picklists in the Quick Find box, then select State and Country Picklists.

For more information, see “Let Users Select State and Country from Picklists” in the Salesforce online help.

This type extends the Metadata metadata type and inherits its fullName field.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.
Declarative Metadata File Suffix and Directory Location

AddressSettings values are stored in a single file named `Address.settings` in the `settings` directory. The `.settings` files are different from other named components because there is only one settings file for each settings component.

Version

AddressSettings is available in API versions 27.0 and later.

CountriesAndStates

This complex metadata type represents valid definitions of states and countries in picklists.

Note: You can use the Metadata API to edit existing states and countries in state and country picklists. You can’t use the Metadata API to create or delete new states or countries.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>countries</td>
<td><code>Country[]</code></td>
<td>The countries available in picklists.</td>
</tr>
</tbody>
</table>

Country

This metadata type provides the definition for a country in a picklist.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td><code>boolean</code></td>
<td>Determines whether the value is available in the API.</td>
</tr>
</tbody>
</table>

Important: After you enable state and country picklists in your Salesforce organization, you can’t set the active status to `false`.

| integrationValue  | `string`   | A customizable text value that is linked to a state or country code. Integration values for standard states and countries default to the full ISO-standard state and country names. Integration values function similarly to the API names of custom fields and objects. Configuring integration values allows integrations that you set up before enabling state and country picklists to continue to work. |

Important: If you don’t specify integration values before enabling state and country picklists in your organization, records use the default value provided by Salesforce. If you change integration values later, records created or updated from that point on use your edited values.

| isoCode           | `string`   | The ISO-standard code populates this field when you issue a `retrieve()` call. This field is read only in the API but you can edit the label in Setup. You can’t edit the `isoCode` of standard states and countries. |

754
### Field Type

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>The label is what users see in picklists in Salesforce. This field is read only in the API but you can edit the label in Setup.</td>
</tr>
<tr>
<td>orgDefault</td>
<td>boolean</td>
<td>Sets a country as the default value for new records in the Salesforce organization.</td>
</tr>
<tr>
<td>standard</td>
<td>boolean</td>
<td>Standard states and countries are states and countries that are included with Salesforce. You can't edit the standard attribute.</td>
</tr>
<tr>
<td>states</td>
<td>State[]</td>
<td>The states or provinces that are part of the country.</td>
</tr>
<tr>
<td>visible</td>
<td>boolean</td>
<td>Makes the state or country available to users in Salesforce. States or countries that are visible must also be active.</td>
</tr>
</tbody>
</table>

### State

This metadata type provides the definition for a state in a picklist.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Determines whether the value is available in the API. <strong>Important:</strong> After you enable state and country picklists in your Salesforce organization, you can't set the active status to false.</td>
</tr>
<tr>
<td>integrationValue</td>
<td>string</td>
<td>A customizable text value that is linked to a state or country code. Integration values for standard states and countries default to the full ISO-standard state and country names. Integration values function similarly to the API names of custom fields and objects. Configuring integration values allows integrations that you set up before enabling state and country picklists to continue to work. <strong>Important:</strong> If you don't specify integration values before enabling state and country picklists in your organization, records use the default value provided by Salesforce. If you change integration values later, records created or updated from that point on use your edited values.</td>
</tr>
<tr>
<td>isoCode</td>
<td>string</td>
<td>The ISO-standard code populates this field when you issue a retrieve() call. This field is read only in the API but you can edit the label in Setup.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label is what users see in picklists in Salesforce. This field is read only in the API but you can edit the label in Setup.</td>
</tr>
</tbody>
</table>
### Field Type Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>standard</td>
<td>boolean</td>
<td>Standard states and countries are states and countries that are included with Salesforce. You can't edit the standard attribute.</td>
</tr>
<tr>
<td>visible</td>
<td>boolean</td>
<td>Makes the state or country available to users in Salesforce. States or countries that are visible must also be active.</td>
</tr>
</tbody>
</table>

---

**Declarative Metadata Sample Definition**

The following is sample XML that configures state and country picklists for the United States and Canada for use in an organization. It also makes the country of Greenland available only in the API. This example is supported in API version 46.0.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<AddressSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <countriesAndStates>
    <countries>
      <country>
        <active>true</active>
        <integrationValue>United States</integrationValue>
        <isoCode>US</isoCode>
        <label>United States</label>
        <orgDefault>true</orgDefault>
        <standard>true</standard>
        <states>
          <state>
            <active>true</active>
            <integrationValue>Alabama</integrationValue>
            <isoCode>AL</isoCode>
            <label>Alabama</label>
            <standard>true</standard>
            <visible>true</visible>
          </state>
          <state>
            <active>true</active>
            <integrationValue>Alaska</integrationValue>
            <isoCode>AK</isoCode>
            <label>Alaska</label>
            <standard>true</standard>
            <visible>true</visible>
          </state>
        </states>
      </country>
      <country>
        <active>true</active>
        <integrationValue>Canada</integrationValue>
        <isoCode>CA</isoCode>
        <label>Canada</label>
        <orgDefault>false</orgDefault>
        <states>
          
        </states>
      </country>
    </countries>
  </countriesAndStates>
</AddressSettings>
```
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
  Settings

BotSettings

Represents an organization’s Einstein Bot settings, such as whether or not Einstein Bots is enabled. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

Bot components have the suffix .bot and are stored in the bot folder.
Version

Bot components are available in API version 46.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableBots</td>
<td>boolean</td>
<td>Indicates whether Einstein Bots is enabled (true) or not (false).</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a BotSetting. This example has been trimmed to make it easier to read.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<BotSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableBots>true</enableBots>
</BotSettings>
```

The following is an example package.xml manifest that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Bot</members>
    <name>Settings</name>
  </types>
  <version>46.0</version>
</Package>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

BusinessHoursSettings

Represents the metadata used to manage settings for business hours and holidays in entitlements, entitlement templates, campaigns, and cases. This type extends the Metadata metadata type and inherits its fullName field.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

File Suffix and Directory Location

Business hours and holidays settings are stored in a single file named businessHours.settings in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.
Version

BusinessHoursSettings is available in API version 29.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>businessHours</td>
<td>BusinessHoursEntry[]</td>
<td>Represents the application of business hours to entitlements, entitlement templates, campaigns, and cases.</td>
</tr>
<tr>
<td>holidays</td>
<td>Holidays[]</td>
<td>Represents a holiday and its usage in businessHours.</td>
</tr>
</tbody>
</table>

BusinessHoursEntry

Represents the application of business hours to entitlements, entitlement templates, campaigns, and cases.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeZoneId</td>
<td>string</td>
<td>The time zone for the time that defines business hours.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the business hours. This name should be unique.</td>
</tr>
<tr>
<td>active</td>
<td>string</td>
<td>Indicates whether the business hours are active.</td>
</tr>
<tr>
<td>default</td>
<td>string</td>
<td>Indicates whether the business hours are used as the default business hours.</td>
</tr>
<tr>
<td>mondayStartTime</td>
<td>string</td>
<td>Start time for the business hours on Monday. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
<tr>
<td>mondayEndTime</td>
<td>string</td>
<td>End time for the business hours on Monday. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
<tr>
<td>tuesdayStartTime</td>
<td>string</td>
<td>Start time for the business hours on Tuesday. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
<tr>
<td>tuesdayEndTime</td>
<td>string</td>
<td>End time for the business hours on Tuesday. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
<tr>
<td>wednesdayStartTime</td>
<td>string</td>
<td>Start time for the business hours on Wednesday. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
<tr>
<td>wednesdayEndTime</td>
<td>string</td>
<td>End time for the business hours on Wednesday. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
<tr>
<td>thursdayStartTime</td>
<td>string</td>
<td>Start time for the business hours on Thursday. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
</tbody>
</table>
### BusinessHoursSettings

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>thursdayEndTime</td>
<td>string</td>
<td>End time for the business hours on Thursday. Uses the format HH:mm:ss.SSSZ. The value 00:00:00.000Z specifies midnight on Thursday.</td>
</tr>
<tr>
<td>fridayStartTime</td>
<td>string</td>
<td>Start time for the business hours on Friday. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
<tr>
<td>fridayEndTime</td>
<td>string</td>
<td>End time for the business hours on Friday. Uses the format HH:mm:ss.SSSZ. The value 00:00:00.000Z specifies midnight on Friday.</td>
</tr>
<tr>
<td>saturdayStartTime</td>
<td>string</td>
<td>Start time for the business hours on Saturday. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
<tr>
<td>saturdayEndTime</td>
<td>string</td>
<td>End time for the business hours on Saturday. Uses the format HH:mm:ss.SSSZ. The value 00:00:00.000Z specifies midnight on Saturday.</td>
</tr>
<tr>
<td>sundayStartTime</td>
<td>string</td>
<td>Start time for the business hours on Sunday. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
<tr>
<td>sundayEndTime</td>
<td>string</td>
<td>End time for the business hours on Sunday. Uses the format HH:mm:ss.SSSZ. The value 00:00:00.000Z specifies midnight on Sunday.</td>
</tr>
</tbody>
</table>

### Holidays

Represents a holiday and its usage in `businessHours`.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the holiday. This name does not have to be unique.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the holiday.</td>
</tr>
<tr>
<td>isRecurring</td>
<td>string</td>
<td>Indicates whether the holiday is recurring.</td>
</tr>
<tr>
<td>activityDate</td>
<td>string</td>
<td>The date of the holiday. Use for non-recurring holidays. Uses the format HH:mm:ss.SSSZ.</td>
</tr>
<tr>
<td>recurrenceStartDate</td>
<td>string</td>
<td>The date the holiday starts recurring. Uses the format yyyy-mm-dd.</td>
</tr>
<tr>
<td>recurrenceEndDate</td>
<td>string</td>
<td>The date the holiday stops recurring. Uses the format yyyy-mm-dd. Optional.</td>
</tr>
<tr>
<td>startTime</td>
<td>string</td>
<td>The start time on the date of the holiday. Uses the format HH:mm:ss.SSSZ. The value must be both null or both not null. If they are both null, indicates the whole day.</td>
</tr>
<tr>
<td>endTime</td>
<td>string</td>
<td>The end time on the date of the holiday. Uses the format HH:mm:ss.SSSZ. The value must be both null or both not null. If they are both null, indicates the whole day.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>recurrenceType</td>
<td>string</td>
<td>The recurrence type of the holiday. Valid values: RecursDaily,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RecursEvery weekday, RecursMonthly, RecursMonthlyNth, RecursWeekly,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RecursYearly, RecursYearlyNth.</td>
</tr>
<tr>
<td>recurrenceInterval</td>
<td>string</td>
<td>The interval of weeks, months, or years the holiday recurs.</td>
</tr>
<tr>
<td>recurrenceDayOfWeek</td>
<td>string</td>
<td>The day of week the holiday recurs. Valid values: Monday, Tuesday,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Wednesday, Thursday, Friday, Saturday, Sunday.</td>
</tr>
<tr>
<td>recurrenceDayOfMonth</td>
<td>string</td>
<td>The day of month the holiday recurs. Valid values: integers 1-31.</td>
</tr>
<tr>
<td>recurrenceInstance</td>
<td>string</td>
<td>Valid values: First, Second, Third, Fourth, Last. Only used for</td>
</tr>
<tr>
<td></td>
<td></td>
<td>recurrenceType RecursMonthlyNth and RecursYearlyNth. For example,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>if the recurrenceInstance value is First, the holiday recurs on the first</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monday of the month every 3 months.</td>
</tr>
<tr>
<td>recurrenceMonthOfYear</td>
<td>string</td>
<td>Valid values: January, February, March, April, May, June, July, August,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>September, October, November, December.</td>
</tr>
<tr>
<td>businessHours</td>
<td>string</td>
<td>The name of the business hours setting that applies to this holiday.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example `businesshours.settings` metadata file:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<BusinessHoursSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <businessHours>
    <active>true</active>
    <default>true</default>
    <fridayEndTime>00:00:00.000Z</fridayEndTime>
    <fridayStartTime>00:00:00.000Z</fridayStartTime>
    <mondayEndTime>00:00:00.000Z</mondayEndTime>
    <mondayStartTime>00:00:00.000Z</mondayStartTime>
    <saturdayEndTime>00:00:00.000Z</saturdayEndTime>
    <saturdayStartTime>00:00:00.000Z</saturdayStartTime>
    <sundayEndTime>00:00:00.000Z</sundayEndTime>
    <sundayStartTime>00:00:00.000Z</sundayStartTime>
    <thursdayEndTime>00:00:00.000Z</thursdayEndTime>
    <thursdayStartTime>00:00:00.000Z</thursdayStartTime>
    <timeZoneId>America/Los_Angeles</timeZoneId>
    <tuesdayEndTime>00:00:00.000Z</tuesdayEndTime>
    <tuesdayStartTime>00:00:00.000Z</tuesdayStartTime>
    <wednesdayEndTime>00:00:00.000Z</wednesdayEndTime>
    <wednesdayStartTime>00:00:00.000Z</wednesdayStartTime>
  </businessHours>
  <businessHours>
    <active>true</active>
    <default>false</default>
    <fridayEndTime>00:00:00.000Z</fridayEndTime>
    <fridayStartTime>00:00:00.000Z</fridayStartTime>
  </businessHours>
</BusinessHoursSettings>
```
The following is an example `package.xml` manifest that references the BusinessHoursSettings definitions:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>BusinessHours</members>
    <name>Settings</name>
  </types>
  <version>29.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

The wildcard character * (asterisk) in the `package.xml` manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
CaseSettings

Represents an organization’s case settings, such as the default case owner, which case-related features are enabled, and which Classic email templates are used for various case activities.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

File Suffix and Directory Location

CaseSettings values are stored in the Case.settings file in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

CaseSettings is available in API version 27.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>caseAssignNotificationTemplate</td>
<td>string</td>
<td>Specifies the email template used for case assignment notifications. The format must be folderName/emailTemplateName. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>caseCloseNotificationTemplate</td>
<td>string</td>
<td>Specifies the email template used for case close notifications. The format must be folderName/emailTemplateName. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>caseCommentNotificationTemplate</td>
<td>string</td>
<td>Specifies the email template used for case comment notifications. The format must be folderName/emailTemplateName. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>caseCreateNotificationTemplate</td>
<td>string</td>
<td>Specifies the email template used for case create notifications. The format must be folderName/emailTemplateName. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>caseFeedItemSettings</td>
<td>FeedItemSettings[]</td>
<td>Specifies the settings for feed items in feed-based case page layouts. This field is available in API version 32.0 and later.</td>
</tr>
<tr>
<td>closeCaseThroughStatusChange</td>
<td>boolean</td>
<td>Indicates whether Closed is included in the Case Status field on case edit pages (true) or not (false).</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>defaultCaseOwner</td>
<td>string</td>
<td>Specifies the default owner of a case when assignment rules fail to locate an owner.</td>
</tr>
<tr>
<td>defaultCaseOwnerType</td>
<td>string</td>
<td>Specifies whether the default case owner is a user or a queue.</td>
</tr>
</tbody>
</table>
| defaultCaseUser            | string     | Specifies the user listed in the Case History related list for automated case changes from:  
  - Assignment rules  
  - Escalation rules  
  - On-Demand Email-to-Case  
  - Cases logged in the Self-Service portal  
  Lightning email templates aren’t packageable. We recommend using a Classic email template. |
| emailToCase                | EmailToCaseSettings | The organization’s Email-to-Case settings.                                   |
| enableCaseFeed             | boolean    | Indicates whether Case Feed is enabled (true) or not (false).                 |
| enableDraftEmails          | boolean    | Indicates whether draft emails are enabled (true) or not (false). Enabling email drafts requires that Case Feed and Email-to-Case are also enabled. |
| enableEarlyEscalationRuleTriggers | boolean | Indicates whether early triggers on escalation rules are enabled (true) or not (false). |
| enableE2CSourceTracking    | boolean    | Indicates whether Set Case Source to Email is enabled (true) or not (false). After you enable this setting, the Case Source field is updated to Email for all cases that originate from Email-to-Case. Associated emails are marked as Read when the agent opens the case. |
| enableNewEmailDefaultTemplate | boolean | Indicates whether default email templates are enabled (true) or not (false). Default email templates are available only if draft emails are enabled.  
  Lightning email templates aren’t packageable. We recommend using a Classic email template. |
<p>| enableSuggestedArticlesApplication | boolean | Indicates whether the Suggested Articles list appears on case pages. (true) or not (false). Is only valid if enableSuggestedSolutions=false. |
| enableSuggestedArticlesCustomerPortal | boolean | Indicates whether the Suggested Articles list appears on customer portal pages (true) or not (false). Is only valid if enableSuggestedSolutions=false. |
| enableSuggestedArticlesPartnerPortal | boolean | Indicates whether the Suggested Articles list appears on partner portal pages (true) or not (false). Is only valid if enableSuggestedSolutions=false. |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableSuggestedSolutions</td>
<td>boolean</td>
<td>Indicates whether the View Suggested Solutions or Find Articles button appears on case detail pages (true) or not (false). Is only valid if enableSuggestedArticlesApplication, enableSuggestedArticlesCustomerPortal, and enableSuggestedArticlesPartnerPortal=false.</td>
</tr>
<tr>
<td>keepRecordTypeOnAssignmentRule</td>
<td>boolean</td>
<td>Indicates whether, when applying assignment rules to manually created records, to keep the existing record type (true) or to override the existing record type with the assignee’s default record type (false).</td>
</tr>
<tr>
<td>newEmailDefaultTemplateClass</td>
<td>string</td>
<td>Specifies the Apex class that defines the default email template for new email messages in Case Feed. This field appears only when enableNewEmailDefaultTemplate=true. Lightning email templates aren’t packageable. We recommend using a Classic email template.</td>
</tr>
<tr>
<td>notifyContactOnCaseComment</td>
<td>boolean</td>
<td>Indicates whether contacts who are not members of your Self-Service portal can be notified when a new comment is added to a case (true) or not (false).</td>
</tr>
<tr>
<td>notifyDefaultCaseOwner</td>
<td>boolean</td>
<td>Indicates whether the default case owner is notified when assigned a new case (true) or not (false).</td>
</tr>
<tr>
<td>notifyOwnerOnCaseComment</td>
<td>boolean</td>
<td>Indicates whether the case owner is notified when a comment is added to a case (true) or not (false).</td>
</tr>
<tr>
<td>notifyOwnerOnCaseOwnerChange</td>
<td>boolean</td>
<td>Indicates whether the Send Notification Email checkbox on cases is automatically selected when users change a case owner to another user (true).</td>
</tr>
<tr>
<td>showEmailAttachmentsInCase</td>
<td>boolean</td>
<td>Indicates whether the case Attachments related list shows email attachments. If true, the page displays an email icon next to each attachment from an email in the Attachments related list for cases. The related list’s list view also includes a Source column that identifies the attachment’s origin. If false, email attachments aren’t displayed in the Attachments related list for cases. This field is available in API version 40.0 and later.</td>
</tr>
<tr>
<td>showFewerCloseActions</td>
<td>boolean</td>
<td>Indicates whether the Save &amp; Close button on case edit pages and the Cls link on Cases related lists are hidden (true) or shown (false).</td>
</tr>
<tr>
<td>systemUserEmail</td>
<td>string</td>
<td>Specifies the email address used when the default case user is the system user.</td>
</tr>
</tbody>
</table>

765
### CaseSettings

Represents an organization's Email-to-Case settings.

#### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>useSystemEmailAddress</td>
<td>boolean</td>
<td>Indicates whether case comment, case attachment, and case assignment email notifications are sent from a system address (true) or whether case notifications appear to be sent from the user or contact updating the case (false).</td>
</tr>
<tr>
<td>useSystemUserAsDefaultCaseUser</td>
<td>boolean</td>
<td>Indicates whether the system user is used as the automated case user (true) or not (false). If false, then you must specify a value for the defaultCaseUser field.</td>
</tr>
<tr>
<td>webToCase</td>
<td>WebToCaseSettings</td>
<td>The organization's Web-to-Case settings.</td>
</tr>
<tr>
<td>enableEmailToCase</td>
<td>boolean</td>
<td>Indicates whether Email-to-Case is enabled (true) or not (false). Note: once Email-to-Case is enabled, it can't be disabled.</td>
</tr>
<tr>
<td>enableHtmlEmail</td>
<td>boolean</td>
<td>Indicates whether HTML email is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableOnDemandEmailToCase</td>
<td>boolean</td>
<td>Indicates whether On-Demand Email-to-Case is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableThreadIDInBody</td>
<td>boolean</td>
<td>Indicates whether the Thread ID for a case is inserted in the body of an email (true) or not (false).</td>
</tr>
<tr>
<td>enableThreadIDInSubject</td>
<td>boolean</td>
<td>Indicates whether the Thread ID for a case is inserted in the subject line of an email (true) or not (false).</td>
</tr>
<tr>
<td>notifyOwnerOnNewCaseEmail</td>
<td>boolean</td>
<td>Indicates whether the owner of a case receives a notification when a new email related to the case is received (true) or not (false).</td>
</tr>
<tr>
<td>overEmailLimitAction</td>
<td>EmailToCaseOnFailureActionType</td>
<td>Specifies what happens to email messages received after an organization exceeds its daily Email-to-Case limits. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>- Bounce</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Discard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Requeue</td>
</tr>
<tr>
<td>preQuoteSignature</td>
<td>boolean</td>
<td>Indicates whether the user signature is inserted after the reply but before the email thread in an outbound email (true) or at the end of the email (false).</td>
</tr>
</tbody>
</table>
**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>routingAddresses</td>
<td>EmailToCaseRoutingAddress[]</td>
<td>The organization’s Email-to-Case routing address settings.</td>
</tr>
<tr>
<td>unauthorizedSenderAction</td>
<td>EmailToCaseOnFailureActionType (enumeration of type string)</td>
<td>Specifies what happens to email messages received from invalid senders. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Bounce</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Discard</td>
</tr>
</tbody>
</table>

**EmailToCaseRoutingAddress**

Represents an organization’s Email-to-Case routing address.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addressType</td>
<td>EmailToCaseRoutingAddressType (enumeration of type string)</td>
<td>Specifies the type of Email-to-Case routing address. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• EmailToCase—A routing address used with Email-to-Case or On-Demand Email-to-Case.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Outlook—A routing address used with Salesforce for Outlook to create cases from Outlook. Requires that On-Demand Email-to-Case is enabled.</td>
</tr>
<tr>
<td>authorizedSenders</td>
<td>string</td>
<td>Specifies the email addresses or domains from which On-Demand Email-to-Case can receive email. Include multiple entries in a comma-separated list.</td>
</tr>
<tr>
<td>caseOrigin</td>
<td>string</td>
<td>Specifies the default case origin for cases created through this routing address.</td>
</tr>
<tr>
<td>caseOwner</td>
<td>string</td>
<td>Specifies the default owner of cases created through this routing address. The case owner can be either a user or a queue. Specify the case owner using a Salesforce username. Specifying a case owner here in the routing address settings value of defaultCaseOwner in CaseSettings.</td>
</tr>
<tr>
<td>caseOwnerType</td>
<td>string</td>
<td>Specifies whether the default case owner is a user or a queue.</td>
</tr>
<tr>
<td>casePriority</td>
<td>string</td>
<td>Specifies the default case priority for cases created through this routing address.</td>
</tr>
<tr>
<td>createTask</td>
<td>boolean</td>
<td>Indicates whether a task is automatically assigned to the case owner when a case is created through an email (true) or not (false).</td>
</tr>
<tr>
<td>emailAddress</td>
<td>string</td>
<td>Specifies the email address used to route email messages that are submitted as cases.</td>
</tr>
</tbody>
</table>
### Field Name

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emailServicesAddress</td>
<td>string</td>
<td>Specifies the Salesforce-generated routing address used for setting up Email-to-Case forwarding. This field value is read-only and can't be modified.</td>
</tr>
<tr>
<td>isVerified</td>
<td>boolean</td>
<td>Indicates whether the customer has verified the routing address (typically by clicking a confirmation email). This field value is read-only and can't be modified.</td>
</tr>
<tr>
<td>routingName</td>
<td>string</td>
<td>Specifies the name of the Email-to-Case routing address.</td>
</tr>
<tr>
<td>saveEmailHeaders</td>
<td>boolean</td>
<td>Indicates whether email routing and envelope information are saved (true) or not (false).</td>
</tr>
<tr>
<td>taskStatus</td>
<td>string</td>
<td>Specifies the default status on tasks automatically assigned to the case owner when email is submitted as a case. Only applies if createTask is set to true.</td>
</tr>
</tbody>
</table>

### FeedItemSettings

Represents an organization’s feed item settings. Available in API version 32.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>characterLimit</td>
<td>int</td>
<td>Specifies the maximum number of characters displayed for each feed item.</td>
</tr>
<tr>
<td>collapseThread</td>
<td>boolean</td>
<td>Indicates whether earlier messages in an email thread are removed from email feed items (true) or not (false).</td>
</tr>
</tbody>
</table>
| displayFormat    | FeedItemDisplayFormat (enumeration of type string) | Indicates how email feed items are displayed. Valid values are:  
  - Default—Blank lines in email feed items are displayed.  
  - HideBlankLines—Blank lines in email feed items are not displayed. |
| feedItemType     | FeedItemType (enumeration of type string) | The type of feed item to which the settings apply. For FeedItemSettings, the only valid feedItemType value is EmailMessageEvent. |

### WebToCaseSettings

Represents an organization’s Web-to-Case settings.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>caseOrigin</td>
<td>string</td>
<td>Specifies the default case origin for cases created through this web form. Only applies if enableWebToCase is set to true.</td>
</tr>
</tbody>
</table>
### Field Name: defaultResponseTemplate
**Field Type:** string
**Description:** Specifies the default template used for email responses to cases submitted through a Self-Service portal. Only applies if enableWebToCase is set to true. Lightning email templates aren’t packageable. We recommend using a Classic email template.

### Field Name: enableWebToCase
**Field Type:** boolean
**Description:** Indicates whether Web-to-Case is enabled (true) or not (false).

---

**Declarative Metadata Sample Definition**

This code sample is an example of a case settings file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CASESETTINGS xmlns="http://soap.sforce.com/2006/04/metadata">
  <caseAssignNotificationTemplate>
    unfiled$public/SupportCaseAssignmentNotification
  </caseAssignNotificationTemplate>
  <caseCloseNotificationTemplate>
    unfiled$public/SupportCaseCloseNotification
  </caseCloseNotificationTemplate>
  <caseCommentNotificationTemplate>
    unfiled$public/SupportCaseCommentNotification
  </caseCommentNotificationTemplate>
  <caseCreateNotificationTemplate>
    unfiled$public/SupportCaseCreateNotification
  </caseCreateNotificationTemplate>
  <closeCaseThroughStatusChange>true</closeCaseThroughStatusChange>
  <defaultCaseOwner>admin@acme.com</defaultCaseOwner>
  <defaultCaseOwnerType>User</defaultCaseOwnerType>
  <defaultCaseUser>admin@acme.com</defaultCaseUser>
  <emailToCase>
    <enableEmailToCase>true</enableEmailToCase>
    <enableHtmlEmail>false</enableHtmlEmail>
    <enableOnDemandEmailToCase>true</enableOnDemandEmailToCase>
    <enableThreadIDInBody>true</enableThreadIDInBody>
    <enableThreadIDInSubject>true</enableThreadIDInSubject>
    <notifyOwnerOnNewCaseEmail>false</notifyOwnerOnNewCaseEmail>
    <overEmailLimitAction>Bounce</overEmailLimitAction>
    <preQuoteSignature>true</preQuoteSignature>
  </emailToCase>
  <routingAddresses>
    <addressType>EmailToCase</addressType>
    <authorizedSenders>user@acme.com</authorizedSenders>
    <caseOrigin>Email</caseOrigin>
    <casePriority>Medium</casePriority>
    <createTask>true</createTask>
    <emailAddress>support@acme.com</emailAddress>
    <routingName>EmailToCaseRoutingAddress1</routingName>
    <saveEmailHeaders>true</saveEmailHeaders>
    <taskStatus>Not Started</taskStatus>
  </routingAddresses>
</CASESETTINGS>
```
<routingAddresses>
  <addressType>Outlook</addressType>
  <authorizedSenders>user@acme.com</authorizedSenders>
  <caseOrigin>Email</caseOrigin>
  <caseOwner>admin@acme.com</caseOwner>
  <caseOwnerType>User</caseOwnerType>
  <casePriority>High</casePriority>
  <routingName>OutlookRoutingAddress1</routingName>
</routingAddresses>

<unauthorizedSenderAction>Discard</unauthorizedSenderAction>
</emailToCase>
<enableCaseFeed>true</enableCaseFeed>
<enableDraftEmails>true</enableDraftEmails>
<enableEarlyEscalationRuleTriggers>true</enableEarlyEscalationRuleTriggers>
<enableNewEmailDefaultTemplate>true</enableNewEmailDefaultTemplate>
<enableSuggestedArticlesApplication>true</enableSuggestedArticlesApplication>
<enableSuggestedArticlesCustomerPortal>true</enableSuggestedArticlesCustomerPortal>
<enableSuggestedArticlesPartnerPortal>false</enableSuggestedArticlesPartnerPortal>
<enableSuggestedSolutions>false</enableSuggestedSolutions>
<keepRecordTypeOnAssignmentRule>true</keepRecordTypeOnAssignmentRule>
<newEmailDefaultTemplateClass>CaseTemplateController</newEmailDefaultTemplateClass>
<notifyContactOnCaseComment>true</notifyContactOnCaseComment>
<notifyDefaultCaseOwner>true</notifyDefaultCaseOwner>
<notifyOwnerOnCaseComment>true</notifyOwnerOnCaseComment>
<notifyOwnerOnCaseOwnerChange>false</notifyOwnerOnCaseOwnerChange>
<showFewerCloseActions>false</showFewerCloseActions>
<useSystemEmailAddress>true</useSystemEmailAddress>
</webToCase>
  <caseOrigin>Web</caseOrigin>
  <defaultResponseTemplate>unfiled$public/SupportCaseResponse</defaultResponseTemplate>
</webToCase>
</CaseSettings>

### Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- Settings

### ChatterAnswersSettings

Represents the metadata used to manage settings for Chatter Answers.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.
File Suffix and Directory Location

Chatter Answers settings are stored in a single file named ChatterAnswers.settings in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

ChatterAnswersSettings is available in API version 27.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emailFollowersOnBestAnswer</td>
<td>boolean</td>
<td>Indicates whether users are notified when a best answer is selected for a question that they're following (true) or not (false).</td>
</tr>
<tr>
<td>emailFollowersOnReply</td>
<td>boolean</td>
<td>Indicates whether users are notified when other users reply to questions they're following (true) or not (false).</td>
</tr>
<tr>
<td>emailOwnerOnPrivateReply</td>
<td>boolean</td>
<td>Indicates whether users are notified when customer support responds to their questions privately (true) or not (false).</td>
</tr>
<tr>
<td>emailOwnerOnReply</td>
<td>boolean</td>
<td>Indicates whether users are notified when other users reply to their questions (true) or not (false).</td>
</tr>
<tr>
<td>enableAnswerViaEmail</td>
<td>boolean</td>
<td>Indicates whether users can post answers by replying to email notifications (true) or not (false). This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>enableChatterAnswers</td>
<td>boolean</td>
<td>Indicates whether Chatter Answers is enabled in the organization (true) or not (false).</td>
</tr>
<tr>
<td>enableFacebookSSO</td>
<td>boolean</td>
<td>Indicates whether users sign in to your Chatter Answers communities with their Facebook logins (true) or not (false). To enable this feature, you must define and enable a Facebook authentication provider in your organization's security controls and enable Auth Providers in your organization.</td>
</tr>
<tr>
<td>enableInlinePublisher</td>
<td>boolean</td>
<td>Indicates whether users can filter search results by articles or questions before they post a question to any of your Chatter Answers communities (true) or not (false). Also, adds Title and Body fields to questions for easier text input and scanning. This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>enableReputation</td>
<td>boolean</td>
<td>Indicates whether reputations display for users as hover text on their profile pictures (true) or not (false). Reputation is enabled across all zones. To enable the reputation setting, you must enable Reputation in your organization.</td>
</tr>
<tr>
<td>enableRichTextEditor</td>
<td>boolean</td>
<td>Indicates whether the rich text editor is enabled for users to format text and upload images when posting questions (true) or not (false). To enable rich text editor, you must enable Optimize Question Flow.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>facebookAuthProvider</td>
<td>string</td>
<td>The name of an existing Facebook authentication provider. You must choose a Facebook authentication provider to implement Facebook Single Sign On for your Chatter Answers communities.</td>
</tr>
<tr>
<td>showInPortals</td>
<td>boolean</td>
<td>Indicates whether Chatter Answers can be added as a tab to your Customer portal or partner portal (true) or not (false).</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example `chatteranswers.settings` metadata file:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ChatterAnswersSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <emailFollowersOnBestAnswer>true</emailFollowersOnBestAnswer>
  <emailFollowersOnReply>true</emailFollowersOnReply>
  <emailOwnerOnPrivateReply>true</emailOwnerOnPrivateReply>
  <emailOwnerOnReply>true</emailOwnerOnReply>
  <enableChatterAnswers>true</enableChatterAnswers>
  <enableFacebookSSO>true</enableFacebookSSO>
  <enableInlinePublisher>true</enableInlinePublisher>
  <enableReputation>true</enableReputation>
  <enableRichTextEditor>true</enableRichTextEditor>
  <facebookAuthProvider>FacebookAuthProvider</facebookAuthProvider>
  <showInPortals>true</showInPortals>
</ChatterAnswersSettings>
```

The following is an example `package.xml` manifest that references the ChatterAnswersSettings definitions:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>ChatterAnswers</members>
    <name>Settings</name>
  </types>
  <version>29.0</version>
</Package>
```

### Wildcard Support in the Manifest File

The wildcard character `*` (asterisk) in the `package.xml` manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see `Settings`. For information about using the manifest file, see `Deploying and Retrieving Metadata with the Zip File`.

SEE ALSO:
- `Settings`

### CompanySettings

Represents global settings that affect multiple features in your organization.
In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

**Declarative Metadata File Suffix and Directory Location**

CompanySettings values are stored in a single file named Company.settings in the settings directory of the corresponding package directory. The .settings files are different from other named components because there is only one settings file for each settings component.

**Version**

Company Profile Settings are available in API version 27.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fiscalYear</td>
<td>FiscalYearSetting</td>
<td>The organization’s fiscal year setting based on year and start month. Not available if Custom Fiscal Year or Forecasts (Classic) is enabled. When changing fiscal year settings, quotas and adjustments can be purged. For example changing your start month results in purging this data.</td>
</tr>
</tbody>
</table>

**FiscalYearSetting**

Represents your organization’s fiscal year setting.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| fiscalYearNameBasedOn | string     | This field is used to determine the fiscal year name. Valid values are endingMonth or startingMonth. For example, if your fiscal year starts in April 2012 and ends in March 2013, and this value is:  
  • endingMonth, then 2013 is used for the fiscal year name.  
  • startingMonth, then 2012 is used for the fiscal year name.  |
| startMonth          | string     | The month on which the fiscal year is based. |

**Declarative Metadata Sample Definition — Fiscal Year Setting**

A sample XML definition of a fiscal year setting is shown below. Note that this example is supported in API version 27.0 and later.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CompanySettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <fiscalYear>
    <fiscalYearNameBasedOn>endingMonth</fiscalYearNameBasedOn>
    <startMonth>January</startMonth>
  </fiscalYear>
</CompanySettings>
```
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the `package.xml` manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- Settings

ContractSettings

Represents contract settings. For more information, see “Set Up Contracts” in the Salesforce online help.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

File Suffix and Directory Location

There is one contract settings file stored in a file named `Contract.settings` in the `settings` directory. The `.settings` files are different from other named components because there is only one settings file for each settings component.

Version

ContractSettings is available in API version 27.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoCalculateEndDate</td>
<td>boolean</td>
<td>Indicates whether the end date of a contract is automatically calculated (true) or not (false).</td>
</tr>
<tr>
<td>notifyOwnersOnContractExpiration</td>
<td>boolean</td>
<td>Indicates whether account and contract owners are automatically sent email notifications when a contract expires (true) or not (false).</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

This is a sample contract settings file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ContractSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <autoCalculateEndDate>true</autoCalculateEndDate>
  <notifyOwnersOnContractExpiration>false</notifyOwnersOnContractExpiration>
</ContractSettings>
```
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:

Settings

EntitlementSettings

Represents an organization’s entitlement settings.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

File Suffix and Directory Location

EntitlementSettings values are stored in the Entitlements.settings file in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

EntitlementSettings is available in API version 27.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assetLookupLimitedToActiveEntitlementsOrAccount</td>
<td>boolean</td>
<td>Indicates whether entitlements-related lookup filters on cases return only the assets related to the active entitlements on the case’s account (true) or not (false).</td>
</tr>
<tr>
<td>assetLookupLimitedToActiveEntitlementsOrContact</td>
<td>boolean</td>
<td>Indicates whether entitlements-related lookup filters on cases return only the assets related to the active entitlements on the case’s contact (true) or not (false).</td>
</tr>
<tr>
<td>assetLookupLimitedToSameAccount</td>
<td>boolean</td>
<td>Indicates whether entitlements-related lookup filters on cases return only the assets related to the case’s account (true) or not (false).</td>
</tr>
<tr>
<td>assetLookupLimitedToSameContact</td>
<td>boolean</td>
<td>Indicates whether entitlements-related lookup filters on cases return only the assets related to the case’s contact (true) or not (false).</td>
</tr>
<tr>
<td>enableEntitlements</td>
<td>boolean</td>
<td>Indicates whether entitlements are enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableEntitlementVersioning</td>
<td>boolean</td>
<td>Indicates whether entitlement versioning is enabled (true) or not (false). This field is available in API version 28.0 and later.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>entitlementLookupLimitedToActiveStatus</td>
<td>boolean</td>
<td>Indicates whether entitlements-related lookup filters on cases return only active entitlements (true) or not (false).</td>
</tr>
<tr>
<td>entitlementLookupLimitedToSameAccount</td>
<td>boolean</td>
<td>Indicates whether entitlements-related lookup filters on cases return only the entitlements related to the case’s account (true) or not (false).</td>
</tr>
<tr>
<td>entitlementLookupLimitedToSameAsset</td>
<td>boolean</td>
<td>Indicates whether entitlements-related lookup filters on cases return only the entitlements related to the case’s asset (true) or not (false).</td>
</tr>
<tr>
<td>entitlementLookupLimitedToSameContact</td>
<td>boolean</td>
<td>Indicates whether entitlements-related lookup filters on cases return only the entitlements related to the case’s contact (true) or not (false).</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

This is a sample entitlements settings file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<EntitlementSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <assetLookupLimitedToActiveEntitlementsOnAccount>
    false
  </assetLookupLimitedToActiveEntitlementsOnAccount>
  <assetLookupLimitedToActiveEntitlementsOnContact>
    false
  </assetLookupLimitedToActiveEntitlementsOnContact>
  <assetLookupLimitedToSameAccount>
    false
  </assetLookupLimitedToSameAccount>
  <assetLookupLimitedToSameContact>
    false
  </assetLookupLimitedToSameContact>
  <enableEntitlements>
    true
  </enableEntitlements>
  <entitlementLookupLimitedToActiveStatus>
    false
  </entitlementLookupLimitedToActiveStatus>
  <entitlementLookupLimitedToSameAccount>
    false
  </entitlementLookupLimitedToSameAccount>
  <entitlementLookupLimitedToSameAsset>
    false
  </entitlementLookupLimitedToSameAsset>
  <entitlementLookupLimitedToSameContact>
    false
  </entitlementLookupLimitedToSameContact>
</EntitlementSettings>
```
**Wildcard Support in the Manifest File**

The wildcard character `*` (asterisk) in the `package.xml` manifest file doesn't apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**SEE ALSO:**
- Settings

**ExperienceBundleSettings (Developer Preview)**

Represents the settings of ExperienceBundleMetadata of an organization.

**File Suffix and Directory Location**

ExperienceBundleSettings values are stored in a single file named `ExperienceBundleMetadata.settings` in the `settings` directory. The `.settings` files are different from other named components because there is only one settings file for each settings component.

**Version**

ExperienceBundleSettings is available in API version 46.0 and later.

**Note:** This release contains a Developer Preview version of ExperienceBundle Metadata API that is production quality but has known limitations. To provide feedback and suggestions, go to IdeaExchange.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableExperienceBundleMetadata</td>
<td>boolean</td>
<td>Indicates whether ExperienceBundleMetadata is enabled.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

Here's an example `ExperienceBundleSettings.xml` that references the previous definition.

```xml
<ExperienceBundleSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableExperienceBundleMetadata>true</enableExperienceBundleMetadata>
</ExperienceBundleSettings>
```
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:

ExperienceBundle (Developer Preview)

FieldServiceSettings

Represents an organization’s Field Service Lightning settings.

To learn more about Field Service Lightning settings, see Enable Field Service Lightning in the Salesforce Help.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

Version

FieldServiceSettings is available in API version 40.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doesAllowEditSaForCrew</td>
<td>boolean</td>
<td>Lets service crew members edit their service appointments. This setting applies only if doesShareSaWithAr is selected. For assigned resources of type Crew, crew members get Read-Write access to their service appointment and, if doesShareSaParentWoWithAr is selected, to their service appointments’ parent work orders.</td>
</tr>
<tr>
<td>doesShareSaParentWoWithAr</td>
<td>boolean</td>
<td>Shares service appointments’ parent work orders with their assigned resources. This setting applies only if doesShareSaWithAr is selected and sharing access for work orders is set to Private or Public Read Only. Technician assigned resources get Read-Write access to their work orders. For assigned resources of type Crew, the crew leader gets Read-Write access and crew members get Read access. If the service appointment’s parent is a work order line item, assigned resources get access to the associated work order.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>doesShareSaWithAr</td>
<td>boolean</td>
<td>Shares dispatched service appointments with their assigned resources. This setting applies only if sharing access for service appointments is set to Private or Public Read Only. Technician assigned resources get Read-Write access to their service appointments. For assigned resources of type Crew, the crew leader gets Read-Write access and crew members get Read access.</td>
</tr>
</tbody>
</table>
| fieldServiceNotificationsOrgPref           | boolean    | Turns on in-app notifications for the Salesforce app and Lightning Experience users when any of the following actions occurs on a work order or work order line item that they own or follow:  
• A text or file post is added  
• A tracked field is updated  
• The record owner changes  
• The resource assignments change on a related service appointment  
If the option to track all related objects is selected in the feed tracking settings for work orders, users are also notified when child records of work orders—such as service appointments—are created or deleted. |
| fieldServiceOrgPref                        | boolean    | Indicates whether Field Service Lightning is enabled.                                                                                                                                               |
| serviceAppointmentsDueDateOffsetOrgValue  | int        | Indicates the number of days past the Created Date that the Due Date on auto-created service appointments should fall. Work types include an option to automatically add a service appointment to new work orders or work order line items using the work type. |
| workOrderLineItemSearchFields             | string     | The work order line item fields that the search engine should scan to suggest knowledge articles on work order line items.                                                                        |
| workOrderSearchFields                     | string     | The work order fields that the search engine should scan to suggest knowledge articles on work orders.                                                                                              |
Declarative Metadata Sample Definition

This is a sample field service settings file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<FieldServiceSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <fieldServiceNotificationsOrgPref>false</fieldServiceNotificationsOrgPref>
  <fieldServiceOrgPref>true</fieldServiceOrgPref>
  <serviceAppointmentsDueDateOffsetOrgValue>6</serviceAppointmentsDueDateOffsetOrgValue>
  <workOrderSearchFields>WorkOrderNumber</workOrderSearchFields>
  <workOrderSearchFields>Status</workOrderSearchFields>
  <workOrderSearchFields>Address</workOrderSearchFields>
  <workOrderSearchFields>Subject</workOrderSearchFields>
  <workOrderLineItemSearchFields>Subject</workOrderLineItemSearchFields>
</FieldServiceSettings>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

FileUploadAndDownloadSecuritySettings

Represents the security settings for uploading and downloading files. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

FileUploadAndDownloadSecuritySettings components have the suffix .settings and are stored in the settings folder.

Version

FileUploadAndDownloadSecuritySettings components are available in API version 39.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dispositions</td>
<td>FileTypeDispositionAssignmentBean[]</td>
<td>Represents the metadata used to manage filetype behavior. This field is available in API version 39.0 and later.</td>
</tr>
<tr>
<td>noHtmlUploadAsAttachment</td>
<td>boolean</td>
<td>Indicates whether to allow HTML uploads as attachments or document records. This field is available in API version 39.0 and later.</td>
</tr>
</tbody>
</table>
FileTypeDispositionAssignmentBean

Represents the metadata used to manage filetype behavior.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| behavior   | FileDownloadBehavior (enumeration of type string) | One of the following values:  
  - DOWNLOAD  
  - EXECUTE  
  - HYBRID  
  The following filetypes are a security risk and can not have EXECUTE behavior:  
  - EXE  
  - FLASH  
  - HTML  
  - RFC822  
  - SVG  
  - TXML  
  - UNKNOWN  
  - WEBVIEW  
  - XHTML  
  - XML |
| filetype   | FileType (enumeration of type string) | Although more filetypes exist, these are the only ones supported by FileTypeDispositionAssignmentBean:  
  - AVI  
  - EXCEL  
  - EXCEL_X  
  - EXE  
  - FLASH  
  - HTML  
  - MOV  
  - MP3  
  - MP4  
  - MPEG  
  - PDF  
  - POWER_POINT  
  - POWER_POINT_X  
  - RFC822  
  - SVG  
  - TXML  
  - UNKNOWN |
Declarative Metadata Sample Definition

The following is an example of a FileUploadAndDownloadSecuritySettings component.

```xml
<FileUploadAndDownloadSecuritySettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <dispositions>
    <behavior>HYBRID</behavior>
    <fileType>AVI</fileType>
    <securityRiskFileType>false</securityRiskFileType>
  </dispositions>
  <dispositions>
    <behavior>HYBRID</behavior>
    <fileType>WORD</fileType>
    <securityRiskFileType>false</securityRiskFileType>
  </dispositions>
  <dispositions>
    <behavior>HYBRID</behavior>
    <fileType>WORD_X</fileType>
    <securityRiskFileType>false</securityRiskFileType>
  </dispositions>
  <dispositions>
    <behavior>DOWNLOAD</behavior>
    <fileType>EXE</fileType>
    <securityRiskFileType>true</securityRiskFileType>
  </dispositions>
  <dispositions>
    <behavior>DOWNLOAD</behavior>
    <fileType>HTML</fileType>
    <securityRiskFileType>true</securityRiskFileType>
  </dispositions>
  <dispositions>
    <behavior>DOWNLOAD</behavior>
    <fileType>WEBVIEW</fileType>
    <securityRiskFileType>true</securityRiskFileType>
  </dispositions>
  <dispositions>
    <behavior>DOWNLOAD</behavior>
  </dispositions>
</FileUploadAndDownloadSecuritySettings>
```
<dispositions>
  <behavior>HYBRID</behavior>
  <fileType>MOV</fileType>
  <securityRiskFileType>false</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>HYBRID</behavior>
  <fileType>MP3</fileType>
  <securityRiskFileType>false</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>HYBRID</behavior>
  <fileType>MP4</fileType>
  <securityRiskFileType>false</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>HYBRID</behavior>
  <fileType>MPEG</fileType>
  <securityRiskFileType>false</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>HYBRID</behavior>
  <fileType>PDF</fileType>
  <securityRiskFileType>false</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>HYBRID</behavior>
  <fileType>POWER_POINT</fileType>
  <securityRiskFileType>false</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>HYBRID</behavior>
  <fileType>POWER_POINT_X</fileType>
  <securityRiskFileType>false</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>SVG</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>FLASH</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>TXML</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>RFC822</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MJPEG</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG2</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG4</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MP2</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MP3</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MP4</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG2</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG4</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MJPEG</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG2</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG4</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MJPEG</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG2</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG4</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MJPEG</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG2</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG4</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MJPEG</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG2</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MPEG4</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
<dispositions>
  <behavior>DOWNLOAD</behavior>
  <fileType>MJPEG</fileType>
  <securityRiskFileType>true</securityRiskFileType>
</dispositions>
The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>FileUploadAndDownloadSecurity</members>
    <name>Settings</name>
  </types>
  <version>39.0</version>
</Package>
```
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

ForecastingSettings

Represents the Collaborative Forecasts settings options. This type extends the Metadata metadata type and inherits its fullName field.

Note: This information only applies to Collaborative Forecasts.

File Suffix and Directory Location

ForecastingSettings values are stored in a single file named Forecasting.settings in the settings directory of the corresponding package directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

ForecastingSettings components are available in API version 28 and later. The structure of the ForecastingSettings type changed significantly in API version 30.0.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayCurrency</td>
<td>DisplayCurrency</td>
<td>The currency for displaying forecasts; either the organization’s corporate currency or each forecast owner’s personal currency setting. This is the default currency used in Collaborative Forecasts and selected in setup. The selection must be one of the currencies enabled for use in the organization, and only one selection is allowed. The default is Corporate. The valid values are: Corporate, Personal</td>
</tr>
<tr>
<td>enableForecasts</td>
<td>boolean</td>
<td>Indicates if Collaborative Forecasts is enabled or not. Set to true to enable Collaborative Forecasts and false to disable the functionality. Warning: Disabling Forecasts can result in data loss. Refer to the online Help before disabling any functionality.</td>
</tr>
<tr>
<td>forecastingCategoryMappings</td>
<td>ForecastCategoryMapping</td>
<td>A list of mappings associating forecast types with forecast rollups.</td>
</tr>
<tr>
<td>forecastingDisplayedFamilySettings</td>
<td>string</td>
<td>The product families chosen to allow forecasting on in Lightning Experience. This field is available in API version 40.0 and later.</td>
</tr>
</tbody>
</table>
### ForecastingSettings

The settings for each forecast type. An organization can have up to four forecast types active. Omitting a previously enabled forecast type that has a minimum API version less than or equal to the metadata package version deletes its quota and adjustment data from the organization.

**Warning:** Omitting a forecast type field from the XML can deactivate that forecast type: if the forecast type was available in the release specified by the XML package version, that forecast type is deactivated and its quota and adjustment data are deleted.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>forecastingTypeSettings</td>
<td>ForecastingTypeSettings</td>
<td>A list of forecast types. For field values, see ForecastingTypeSettings. The maximum number of forecast types is four.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Indicates whether the forecast type specified in the name field is active.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Setting the active field to false purges all forecasting data, adjustments, and quotas for the forecast type. When active is set to true, some values on the Forecasts tab may not appear immediately. An in-process icon appears to indicate that the values are being calculated.</td>
</tr>
<tr>
<td>adjustmentsSettings</td>
<td>AdjustmentsSettings</td>
<td>This enables or disables the Forecasts adjustments option in Forecasts.</td>
</tr>
</tbody>
</table>
| displayedCategoryApiNames | string | This read-only field appears four times to specify the four forecast rollup categories displayed in the Forecasts tab, for either cumulative forecast rollups, or individual forecast category rollups. Always use the same 4 values for both displayedCategoryApiNames and forecastedCategoryApiNames. Valid values for organizations using cumulative forecast rollups:  
  - openpipeline  
  - bestcaseforecast  
  - commitforecast  
  - closedonly  
  Valid values for organizations using individual forecast category rollups:  
  - pipelineonly  
  - bestcaseonly  
  - commitonly  
  - closedonly  

| forecastedCategoryApiNames | string | This field appears four times to specify the four forecast rollup categories used in the organization, for either cumulative forecast rollups, or individual forecast category rollups. Valid values for organizations using cumulative forecast rollups:  
  - openpipeline  

### Forecasting Settings Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bestcaseforecast</td>
<td></td>
<td></td>
</tr>
<tr>
<td>commitforecast</td>
<td></td>
<td></td>
</tr>
<tr>
<td>closedonly</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Valid values for organizations using individual forecast category rollups:

- pipelineonly
- bestcaseonly
- commitonly
- closedonly

Changing from one set of four values to the other changes the organization setting for Enable Cumulative Forecast Rollups in Setup. If this field is omitted, the setting is not changed.

<table>
<thead>
<tr>
<th>ForecastingDateType</th>
<th>ForecastingDateType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(enumeration of type string)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The date type that forecast amounts are based on in Collaborative Forecasts.

Valid values are:

- OpportunityCloseDate (default)
- ProductDate
- ScheduleDate

Available in API version 42.0 and later. In API version 42.0 only, date types are read only and available only via API.

<table>
<thead>
<tr>
<th>forecastRangeSettings</th>
<th>ForecastRangeSettings</th>
<th>The default periods and range selections in Collaborative Forecasts.</th>
</tr>
</thead>
<tbody>
<tr>
<td>hasProductFamily</td>
<td>boolean</td>
<td>Indicates whether the forecasting type has product family forecasts enabled. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>isAmount</td>
<td>boolean</td>
<td>This read-only field indicates whether the forecast type is based on revenue amounts. The value of <code>isAmount</code> is always the opposite of the value of <code>isQuantity</code>.</td>
</tr>
<tr>
<td>isAvailable</td>
<td>boolean</td>
<td>This read-only field indicates whether the forecast type can currently be used in the organization. For example, the revenue splits forecast type can’t be used in an organization that doesn’t have Opportunity Splits enabled.</td>
</tr>
<tr>
<td>isQuantity</td>
<td>boolean</td>
<td>This read-only field indicates whether the forecast type is based on product quantities. The value of <code>isQuantity</code> is always the opposite of the value of <code>isAmount</code>.</td>
</tr>
<tr>
<td>managerAdjustableCategoryApiNames</td>
<td>string</td>
<td>This read-only field appears twice to specify the two forecast rollup categories that forecast managers can adjust in the organization for either cumulative forecast rollups or individual forecast category rollups. This field can only be used when the <code>enableAdjustments</code> field contains a value of true. If both the <code>managerAdjustableCategoryApiNames</code> and <code>ownerAdjustableCategoryApiNames</code> fields are being used,</td>
</tr>
</tbody>
</table>
they must contain the same two values. Their values must also be consistent with the values of the enableAdjustments and enableOwnerAdjustments fields.

Valid values for organizations using cumulative forecast rollups:

- bestcaseforecast
- commitforecast

Valid values for organizations using individual forecast category rollups:

- bestcaseonly
- commitonly

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>This read-only field indicates the UI label for the forecast type.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the forecast type. Each forecast type requires a specific string.</td>
</tr>
</tbody>
</table>

Valid values include:

- **LineItemQuantityProductDate**: Product Families - Quantity by product date. Available in API version 43.0 and later. Reserved for future use.
- **LineItemQuantityScheduleDate**: Product Families - Quantity by schedule date. Available in API version 43.0 and later. Reserved for future use.
- **LineItemRevenueProductDate**: Product Families - Revenue by product date. Available in API version 43.0 and later. Reserved for future use.
- **LineItemRevenueScheduleDate**: Product Families - Revenue by schedule date. Available in API version 43.0 and later. Reserved for future use.
- **OpportunityLineItemQuantity**: Product Families - Quantity
- **OpportunityLineItemRevenue**: Product Families - Revenue
- **OpportunityOverlayRevenue**: Opportunity Overlay Splits - Revenue
- **OpportunityQuantity**: Opportunities - Quantity
- **OpportunityQuantityProductDate**: Opportunities - Quantity by product date. Available in API version 43.0 and later.
- **OpportunityQuantityScheduleDate**: Opportunities - Quantity by schedule date. Available in API version 43.0 and later.
- **OpportunityRevenue**: Opportunities - Revenue
- **OpportunityRevenueProductDate**: Opportunities - Revenue by product date. Available in API version 43.0 and later.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• OpportunityRevenueScheduleDate</td>
<td>string</td>
<td>Revenue by schedule date. Available in API version 43.0 and later.</td>
</tr>
<tr>
<td>• OpportunitySplitRevenue</td>
<td>string</td>
<td>Opportunity Revenue Splits - Revenue</td>
</tr>
<tr>
<td>• <strong>Territory_Model_NameN</strong></td>
<td>string</td>
<td>Territories - Revenue, where <strong>Territory_Model_Name</strong> is the name of your active territory model in the API and can be followed by ( N ), an auto-generated number. Available in API version 44.0 and later.</td>
</tr>
<tr>
<td>• <strong>Territory_Model_NameN</strong>_ProductFamily</td>
<td>string</td>
<td>Territories - Product Families, where <strong>Territory_Model_Name</strong> is the name of your active territory model in the API and can be followed by ( N ), an auto-generated number. Available in API version 45.0 and later.</td>
</tr>
<tr>
<td>• The name of a custom opportunity split type that has been enabled as a forecast type. Custom split types are based on currency fields, which can contain revenue amounts only.</td>
<td>string</td>
<td></td>
</tr>
</tbody>
</table>

| opportunityListFieldsLabelMappings | string       | A read-only list of the API names and UI labels for all fields on the Opportunity object.                                                                                         |
| opportunityListFieldsSelectedSettings | string       | The fields selected to appear in the opportunity pane of the forecast page for the forecast type. **Opportunity Name** is required. You can select up to 15 fields. |
| opportunityListFieldsUnselectedSettings | string       | The fields not selected to appear in the opportunity pane of the forecast page for the forecast type.                                                                           |
| opportunitySplitName             | string       | Indicates whether the forecasting type has a split type, and if so, the name of the split type. Available in API version 41.0 and later. |
| ownerAdjustableCategoryApiNames   | string       | This read-only field appears twice to specify the two forecast rollup categories that forecast owners can adjust in the organization, for either cumulative forecast rollups, or individual forecast category rollups. This field can only be used when the enableOwnerAdjustments field contains a value of true. If both the managerAdjustableCategoryApiNames and ownerAdjustableCategoryApiNames fields are being used, they must contain the same two values. Their values must also be consistent with the values of the enableAdjustments and enableOwnerAdjustments fields. Valid values for organizations using cumulative forecast rollups:  
  • bestcaseforecast  
  • commitforecast  
  Valid values for organizations using individual forecast category rollups:  
  • bestcaseonly  
  • commitonly  

789
**Field Name**: quotasSettings  
**Field Type**: QuotasSettings  
**Description**: This enables or disables the quota option in Forecasts.

**Field Name**: territory2ModelName  
**Field Type**: string  
**Description**: Indicates whether the forecasting type has a Territory2 model, and if so, the name of the Territory2 model. Available in API version 41.0 and later.

---

**AdjustmentsSettings**

The adjustment options for Collaborative Forecasts.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableAdjustments</td>
<td>boolean</td>
<td>Set to <code>true</code> to enable Collaborative Forecasts manager adjustments and <code>false</code> to disable them. All forecast types must contain the same <code>enableAdjustments</code> value. <strong>Warning</strong>: Disabling adjustments results in Collaborative Forecasts adjustment data being purged.</td>
</tr>
<tr>
<td>enableOwnerAdjustments</td>
<td>boolean</td>
<td>Set to <code>true</code> to enable Collaborative Forecasts owner adjustments and <code>false</code> to disable them. All forecast types must contain the same <code>enableAdjustments</code> value. <strong>Warning</strong>: Disabling adjustments results in Collaborative Forecasts adjustment data being purged.</td>
</tr>
</tbody>
</table>

---

**ForecastRangeSettings**

The default periods and range selections in Collaborative Forecasts. Users can forecast up to 15 months, 15 fiscal periods, or 8 quarters in the future or past. If your forecast range includes the current month, period, or quarter, the forecasts page displays the current month, period, or quarter by default. If not, the first month, period, or quarter of the range is selected. All forecast types must contain the same `forecastRangeSettings` field values. **Warning**: If you change the time period from monthly to quarterly or quarterly to monthly, or you change the standard fiscal year, all adjustments and quotas are purged. If you enable custom fiscal years, creating the first custom fiscal year deletes any quotas and adjustments in the corresponding and subsequent standard fiscal years. These changes trigger a forecast recalculation that can take significant time, depending on the quantity of your data.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>beginning</td>
<td>int</td>
<td>Indicates the beginning month or quarter to display by default.</td>
</tr>
<tr>
<td>displaying</td>
<td>int</td>
<td>Indicates the number of months or quarters to display by default. The maximum number of months is 12 and quarters is 8.</td>
</tr>
</tbody>
</table>
| periodType| PeriodTypes (enumeration of type string) | Indicates what type of period to use. Valid values:  
  - Month  
  - Quarter |
OpportunityListFieldsLabelMappings
A read-only list of the API names and UI labels for all fields on the Opportunity object.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>The API name of the Opportunity field.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The UI label of the Opportunity field.</td>
</tr>
</tbody>
</table>

OpportunityListFieldsSelectedSettings
The fields selected to appear in the opportunity pane of the forecast page for the forecast type. Opportunity Name is required. You can select up to 15 fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>Specifies names of fields to display in the opportunity pane.</td>
</tr>
</tbody>
</table>

OpportunityListFieldsUnselectedSettings
The fields not selected to appear in the opportunity pane of the forecast page for the forecast type.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>Specifies names of fields not displayed in the opportunity pane.</td>
</tr>
</tbody>
</table>

QuotasSettings
QuotasSettings indicates if quotas are available in Collaborative Forecasts.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showQuotas</td>
<td>boolean</td>
<td>Set to true to enable quotas. All forecast types must contain the same showQuotas field value.</td>
</tr>
</tbody>
</table>

ForecastingCategoryMappings
The forecasting category mappings for Collaborative Forecasts. This subtype appears eight times within the ForecastingSettings type. Each occurrence includes fields that specify a type of forecast category rollup, which forecast categories each rollup includes, and the weight of each forecast category in the rollup. Organizations using either cumulative forecast rollups or individual forecast category columns must include all eight occurrences of this subtype.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>forecastingItemCategoryApiName</td>
<td>string</td>
<td>This field specifies the API name of the rollup type. The valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• openpipeline</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>bestcaseforecast</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>commitforecast</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>pipelineonly</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>bestcaseonly</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>commitonly</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>closedonly</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>omittedonly</code></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**WeightedSourceCategories**

This field can occur more than once when specifying more than one forecast category to include in the rollup type. Each occurrence contains two subfields that specify a forecast category to include in the forecast rollup type and its weight. Some rollup types include more than one forecast category. This list shows the forecast categories that are included in each rollup type.

- Rollup: openpipeline, Forecast categories: pipeline, best case, commit
- Rollup: bestcaseforecast, Forecast categories: best case, commit, closed
- Rollup: commitforecast, Forecast categories: commit, closed
- Rollup: pipelineonly, Forecast categories: pipeline
- Rollup: bestcaseonly, Forecast categories: best case
- Rollup: commitonly, Forecast categories: commit
- Rollup: closedonly, Forecast categories: closed
- Rollup: omittedonly, Forecast categories: omitted

**sourceCategoryApiName**

Specifies the API name of a forecast category to include in the rollup type. The valid values are.

- pipeline
- best case
- commit
- closed
- omitted
### ForecastingDisplayedFamilySettings

The product families that an admin chooses to allow forecasting on in Lightning Experience. This field is available in API version 40.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>productFamily</td>
<td>string</td>
<td>The product family available to forecast on. Each product family is unique.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of a ForecastingSettings component that enables the Opportunity-Revenue and Product Family-Quantity forecast types:

```xml
<forecastingSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableForecasts>true</enableForecasts>
  <forecastingTypeSettings>
    <active>true</active>
    <adjustmentsSettings>
      <enableAdjustments>true</enableAdjustments>
    </adjustmentsSettings>
    <name>OpportunityRevenue</name>
    <forecastRangeSettings>
      <beginning>0</beginning>
      <displaying>6</displaying>
      <periodType>Month</periodType>
    </forecastRangeSettings>
    <opportunityListFieldsSelectedSettings>
      <field>OPPORTUNITY.NAME</field>
    </opportunityListFieldsSelectedSettings>
    <quotasSettings>
      <showQuotas>true</showQuotas>
    </quotasSettings>
  </forecastingTypeSettings>
  <forecastingTypeSettings>
    <active>false</active>
    <adjustmentsSettings>
      <enableAdjustments>true</enableAdjustments>
    </adjustmentsSettings>
    <name>OpportunityLineItemQuantity</name>
    <forecastRangeSettings>
      <beginning>0</beginning>
    </forecastRangeSettings>
  </forecastingTypeSettings>
</forecastingSettings>
```
<displaying>6</displaying>
<periodType>Month</periodType>
</forecastRangeSettings>
<opportunityListFieldsSelectedSettings>
  <field>OPPORTUNITY.NAME</field>
</opportunityListFieldsSelectedSettings>
<quoteSettings>
  <showQuotas>true</showQuotas>
</quoteSettings>
<displayedCategoryApiNames>
  pipelineonly
  bestcaseonly
  commitonly
  closedonly
</displayedCategoryApiNames>
<forecastedCategoryApiNames>
  commitonly
  closedonly
  bestcaseonly
  pipelineonly
</forecastedCategoryApiNames>
<managerAdjustableCategoryApiNames>
  commitonly
  closedonly
  bestcaseonly
  pipelineonly
</managerAdjustableCategoryApiNames>
<ownerAdjustableCategoryApiNames>
  commitonly
  closedonly
  bestcaseonly
  pipelineonly
</ownerAdjustableCategoryApiNames>
</forecastingTypeSettings>
<forecastingCategoryMappings>
  <forecastingItemCategoryApiName>commitonly</forecastingItemCategoryApiName>
  <weightedSourceCategories>
    <sourceCategoryApiName>commit</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
</forecastingCategoryMappings>
<forecastingCategoryMappings>
  <forecastingItemCategoryApiName>closedonly</forecastingItemCategoryApiName>
  <weightedSourceCategories>
    <sourceCategoryApiName>closed</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
</forecastingCategoryMappings>
<forecastingCategoryMappings>
  <forecastingItemCategoryApiName>bestcaseforecast</forecastingItemCategoryApiName>
  <weightedSourceCategories>
    <sourceCategoryApiName>commit</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
  <weightedSourceCategories>
    <sourceCategoryApiName>best case</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
  <weightedSourceCategories>
    <sourceCategoryApiName>closed</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
</forecastingCategoryMappings>

794
<forecastingCategoryMappings>
  <forecastingItemCategoryApiName>omittedonly</forecastingItemCategoryApiName>
  <weightedSourceCategories>
    <sourceCategoryApiName>omitted</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
</forecastingCategoryMappings>

<forecastingCategoryMappings>
  <forecastingItemCategoryApiName>openpipeline</forecastingItemCategoryApiName>
  <weightedSourceCategories>
    <sourceCategoryApiName>commit</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
  <weightedSourceCategories>
    <sourceCategoryApiName>best case</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
  <weightedSourceCategories>
    <sourceCategoryApiName>pipeline</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
</forecastingCategoryMappings>

<forecastingCategoryMappings>
  <forecastingItemCategoryApiName>bestcaseonly</forecastingItemCategoryApiName>
  <weightedSourceCategories>
    <sourceCategoryApiName>best case</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
</forecastingCategoryMappings>

<forecastingCategoryMappings>
  <forecastingItemCategoryApiName>commitforecast</forecastingItemCategoryApiName>
  <weightedSourceCategories>
    <sourceCategoryApiName>closed</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
  <weightedSourceCategories>
    <sourceCategoryApiName>commit</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
</forecastingCategoryMappings>

<forecastingCategoryMappings>
  <forecastingItemCategoryApiName>pipelineonly</forecastingItemCategoryApiName>
  <weightedSourceCategories>
    <sourceCategoryApiName>pipeline</sourceCategoryApiName>
    <weight>1.0</weight>
  </weightedSourceCategories>
</forecastingCategoryMappings>
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
  Settings

IdeasSettings

Represents the metadata used to manage settings for Ideas.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

File Suffix and Directory Location

IdeasSettings is stored in one file named Ideas.settings in the settings folder of the corresponding package directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

IdeasSettings is available in API version 27.0 and later.

Ideas

Represents settings for Ideas and Idea Themes.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableIdeaThemes</td>
<td>boolean</td>
<td>Indicates whether Idea Themes is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableIdeas</td>
<td>boolean</td>
<td>Indicates whether Ideas is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableIdeasReputation</td>
<td>boolean</td>
<td>Indicates whether Reputation is enabled (true) or not (false). You can’t enable IdeasReputation without enabling the Ideas Reputation permission in your organization. This field is available in API version 28.0 and later.</td>
</tr>
<tr>
<td>enableChatterProfile</td>
<td>boolean</td>
<td>Indicates that the Chatter user profile is used for Ideas user profiles. If enableChatterProfile is true, the ideasProfilePage value must not be specified. If enableChatterProfile is false, then specify a ideasProfilePage value, otherwise the Ideas zone profile is used. This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>ideasProfilePage</td>
<td>string</td>
<td>The name of the Visualforce page to use for a custom Ideas user profile. If enableChatterProfile is false, then specify a</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
ideasProfilePage | | The ideasProfilePage value, otherwise the ideas zone profile is used. This field is available in API version 29.0 and later.
halfLife | double | Indicates how quickly old ideas drop in ranking on the Popular Ideas subtab. The half-life setting determines how the number of days after which old ideas drop in ranking on the Popular Ideas subtab, to make room for ideas with more recent votes. A shorter half-life moves older ideas down the page faster than a longer half-life.

---

### Declarative Metadata Sample Definition

The following is an example ideas.settings metadata file:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<IdeasSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableIdeaThemes>true</enableIdeaThemes>
  <enableIdeas>true</enableIdeas>
  <enableIdeasReputation>true</enableIdeasReputation>
  <enableChatterProfile>false</enableChatterProfile>
  <ideasProfilePage>name of Visualforce page</ideasProfilePage>
  <halfLife>2.6</halfLife>
</IdeasSettings>
```

---

### Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
- Settings

---

### IoTSettings

Represents the organization’s IoT settings, such as whether or not IoT or IoT Insights is enabled. This type extends the Metadata metadata type and inherits its fullName field.

---

### File Suffix and Directory Location

IoTSettings components have the suffix IoT.settings and are stored in the settings folder. The .settings files are different from other named components because there is only one settings file for each settings component. In the package manifest, all organization settings metadata types are accessed using the Settings name.

---

### Version

IoTSettings components are available in API version 44.0 and later.
**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableIoT</td>
<td>boolean</td>
<td>Indicates whether IoT is enabled (true) or not (false).</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example of a IoTSettings component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<IoTSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableIoT>true</enableIoT>
</IoTSettings>
```

**Wildcard Support in the Manifest File**

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**KnowledgeSettings**

Represents the metadata used to manage settings for Salesforce Knowledge.

This type extends the Metadata metadata type and inherits its fullName field.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

**File Suffix and Directory Location**

KnowledgeSettings values are stored in a single file named Knowledge.settings in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

**Version**

KnowledgeSettings is available in API version 27.0 and later.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>answers</td>
<td>KnowledgeAnswerSettings</td>
<td>Represents the metadata used to manage settings for Salesforce Knowledge and Answers.</td>
</tr>
<tr>
<td>cases</td>
<td>KnowledgeCaseSettings</td>
<td>Represents the metadata used to manage settings for Salesforce Knowledge and Cases.</td>
</tr>
<tr>
<td>defaultLanguage</td>
<td>string</td>
<td>Required. The default language for Salesforce Knowledge. Use the abbreviation for the</td>
</tr>
</tbody>
</table>
# Metadata Types

## KnowledgeSettings

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableChatterQuestionKBDeflection</td>
<td>boolean</td>
<td>Indicates whether tracking for case deflection via Chatter is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableCreateEditOnArticlesTab</td>
<td>boolean</td>
<td>Indicates whether users can create and edit articles on the articles tab (true) or not (false).</td>
</tr>
<tr>
<td>enableExternalMediaContent</td>
<td>boolean</td>
<td>Indicates whether connecting to external media is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableKnowledge</td>
<td>boolean</td>
<td>Indicates whether Salesforce Knowledge is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableLightningKnowledge</td>
<td>boolean</td>
<td>Indicates whether Lightning Knowledge is enabled (true) or not (false).</td>
</tr>
<tr>
<td>languages</td>
<td>KnowledgeLanguageSettings</td>
<td>A list of languages enabled for Salesforce Knowledge.</td>
</tr>
<tr>
<td>showArticleSummariesCustomerPortal</td>
<td>boolean</td>
<td>Indicates whether article summaries appear in the Customer Portal (true) or not (false).</td>
</tr>
<tr>
<td>showArticleSummariesInternalApp</td>
<td>boolean</td>
<td>Indicates whether article summaries appear in the internal knowledge base (true) or not (false).</td>
</tr>
<tr>
<td>showArticleSummariesPartnerPortal</td>
<td>boolean</td>
<td>Indicates whether article summaries appear in the partner portal (true) or not (false).</td>
</tr>
<tr>
<td>showValidationStatusField</td>
<td>boolean</td>
<td>Indicates whether validation status appears on articles (true) or not (false).</td>
</tr>
<tr>
<td>suggestedArticles</td>
<td>KnowledgeSuggestedArticlesSettings</td>
<td>Represents the metadata used to manage settings for the case fields used to suggest articles for cases. Available in API version 37.0 and later.</td>
</tr>
</tbody>
</table>

## KnowledgeAnswerSettings

Represents the metadata used to manage settings for Salesforce Knowledge and Answers.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignTo</td>
<td>string</td>
<td>Specifies the username an article is assigned to from Answers.</td>
</tr>
<tr>
<td>defaultArticleType</td>
<td>string</td>
<td>The default article type for articles created from Answers. Uses the API name of the article type.</td>
</tr>
<tr>
<td>enableArticleCreation</td>
<td>boolean</td>
<td>Indicates whether users can create articles from Answers (true) or not (false).</td>
</tr>
</tbody>
</table>
### KnowledgeCaseSettings

Represents the metadata used to manage settings for Salesforce Knowledge and Cases.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>articlePDFCreationProfile</td>
<td>string</td>
<td>The profile used to create a PDF of an article from Cases.</td>
</tr>
<tr>
<td>articlePublicSharingSites</td>
<td>KnowledgeSitesSettings</td>
<td>Represents the metadata used to manage settings for Salesforce Knowledge and Sites.</td>
</tr>
<tr>
<td>articlePublicSharingCommunities</td>
<td>KnowledgeSitesSettings</td>
<td>Represents the metadata used to manage settings for Salesforce Knowledge and Communities.</td>
</tr>
<tr>
<td>articlePublicSharingSitesChatterAnswers</td>
<td>KnowledgeSitesSettings</td>
<td>Represents the metadata used to manage settings for Salesforce Knowledge and Sites with Chatter Answers.</td>
</tr>
<tr>
<td>assignTo</td>
<td>string</td>
<td>Specifies the username an article is assigned to from Cases.</td>
</tr>
<tr>
<td>customizationClass</td>
<td>string</td>
<td>Specifies the Apex class used for customization.</td>
</tr>
<tr>
<td>defaultContributionArticleType</td>
<td>string</td>
<td>The default article type for articles created from Cases.</td>
</tr>
</tbody>
</table>
| editor                              | KnowledgeCaseEditor (enumeration of type string) | Indicates the rich text editor type. Valid values:  
• simple  
• standard |
| enableArticleCreation               | boolean         | Indicates whether users can create articles from Cases (true) or not (false). Controls whether other fields on KnowledgeCaseSettings can be set. |
| enableArticlePublicSharingSites     | boolean         | Indicates whether articles can be shared via a public site (URL) from Cases (true) or not (false). |
| enableCaseDataCategoryMapping       | boolean         | Indicates whether Case Data Category mapping is enabled (true) or not (false). |
| useProfileForPDFCreation            | boolean         | Indicates whether a profile is used to create a PDF of an article from Cases (true) or not (false). |

### KnowledgeSitesSettings

Represents the metadata used to manage settings for Salesforce Knowledge and Sites.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>site</td>
<td>string[]</td>
<td>Specifies the site used for Salesforce Knowledge and Sites.</td>
</tr>
</tbody>
</table>
## KnowledgeLanguageSettings

A list of languages enabled for Salesforce Knowledge. KnowledgeLanguageSettings is available in API version 28.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>language</td>
<td>KnowledgeLanguage</td>
<td>Represents the metadata used to manage settings for the languages enabled for Salesforce Knowledge.</td>
</tr>
</tbody>
</table>

## KnowledgeLanguage

Represents the metadata used to manage settings for the languages enabled for Salesforce Knowledge. KnowledgeLanguage is available in API version 28.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Indicates whether the language is enabled (true) or not (false).</td>
</tr>
<tr>
<td>defaultAssignee</td>
<td>string</td>
<td>The default assignee for articles in the language.</td>
</tr>
<tr>
<td>defaultAssigneeType</td>
<td>KnowledgeLanguageLookupValueTypeDef</td>
<td>Indicates the default assignee type. Valid values:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• User</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Queue</td>
</tr>
<tr>
<td>defaultReviewer</td>
<td>string</td>
<td>The default reviewer for articles in the language.</td>
</tr>
<tr>
<td>defaultReviewerType</td>
<td>KnowledgeLanguageLookupValueTypeDef</td>
<td>Indicates the default reviewer type. Valid values:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• User</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Queue</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The code for the language name, for example: English is en. See “What languages does Salesforce support?” in the Salesforce online help for a list of supported languages and their codes.</td>
</tr>
</tbody>
</table>

## KnowledgeSuggestedArticlesSettings

Represents the metadata used to manage settings for the articles suggested for cases, work orders, and work order line items. The Work Order and Work Order Line Item objects must be enabled in the org to use the associated fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>caseFields</td>
<td>KnowledgeCaseFieldsSettings</td>
<td>Represents a list of the case fields used to suggest articles for the case.</td>
</tr>
<tr>
<td>useSuggestedArticlesForCase</td>
<td>boolean</td>
<td>Indicates whether case content is used to suggest articles for cases (true) or not (false).</td>
</tr>
</tbody>
</table>
### KnowledgeWorkOrderFieldsSettings

Represents a list of the work order fields used to suggest articles for the work order. Available in API version 39.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workOrderFields</td>
<td>KnowledgeWorkOrderFieldsSettings</td>
<td>Represents a list of the work order fields used to suggest articles for the work order.</td>
</tr>
<tr>
<td>workOrderLineItemFields</td>
<td>KnowledgeWorkOrderLineItemFieldsSettings</td>
<td>Represents a list of the work order line item fields used to suggest articles for the work order line item.</td>
</tr>
</tbody>
</table>

### KnowledgeCaseFieldsSettings

Represents a list of the case fields used to suggest articles for the case. Available in API version 37.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>KnowledgeCaseField[]</td>
<td>Specifies the names of the case fields used to suggest articles for the case.</td>
</tr>
</tbody>
</table>

### KnowledgeCaseField

Represents the name of the case field used to suggest articles for the case. Available in API version 37.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Specifies the name of the case field used to suggest articles for the case.</td>
</tr>
</tbody>
</table>

### KnowledgeWorkOrderFieldsSettings

Represents a list of the work order fields used to suggest articles for the work order. Available in API version 39.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>KnowledgeWorkOrderField[]</td>
<td>Specifies the names of the work order fields used to suggest articles for the work order.</td>
</tr>
</tbody>
</table>

### KnowledgeWorkOrderField

Represents the name of the work order field used to suggest articles for the work order. Available in API version 39.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Specifies the name of the work order field used to suggest articles for the work order.</td>
</tr>
</tbody>
</table>

### KnowledgeWorkOrderLineItemFieldsSettings

Represents a list of the work order line item fields used to suggest articles for the work order line item. Available in API version 39.0 and later.
### KnowledgeWorkOrderLineItemField

Represents the name of the work order line item field used to suggest articles for the work order line item. Available in API version 39.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Specifies the name of the work order line item field used to suggest articles for the work order line item.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

This is a sample Knowledge settings file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<KnowledgeSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <answers>
    <enableArticleCreation>false</enableArticleCreation>
  </answers>
  <cases>
    <articlePDFCreationProfile>partner portal knowledge profile</articlePDFCreationProfile>
    <articlePublicSharingSites>
      <site>KnowledgeSite</site>
      <site>PKB2Site</site>
      <site>ChatterAnswersSite</site>
    </articlePublicSharingSites>
    <articlePublicSharingSitesChatterAnswers>
      <site>ChatterAnswersSite</site>
    </articlePublicSharingSitesChatterAnswers>
    <assignTo>testall@kb.org</assignTo>
    <defaultContributionArticleType>Support</defaultContributionArticleType>
    <editor>simple</editor>
    <enableArticleCreation>true</enableArticleCreation>
    <enableArticlePublicSharingSites>true</enableArticlePublicSharingSites>
    <useProfileForPDFCreation>true</useProfileForPDFCreation>
  </cases>
  <defaultLanguage>ja</defaultLanguage>
  <enableCreateEditOnArticlesTab>true</enableCreateEditOnArticlesTab>
  <enableExternalMediaContent>true</enableExternalMediaContent>
  <enableKnowledge>true</enableKnowledge>
  <showArticleSummariesCustomerPortal>true</showArticleSummariesCustomerPortal>
  <showArticleSummariesInternalApp>true</showArticleSummariesInternalApp>
  <showArticleSummariesPartnerPortal>true</showArticleSummariesPartnerPortal>
  <showValidationStatusField>true</showValidationStatusField>
  <suggestedArticles>
    <caseFields>
```

803
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
Settings

LeadConvertSettings

Represents an organization’s custom field mappings for lead conversion. Custom fields can be mapped from Leads to Accounts, Contacts, and Opportunities. Options for creating opportunities during lead conversion can also be specified. This type extends the Metadata metadata type and inherits its fullName field.

Version

LeadConvertSettings is available in API versions 39.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowOwnerChange</td>
<td>boolean</td>
<td>Indicates whether to include the RecordOwner field in the Convert Lead dialog box (true) or not (false).</td>
</tr>
<tr>
<td>objectMapping</td>
<td>metadata type</td>
<td>A set of inputObject, mappingFields, and outputObject entries. Up to three objectMapping types can be declared—one each for Account, Contact, and Opportunity.</td>
</tr>
<tr>
<td>inputObject</td>
<td>string</td>
<td>The name of the object type containing the source fields for mapping. The value will always be Lead.</td>
</tr>
<tr>
<td>mappingFields</td>
<td>metadata type</td>
<td>A set of inputField and outputField entries.</td>
</tr>
<tr>
<td>inputField</td>
<td>string</td>
<td>The name of a custom lead field supplying source data during lead conversion.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>outputField</td>
<td>string</td>
<td>The name of a custom account, contact, or opportunity field that will receive data from source field named in the accompanying inputField entry.</td>
</tr>
<tr>
<td>outputObject</td>
<td>string</td>
<td>The name of the object type receiving data during lead conversion—Account, Contact, or Opportunity.</td>
</tr>
<tr>
<td>opportunityCreationOptions</td>
<td>string</td>
<td>This optional field determines whether the Opportunity field is visible or required in the Convert Lead dialog box. Valid values include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VisibleOptional—The Opportunity field is included in the dialog box but not required. A new opportunity is created if the user enters an opportunity name. This is the default value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VisibleRequired—The Opportunity field is included in the dialog box and is required. A new opportunity is created based on the name entered by the user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NotVisible—The Opportunity field is not included in the dialog box. No opportunity is created.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of the LeadConvertSettings type:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LeadConvertSettings xmlns="http://soap.sforce.com/2006/04/metadata">
    <allowOwnerChange>false</allowOwnerChange>
    <objectMapping>
        <inputObject>Lead</inputObject>
        <mappingFields>
            <inputField>custom_lead_field_1</inputField>
            <outputField>custom_account_field_1</outputField>
        </mappingFields>
        <mappingFields>
            <inputField>custom_lead_field_2</inputField>
            <outputField>custom_account_field_2</outputField>
        </mappingFields>
        <mappingFields>
            <inputField>custom_lead_field_3</inputField>
            <outputField>custom_account_field_3</outputField>
        </mappingFields>
        <outputObject>Account</outputObject>
    </objectMapping>
    <objectMapping>
        <inputObject>Lead</inputObject>
        <mappingFields>
            <inputField>custom_lead_field_4</inputField>
            <outputField>custom_opportunity_field_1</outputField>
        </mappingFields>
        <outputObject>Opportunity</outputObject>
    </objectMapping>
</LeadConvertSettings>
```
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

LiveAgentSettings

Represents an organization’s Chat settings, such as whether or not Chat is enabled. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

LiveAgentSettings values are stored in the LiveAgent.settings file in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

Version

LiveAgentSettings is available in API version 28.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableLiveAgent</td>
<td>boolean</td>
<td>Indicates whether Chat is enabled (true) or not (false).</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

This is a sample Chat settings file.

```
<?xml version="1.0" encoding="UTF-8"?>
<LiveAgentSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableLiveAgent>true</enableLiveAgent>
</LiveAgentSettings>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
**LiveMessageSettings**

Represents an org’s LiveMessage settings.

**Version**

LiveMessageSettings components are available in API version 42.0 and later.

**File Suffix and Directory Location**

LiveMessageSettings values are stored in the LiveMessage.settings file in the settings folder. The .settings files are different from other named components because there is only one settings file for each settings component.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableLiveMessage</td>
<td>boolean</td>
<td>Turns LiveMessage on (true) or off (false) in an org. The default value is false.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example of a `liveMessageSettings` component.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<LiveMessageSettings xmlns="http://soap.sforce.com/2006/04/metadata">
    <enableLiveMessage>true</enableLiveMessage>
</LiveMessageSettings>
```

The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>LiveMessage</members>
        <name>Settings</name>
    </types>
    <version>44.0</version>
</Package>
```

**Wildcard Support in the Manifest File**

The wildcard character `*` (asterisk) in the `package.xml` manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see [Settings](#). For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](#).

**MacroSettings**

Represents an organization’s Macro settings, such as whether or not folders is enabled. This type extends the `Metadata` metadata type and inherits its `fullName` field.
File Suffix and Directory Location

MacroSettings values are stored in the Macro.settings file in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

Version

MacroSettings is available in API version 39.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableAdvancedSearch</td>
<td>boolean</td>
<td>Indicates whether users can search all macro text fields (true) or not (false).</td>
</tr>
<tr>
<td>macrosInFolders</td>
<td>boolean</td>
<td>Indicates whether users can organize and share macros using folders (true) or not (false). Available in API version 44.0 and later.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

This is a sample Macro settings file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<MacroSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableAdvancedSearch>true</enableAdvancedSearch>
  <macrosInFolders>true</macrosInFolders>
</MacroSettings>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

MapAndLocationSettings

Represents an org’s map and location settings.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

Declarative Metadata File Suffix and Directory Location

MapAndLocationSettings values are stored in a single file named Mapandlocation.settings in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.
Version

Map and location settings are available in API version 46.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableAddressAutoComplete</td>
<td>boolean</td>
<td>Indicates whether auto-complete is enabled on address fields (true) or not (false).</td>
</tr>
<tr>
<td>enableMapsAndLocation</td>
<td>boolean</td>
<td>Indicates whether the maps and location services are enabled (true) or not (false)</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

This is a sample `mapandlocation.settings` metadata file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<MapsAndLocationSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableAddressAutoComplete>false</enableAddressAutoComplete>
  <enableMapsAndLocation>false</enableMapsAndLocation>
</MapsAndLocationSettings>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the `package.xml` manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

MobileSettings

Represents an organization’s mobile settings.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

Declarative Metadata File Suffix and Directory Location

MobileSettings values are stored in a single file named `Mobile.settings` in the `settings` directory. The `.settings` files are different from other named components because there is only one settings file for each settings component.

Note: MobileSettings is no longer available in API versions 25.0 and 26.0.

Version

Mobile settings are available in API version 27.0 and later.
## Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chatterMobile (Deprecated)</td>
<td>ChatterMobileSettings</td>
<td>The settings for devices running Chatter mobile.</td>
</tr>
<tr>
<td>dashboardMobile (Deprecated)</td>
<td>DashboardMobileSettings</td>
<td>The settings for devices running the mobile dashboards app.</td>
</tr>
<tr>
<td>touchMobile (Deprecated)</td>
<td>TouchMobileSettings</td>
<td>The settings for devices running Salesforce Touch.</td>
</tr>
</tbody>
</table>

### ChatterMobileSettings

These fields are deprecated. Represents your organization’s Chatter Mobile settings.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPadAuthorized</td>
<td>boolean</td>
<td>Indicates whether iPad devices are enabled for Chatter Mobile (true) or not (false).</td>
</tr>
<tr>
<td>IPhoneAuthorized</td>
<td>boolean</td>
<td>Indicates whether iPhone devices are enabled for Chatter Mobile (true) or not (false).</td>
</tr>
<tr>
<td>androidAuthorized</td>
<td>boolean</td>
<td>Indicates whether Android devices are enabled for Chatter Mobile (true) or not (false).</td>
</tr>
<tr>
<td>blackBerryAuthorized</td>
<td>boolean</td>
<td>Indicates whether Blackberry devices are enabled for Chatter Mobile (true) or not (false).</td>
</tr>
<tr>
<td>enableChatterMobile</td>
<td>boolean</td>
<td>Indicates whether Chatter Mobile has been enabled for your organization (true) or not (false).</td>
</tr>
<tr>
<td>enablePushNotifications</td>
<td>boolean</td>
<td>Indicates whether Chatter push notifications have been enabled for your organization (true) or not (false).</td>
</tr>
</tbody>
</table>

**Note:** Setting this to true enables you to set all of the other settings. If you change this setting from true to false, and also try to change any of the other ChatterMobile settings, your deploy will fail with an error.
### Metadata Types

#### MobileSettings

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| sessionTimeout | MobileSessionTimeout (enumeration of type string) | The length of time after which users without activity are prompted to log out or continue working. Valid values are:  
- Never  
- OneMinute  
- FiveMinutes  
- TenMinutes  
- ThirtyMinutes |

#### DashboardMobileSettings

These fields are deprecated. Represents your organization’s Mobile Dashboards iPad app settings.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableDashboardIPadApp</td>
<td>boolean</td>
<td>Indicates whether Mobile Dashboards iPad app has been enabled for your organization (true) or not (false)</td>
</tr>
</tbody>
</table>

#### TouchMobileSettings

These fields are deprecated. Salesforce Touch has been upgraded to the Salesforce app.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableTouchBrowserIPad</td>
<td>boolean</td>
<td>Indicates whether your organization has the Salesforce Touch mobile browser app enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableTouchAppIPad</td>
<td>boolean</td>
<td>Indicates whether your organization has the Salesforce Touch downloadable app enabled (true) or not (false).</td>
</tr>
</tbody>
</table>

#### Declarative Metadata Sample Definition

This is a sample mobile.settings metadata file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<MobileSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <chatterMobile>
    <IPadAuthorized>true</IPadAuthorized>
    <IPhoneAuthorized>true</IPhoneAuthorized>
    <androidAuthorized>true</androidAuthorized>
    <blackBerryAuthorized>true</blackBerryAuthorized>
    <enableChatterMobile>true</enableChatterMobile>
  </chatterMobile>
</MobileSettings>
```
### Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the `package.xml` manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see **Settings**. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

#### SEE ALSO:
- **Settings**

### NameSettings

Enables or disables middle name and suffix attributes for the following person objects: Contact, Lead, Person Account, and User.

#### File Suffix and Directory Location

NameSettings values are stored in a single file named `Name.settings` in the `settings` folder. The `.settings` files are different from other named components because there is only one settings file for each settings component.

#### Version

NameSettings components are available in API version 31.0 and later.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableMiddleName</td>
<td>boolean</td>
<td>Indicates whether middle names are enabled (true) or disabled (false) for person objects.</td>
</tr>
<tr>
<td>enableNameSuffix</td>
<td>boolean</td>
<td>Indicates whether suffixes are enabled (true) or disabled (false) for person objects.</td>
</tr>
</tbody>
</table>

---

812
Declarative Metadata Sample Definition

The following is an example of a NameSettings component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<NameSettings xmlns="http://soap.sforce.com/2006/04/metadata">
    <enableMiddleName>true</enableMiddleName>
    <enableNameSuffix>false</enableNameSuffix>
</NameSettings>
```

The following is an example package.xml manifest that references the NameSettings definitions.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>Name</members>
        <name>Settings</name>
    </types>
    <version>31.0</version>
</Package>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

NotificationsSettings

Represents an organization’s mobile settings.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

Declarative Metadata File Suffix and Directory Location

NotificationsSettings values are stored in a single file named Notifications.settings in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

Mobile settings are available in API version 46.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableMobileAppPushNotifications</td>
<td>boolean</td>
<td>Indicates whether mobile push notifications are enabled.</td>
</tr>
<tr>
<td>enableNotifications</td>
<td>boolean</td>
<td>Indicates whether notifications are enabled.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

This is a sample notifications.settings metadata file.

```xml
<NotificationsSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableMobileAppPushNotifications>true</enableMobileAppPushNotifications>
  <enableNotifications>true</enableNotifications>
</NotificationsSettings>
```

Wildcard Support in the Manifest File

The wildcard character `*` (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

OauthCustomScope

Represents a permission defining the protected data that a connected app can access from an external entity when Salesforce is the OAuth authorization provider.

File Suffix and Directory Location

OauthCustomScope components have the suffix `.oauthcustomscope` and are stored in the `oauthcustomscopes` directory.

Version

OAuth custom scopes are available in API version 46.0 and later.

Special Access Rules

You must have the “Manage Connected Apps” permission to access this object.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Required. The description of the permission provided to the connected app by the scope. The custom scope's description must be unique, can only include alphanumeric characters, and can be up to 60 characters long.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>Required. Use when referring to the OAuth custom scope from a program.</td>
</tr>
<tr>
<td>isProtected</td>
<td>boolean</td>
<td>Required. Indicates whether this component is protected () or not (false). Protected components cannot be linked to or referenced by components created in the installing org.</td>
</tr>
</tbody>
</table>
### Field Name | Field Type | Description
--- | --- | ---
isPublic | boolean | Indicates whether the object is included in the connected app’s OpenID Connect discovery endpoint. The default setting is false. For more information, see [OpenID Connect Discovery Endpoint](#).
masterLabel | string | Required. The master label for the custom scope record. This label must be unique and begin with a letter. It can include only alphanumeric characters and underscores. It can’t contain spaces.

### Declarative Metadata Sample Definition
The following is an example of an OAuthCustomScope component. In this example, `basicScope` is the name of custom scope entity being retrieved. You can also enter an asterisk (*) to retrieve all the members of this entity type.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OauthCustomScope xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>Example of a basic custom scope</description>
  <developerName>basicScope</developerName>
  <isProtected>false</isProtected>
  <isPublic>true</isPublic>
  <masterLabel>basicScope</masterLabel>
</OauthCustomScope>
```

The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>basicScope</members>
    <name>OauthCustomScope</name>
  </types>
  <version>46.0</version>
</Package>
```

### Usage
An OAuth custom scope tells an external entity about a connected app’s permissions to access protected data. The OAuth custom scope you create in your Salesforce org corresponds to the same custom scope defined in your external entity and assigned to the resource.

For example, you define an Order Status custom scope in your external entity that allows access to customer order status data in your order system’s API. In Salesforce, you create an OAuth custom scope that you also name Order Status. You assign this custom scope to the connected app requesting access to the order status API. When the external entity receives the connected app’s request to access a customer’s order status, it validates the connected app’s access token and Order Status scope. With a successful validation, the app can access the customer order status information in the order system’s API.

### Wildcard Support in the Manifest File
This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](#).
OmniChannelSettings

Represents the Omni-Channel settings for an org. This type extends the Metadata metadata type and inherits its fullName field. In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

File Suffix and Directory Location

OmniChannelSettings values are stored in the OmniChannel.settings file in the settings folder. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

OmniChannelSettings components are available in API version 44.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableOmniChannel</td>
<td>boolean</td>
<td>Indicates whether Omni-Channel is enabled, giving you access to the objects needed to set up the feature in your org.</td>
</tr>
<tr>
<td>enableOmniSkillsRouting</td>
<td>boolean</td>
<td>Indicates whether skills-based routing is enabled in your org.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a OmniChannelSettings component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OmnichannelSettings xmlns="http://soap.sforce.com/2006/04/metadata">
    <enableOmniChannel>true</enableOmniChannel>
    <enableOmniSkillsRouting>true</enableOmniSkillsRouting>
</OmnichannelSettings>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>OmniChannel</members>
        <name>Settings</name>
    </types>
    <version>44.0</version>
</Package>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
OpportunitySettings

Represents org preferences for features such as automatic opportunity updates and similar-opportunity filters. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

Opportunities values are stored in a single file named Opportunity.settings in the settings directory of the corresponding package directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

OpportunitySettings is available in API version 28.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoActivateNewReminders</td>
<td>boolean</td>
<td>Automatically uses scheduled updates for new opportunities.</td>
</tr>
<tr>
<td>customizableProductSchedulesEnabled</td>
<td>boolean</td>
<td>Lets Salesforce admins customize product schedules by using custom fields, validation rules, and Apex triggers on the LineItemSchedule object. This field is available in API version 46.0 and later.</td>
</tr>
<tr>
<td>Note: If customizable product schedules are enabled, you can use custom fields in default schedules and customize their layout, but any Apex triggers or validation rules that you apply to default schedules are bypassed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>enableUpdateReminders</td>
<td>boolean</td>
<td>Lets users enable automatic, scheduled updates on opportunities.</td>
</tr>
<tr>
<td>enableFindSimilarOpportunities</td>
<td>boolean</td>
<td>Lets users see related or similar existing opportunities.</td>
</tr>
<tr>
<td>enableOpportunityTeam</td>
<td>boolean</td>
<td>Lets users associate team members with opportunities.</td>
</tr>
<tr>
<td>findSimilarOppFilter</td>
<td>FindSimilarOppFilter</td>
<td>Defines parameters for similar opportunities.</td>
</tr>
<tr>
<td>promptToAddProducts</td>
<td>boolean</td>
<td>Prompts users to add related products to an opportunity.</td>
</tr>
</tbody>
</table>

FindSimilarOppFilter

Defines whether to match by entire columns or fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>similarOpportunitiesDisplayColumns</td>
<td>string</td>
<td>The columns to compare.</td>
</tr>
<tr>
<td>similarOpportunitiesMatchFields</td>
<td>string</td>
<td>The fields to compare.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following is an example of the package file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Opportunity</members>
    <name>Settings</name>
  </types>
  <version>28.0</version>
</Package>
```

The package file references the following Opportunity.settings file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OpportunitySettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <autoActivateNewReminders>true</autoActivateNewReminders>
  <enableFindSimilarOpportunities>true</enableFindSimilarOpportunities>
  <findSimilarOppFilter>
    <similarOpportunitiesMatchFields>OPPORTUNITY.Account</similarOpportunitiesMatchFields>
    <similarOpportunitiesMatchFields>OPPORTUNITY.OpportunityCompetitors</similarOpportunitiesMatchFields>
    <similarOpportunitiesMatchFields>CustomField__c</similarOpportunitiesMatchFields>
  </findSimilarOppFilter>
  <enableOpportunityTeam>true</enableOpportunityTeam>
  <enableUpdateReminders>true</enableUpdateReminders>
  <promptToAddProducts>false</promptToAddProducts>
</OpportunitySettings>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

OrderSettings

Represents order settings. This type extends the Metadata metadata type and inherits its fullName field. For more information, see “Set Up Orders” in the Salesforce Help.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

File Suffix and Directory Location

There is one OrderSettings component in a file named Order.settings in the settings folder. The .settings files are different from other named components because there is only one settings file for each settings component.
Version

OrderSettings components are available in API version 30.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableNegativeQuantity</td>
<td>boolean</td>
<td>Indicates whether users in the organization can add order products with quantities of less than zero (true) or not (false).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To enable this preference, enableOrders must be set to true.</td>
</tr>
<tr>
<td>enableOrders</td>
<td>boolean</td>
<td>Indicates whether orders are enabled for the organization (true) or not (false).</td>
</tr>
<tr>
<td>enableReductionOrders</td>
<td>boolean</td>
<td>Indicates whether reduction orders are enabled for the organization (true) or not (false). For more information, see “Reduction Orders” in the Salesforce Help.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To enable this preference, enableOrders must be set to true.</td>
</tr>
<tr>
<td>enableZeroQuantity</td>
<td>boolean</td>
<td>Indicates whether users in the organization can add order products with quantities of zero (true) or not (false). This has a default value of false.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To enable this preference, enableOrders must be set to true.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

This is a sample OrderSettings component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OrderSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableOrders>true</enableOrders>
  <enableReductionOrders>false</enableReductionOrders>
  <enableNegativeQuantity>true</enableNegativeQuantity>
</OrderSettings>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Order</members>
    <name>Settings</name>
  </types>
  <version>30.0</version>
</Package>
```
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

OrgPreferenceSettings

Represents the unique org preference settings in a Salesforce org.

This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

OrgPreferenceSettings values are stored in the OrgPreference.settings file in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

OrgPreferenceSettings components are available in API version 37.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>preferences</td>
<td>OrganizationSettingDetail</td>
<td>The preferences associated with the org settings. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AnalyticsSharingEnable (available in version 40.0 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ApexApprovalLockUnlock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AsyncSaveEnabled (available in version 40.0 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ChatterEnabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CompileOnDeploy (available in version 43.0 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ConsentManagementEnabled (available in version 45.0 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• EnhancedEmailEnabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• EventLogWaveIntegEnabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LoginForensicsEnabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NetworksEnabled (available in version 40.0 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NotesReservedPref01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• OfflineDraftsEnabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PathAssistantsEnabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• S1DesktopEnabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• S1EncryptedStoragePref2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• S1OfflinePref</td>
</tr>
</tbody>
</table>
### OrganizationSettingsDetail

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>settingName</td>
<td>string</td>
<td>The name of the setting. For example, “S1EncryptedStoragePref2.”</td>
</tr>
<tr>
<td>settingValue</td>
<td>boolean</td>
<td>Indicates whether the setting is enabled (true) or not (false).</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of a OrgPreferenceSettings component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OrgPreferenceSettings xmlns="http://soap.sforce.com/2006/04/metadata">
<preferences>
  <settingName>EventLogWaveIntegEnabled</settingName>
  <settingValue>true</settingValue>
</preferences>
<preferences>
  <settingName>SendThroughGmailPref</settingName>
  <settingValue>false</settingValue>
</preferences>
<preferences>
  <settingName>LoginForensicsEnabled</settingName>
  <settingValue>false</settingValue>
</preferences>
<preferences>
  <settingName>EnhancedEmailEnabled</settingName>
  <settingValue>true</settingValue>
</preferences>
<preferences>
  <settingName>NotesReservedPref01</settingName>
  <settingValue>false</settingValue>
</preferences>
<preferences>
  <settingName>S1OfflinePref</settingName>
  <settingValue>true</settingValue>
</preferences>
</OrgPreferenceSettings>
```
<table>
<thead>
<tr>
<th>Setting Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1EncryptedStoragePref2</td>
<td>true</td>
</tr>
<tr>
<td>OfflineDraftsEnabled</td>
<td>false</td>
</tr>
<tr>
<td>ChatterEnabled</td>
<td>true</td>
</tr>
<tr>
<td>SocialProfilesEnable</td>
<td>true</td>
</tr>
<tr>
<td>PathAssistantsEnabled</td>
<td>false</td>
</tr>
<tr>
<td>S1DesktopEnabled</td>
<td>true</td>
</tr>
<tr>
<td>VoiceEnabled</td>
<td>false</td>
</tr>
<tr>
<td>Translation</td>
<td>false</td>
</tr>
<tr>
<td>AnalyticsSharingEnable</td>
<td>true</td>
</tr>
<tr>
<td>NetworksEnabled</td>
<td>true</td>
</tr>
<tr>
<td>AsyncSaveEnabled</td>
<td>false</td>
</tr>
<tr>
<td>ScratchOrgManagementPref</td>
<td>true</td>
</tr>
</tbody>
</table>

</OrgPreferenceSettings>
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

OrgSettings

Represents the settings for org-wide functionality that isn’t associated with any specific feature. This type extends the Metadata metadata type and inherits its fullName field.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

File Suffix and Directory Location

A OrgSettings component file has the suffix .settings and is stored in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

OrgSettings components are available in API version 46.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableCustomerSuccessPortal</td>
<td>boolean</td>
<td>Indicates whether Customer Portal is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableManageSelfServiceUsers</td>
<td>boolean</td>
<td>Indicates whether mass management of self-service users is enabled through the Self-Service Portal (true) or not (false).</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a OrgSettings component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OrgSettings xmlns="http://soap.sforce.com/2006/04/metadata">
    <enableCustomerSuccessPortal>true</enableCustomerSuccessPortal>
    <enableDoNotAutoEnableNewFeatures>false</enableDoNotAutoEnableNewFeatures>
    <enableManageSelfServiceUsers>true</enableManageSelfServiceUsers>
</OrgSettings>
```

Example Package Manifest

The following is an example package manifest used to deploy or retrieve the org settings metadata for an organization:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>Org</members>
    </types>
</Package>
```
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

PathAssistantSettings

Represents the Path preference setting. This type extends the Metadata metadata type and inherits its fullName field.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

File Suffix and Directory Location

PathAssistantSettings components have the suffix .settings and are stored in the settings folder.

Version

PathAssistantSettings components are available in API version 34.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pathAssistantForOpportunityEnabled</td>
<td>boolean</td>
<td>Determines whether the preference is enabled for Path in Opportunity or not. Available in API version 34.0 only.</td>
</tr>
<tr>
<td>pathAssistantEnabled</td>
<td>boolean</td>
<td>Determines whether the preference is enabled for Path or not. Available in API version 35.0 and later.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a PathAssistantSettings component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PathAssistantSettings xmlns="http://soap.sforce.com/2006/04/metadata">
    <pathAssistantEnabled>true</pathAssistantEnabled>
</PathAssistantSettings>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>PathAssistant</members>
    </types>
</Package>
```
Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the `package.xml` manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

ProductSettings

Represents organization preferences for quantity schedules, revenue schedules, and active flag interaction with prices. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

ProductSettings values are stored in a single file named `Product.settings` in the settings directory of the corresponding package directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

ProductSettings is available in API version 28.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableCascadeActivateToRelatedPrices</td>
<td>boolean</td>
<td>When changing active flag on a product record, automatically updates active flag on related prices.</td>
</tr>
<tr>
<td>enableQuantitySchedule</td>
<td>boolean</td>
<td>Enables quantity schedules for products.</td>
</tr>
<tr>
<td>enableRevenueSchedule</td>
<td>boolean</td>
<td>Enables revenue schedules for products.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of the package file.

```
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Product</members>
    <name>Settings</name>
  </types>
  <version>28.0</version>
</Package>
```
The package file references the following `Product.settings` file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ProductSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableCascadeActivateToRelatedPrices>true</enableCascadeActivateToRelatedPrices>
  <enableQuantitySchedule>false</enableQuantitySchedule>
  <enableRevenueSchedule>false</enableRevenueSchedule>
</ProductSettings>
```

### Wildcard Support in the Manifest File

The wildcard character `*` (asterisk) in the `package.xml` manifest file does not apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### ProfileSessionSetting

Represents a profile’s session settings. Use `ProfileSessionSetting` to retrieve the session settings for a given profile. This type extends the Metadata metadata type and inherits its `fullName` field.

### File Suffix and Directory Location

`ProfileSessionSetting` components have the suffix `.profileSessionSetting` and are stored in the `profileSessionSettings` folder.

### Version

`ProfileSessionSetting` components are available in API version 40.0 and later.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>forceLogout</td>
<td>boolean</td>
<td>Required. If <code>true</code>, when the session times out, users are logged out immediately and redirected to a default or custom logout URL.</td>
</tr>
<tr>
<td>profile</td>
<td>string</td>
<td>Required. Name of the user profile.</td>
</tr>
<tr>
<td>requiredSessionLevel</td>
<td><code>SessionSecurityLevel</code></td>
<td>Session security level.</td>
</tr>
<tr>
<td>sessionPersistence</td>
<td>boolean</td>
<td>Beta. If <code>true</code>, keep users logged in to their community until the session times out—even if they close their browser. Use <code>sessionPersistence</code> to reduce how often users must log in to their community. Applies only to the External Identity profile.</td>
</tr>
<tr>
<td>sessionTimeout</td>
<td>int</td>
<td>Required. Specifies how many minutes of inactivity elapse before a user’s authenticated session times out. At the end of the session, the user must log in again. This session timeout value applies to users of the profile and overrides the org-wide timeout value. Changes to the org-wide...</td>
</tr>
</tbody>
</table>
timeout value don't apply to users of this profile. Valid values: 15, 30, 60, 120, 240, 480, 720, 1440.

sessionTimeoutWarning boolean Required. If true, the user receives a warning when the session is about to expire.

SessionSecurityLevel

Session security levels control access to certain types of resources based on the type of authentication used for logging in to the current session. For example, username and password authentication requires the standard session security level. Two-factor authentication requires HIGH_ASSURANCE.

- The HIGH_ASSURANCE security level for this session meets the High Assurance requirements set in the org’s session settings under Session Security Levels.
- The STANDARD security level for this session meets the Standard requirements set in the org’s session settings under Session Security Levels.
- The LOW level is not available or used in the Salesforce UI. It is used at the API level, but users assigned to this level experience unpredictable and reduced functionality.

Declarative Metadata Sample Definition

The following is an example of a ProfileSessionSetting component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ProfileSessionSetting xmlns="http://soap.sforce.com/2006/04/metadata">
  <forceLogout>false</forceLogout>
  <profile>platformportal</profile>
  <requiredSessionLevel>HIGH_ASSURANCE</requiredSessionLevel>
  <sessionTimeout>1440</sessionTimeout>
  <sessionTimeoutWarning>false</sessionTimeoutWarning>
</ProfileSessionSetting>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

QuoteSettings

Enables or disables Quotes, which show proposed prices for products and services. This type extends the Metadata metadata type and inherits its fullName field.
File Suffix and Directory Location

QuoteSettings values are stored in a single file named Quote.settings in the settings directory of the corresponding package directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

QuoteSettings is available in API version 28.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableQuote</td>
<td>boolean</td>
<td>When set to true, users can access Quotes.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of the package file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Quote</members>
    <name>Settings</name>
  </types>
  <version>28.0</version>
</Package>
```

The package file references the following Quote.settings file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<QuoteSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableQuote>true</enableQuote>
</QuoteSettings>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SearchSettings

Represents an org’s search settings.

This type extends the Metadata metadata type and inherits its fullName field.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.
File Suffix and Directory Location

SearchSettings values are stored in a single file named Search.settings in the settings folder. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

SearchSettings is available in API version 37.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>documentContentSearchEnabled</td>
<td>boolean</td>
<td>Indicates if a full-text document search is performed.</td>
</tr>
<tr>
<td>optimizeSearchForCJKEnabled</td>
<td>boolean</td>
<td>Indicates whether the search is optimized for the Japanese, Chinese, and Korean languages. This setting affects sidebar search and the account search for Find Duplicates on a lead record in sidebar search and global search. Enable this option if users are searching mostly in Japanese, Chinese, or Korean, and if the text in searchable fields is mostly in those languages.</td>
</tr>
<tr>
<td>recentlyViewedUsersForBlankLookupEnabled</td>
<td>boolean</td>
<td>Indicates whether the list of records that are returned from a user autocomplete lookup and from a blank user lookup is taken from the user’s recently viewed user records. Otherwise this setting is false if the lookup shows a list of recently accessed user records from across your org (false). Only applies to User object blank lookup searches.</td>
</tr>
<tr>
<td>searchSettingsByObject</td>
<td>SearchSettingsByObject</td>
<td>Represents a list of search settings for each object.</td>
</tr>
<tr>
<td>sidebarAutoCompleteEnabled</td>
<td>boolean</td>
<td>Indicates if autocomplete is enabled for sidebar search. Autocomplete is when users start typing search terms and sidebar search displays a matching list of recently viewed records.</td>
</tr>
<tr>
<td>sidebarDropDownListEnabled</td>
<td>boolean</td>
<td>Indicates if a drop-down list appears in the sidebar search section. From this list, users can select to search within tags, within a specific object, or across all objects.</td>
</tr>
<tr>
<td>sidebarLimitToItemsIOwnCheckboxEnabled</td>
<td>boolean</td>
<td>Indicates if the Limit to Items I Own checkbox appears. The checkbox allows your users to include only records for which they are the record owner when entering search queries in the sidebar.</td>
</tr>
<tr>
<td>singleSearchResultShortcutEnabled</td>
<td>boolean</td>
<td>Indicates if a shortcut is enabled. With the shortcut, users skip the search results page and go directly to the record’s detail page when their search returns only a single item. This setting doesn’t apply to tags, case comments (in advanced search), and global search.</td>
</tr>
<tr>
<td>spellCorrectKnowledgeSearchEnabled</td>
<td>boolean</td>
<td>Indicates if spell check is enabled for Knowledge search.</td>
</tr>
</tbody>
</table>
**SearchSettingsByObject**

Includes `ObjectSearchSetting` field type, which is a list of search settings for each object.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>searchSettingsByObject</td>
<td>ObjectSearchSetting</td>
<td>Contains a list of search settings for each object.</td>
</tr>
</tbody>
</table>

**ObjectSearchSetting**

A list of search settings for each object.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enhancedLookupEnabled</td>
<td>boolean</td>
<td>Indicates if enhanced lookups is enabled for the object.</td>
</tr>
<tr>
<td>lookupAutoCompleteEnabled</td>
<td>boolean</td>
<td>Indicates if autocomplete is enabled for lookup search. Autocomplete is when users edit the lookup field inline by choosing an autosuggestion.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The entity name of the object being configured.</td>
</tr>
<tr>
<td>resultsPerPageCount</td>
<td>int</td>
<td>The number of search results per page.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example of the `Search.settings` file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SearchSettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <documentContentSearchEnabled>true</documentContentSearchEnabled>
  <optimizeSearchForCJKEnabled>true</optimizeSearchForCJKEnabled>
  <recentlyViewedUsersForBlankLookupEnabled>true</recentlyViewedUsersForBlankLookupEnabled>

  <searchSettingsByObject>
    <searchSettingsByObject>
      <enhancedLookupEnabled>false</enhancedLookupEnabled>
      <lookupAutoCompleteEnabled>false</lookupAutoCompleteEnabled>
      <name>Account</name>
      <resultsPerPageCount>25</resultsPerPageCount>
    </searchSettingsByObject>
    <searchSettingsByObject>
      <enhancedLookupEnabled>false</enhancedLookupEnabled>
      <lookupAutoCompleteEnabled>false</lookupAutoCompleteEnabled>
      <name>Activity</name>
      <resultsPerPageCount>25</resultsPerPageCount>
    </searchSettingsByObject>
    <searchSettingsByObject>
      <enhancedLookupEnabled>false</enhancedLookupEnabled>
      <lookupAutoCompleteEnabled>false</lookupAutoCompleteEnabled>
      <name>Asset</name>
      <resultsPerPageCount>25</resultsPerPageCount>
    </searchSettingsByObject>
  </searchSettingsByObject>
</SearchSettings>
```
Example Package Manifest

The following is an example package manifest used to deploy or retrieve the Account settings metadata for an organization.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Search</members>
    <name>Settings</name>
  </types>
  <version>37.0</version>
</Package>
```

Wildcard Support in the Manifest File

The wildcard character `*` (asterisk) in the `package.xml` manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SecuritySettings

Represents an organization’s security settings. Security settings define trusted IP ranges for network access, password and login requirements, and session expiration and security settings.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

Declarative Metadata File Suffix and Directory Location

SecuritySettings values are stored in a single file named `Security.settings` in the `settings` directory. The `.settings` files are different from other named components because there is only one settings file for each settings component.

**Note:** SecuritySettings is no longer available in API versions 25.0 and 26.0.

Version

Security settings are available in API version 27.0 and later.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>networkAccess</td>
<td>NetworkAccess</td>
<td>The trusted IP address ranges from which users can always log in without requiring computer activation.</td>
</tr>
<tr>
<td>passwordPolicies</td>
<td>PasswordPolicies</td>
<td>The requirements for passwords and logins, and assistance with retrieving forgotten passwords.</td>
</tr>
<tr>
<td>sessionSettings</td>
<td>SessionSettings</td>
<td>The settings for session expiration and security.</td>
</tr>
</tbody>
</table>

NetworkAccess

Represents your organization’s trusted IP address ranges for network access.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipRanges</td>
<td>IpRange[]</td>
<td>The trusted IP address ranges from which users can always log in without requiring computer activation.</td>
</tr>
</tbody>
</table>

Note: To add an IP range, deploy all existing IP ranges, including the one you want to add. Otherwise, the existing IP ranges are replaced with the ones you deploy. To remove all the IP ranges in an organization, leave the networkAccess field blank (<networkAccess></networkAccess>).

IpRange

Defines a range of trusted IP addresses for network access.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the trusted IP range. Use this field to identify the range, such as which corporate network corresponds to this range. This field is available in API version 34.0 and later.</td>
</tr>
<tr>
<td>end</td>
<td>string</td>
<td>The IP address that defines the high end of a range of trusted addresses.</td>
</tr>
<tr>
<td>start</td>
<td>string</td>
<td>The IP address that defines the low end of a range of trusted addresses.</td>
</tr>
</tbody>
</table>

PasswordPolicies

Represents your organization’s password and login policies.
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiOnlyUserHomePageURL</td>
<td>string</td>
<td>The URL to which users with the “API Only User” permission are redirected instead of the login page.</td>
</tr>
<tr>
<td>complexity</td>
<td>Complexity (enumeration of type string)</td>
<td>Required. The types of characters that must be used in a user’s password. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NoRestriction—Has no requirements and is the least secure option.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- AlphaNumeric—The default setting. Requires at least one alphabetic character and one number. This value is the default value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SpecialCharacters—Requires at least one alphabetic character, one number, and one of the following characters: ! &quot; # $ % &amp; ' ( ) * + , - ./ ; : &lt; = &gt; ? @ [ \ ] ^ _ ` {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-UpperCaseLowerCaseNumeric—Requires at least one number, one uppercase letter, and one lowercase letter. This value is available in API version 31.0 and later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-UpperCaseLowerCaseNumericSpecialCharacters—Requires at least one number, one uppercase letter, one lowercase letter, and one special character (! &quot; # $ % &amp; ' ( ) * + , - ./ ; : &lt; = &gt; ? @ [ \ ] ^ _ ` {</td>
</tr>
<tr>
<td>expiration</td>
<td>Expiration (enumeration of type string)</td>
<td>Required. The length of time until a user password expires and must be changed. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Never</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ThirtyDays</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SixtyDays</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NinetyDays. This value is the default value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SixMonths</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- OneYear</td>
</tr>
<tr>
<td>minimumPasswordLifetime</td>
<td>boolean</td>
<td>Indicates whether a one-day minimum password lifetime is required (true) or not (false). This field is available in API version 31.0 and later.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>historyRestriction</td>
<td>string</td>
<td>Required. The number of previous passwords saved for users so that they must always reset a new, unique password. Valid values are 0 through 24 passwords remembered. The maximum value of 24 applies to API version 31.0 and later. In earlier versions, the maximum value is 16. The default value is 3.</td>
</tr>
<tr>
<td>lockoutInterval</td>
<td>LockoutInterval (enumeration of type string)</td>
<td>Required. The duration of the login lockout. Valid values are: FifteenMinutes. This value is the default value. ThirtyMinutes SixtyMinutes Forever (must be reset by admin)</td>
</tr>
<tr>
<td>maxLoginAttempts</td>
<td>MaxLoginAttempts (enumeration of type string)</td>
<td>Required. The number of login failures allowed for a user before the user is locked out. Valid values are: NoLimit ThreeAttempts FiveAttempts TenAttempts. This value is the default value.</td>
</tr>
<tr>
<td>minimumPasswordLength</td>
<td>string</td>
<td>Required. The minimum number of characters required for a password. Valid values are from 5 to 50. The default value is 8. This field is available in API version 35.0 and later. Before API version 35.0, specify minimum password length with the enumeration minPasswordLength, with valid values FiveCharacters, EightCharacters (default), TenCharacters, TwelveCharacters (API version 31.0 and later), and FifteenCharacters (API version 34.0 and later).</td>
</tr>
<tr>
<td>obscureSecretAnswer</td>
<td>boolean</td>
<td>Hides the secret answer associated with a password (true) or not (false). Note: If your org uses the Microsoft Input Method Editor (IME) with the input mode set to Hiragana, when you type ASCII characters, they're converted to Japanese characters in normal text fields. However, the IME doesn't work properly in fields with obscured text. If your org's users cannot properly enter their passwords or other values after enabling this feature, disable the feature.</td>
</tr>
<tr>
<td>passwordAssistanceMessage</td>
<td>string</td>
<td>The text that appears in the Account Lockout email and at the bottom of the Confirm Identity screen for users resetting their passwords.</td>
</tr>
<tr>
<td>passwordAssistanceURL</td>
<td>string</td>
<td>The URL that users can click to retrieve forgotten passwords.</td>
</tr>
</tbody>
</table>
### SecuritySettings

Represents your organization’s session expiration and security settings.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| questionRestriction           | QuestionRestriction (enumeration of type string) | Required. The restriction on whether the answer to the password hint question can contain the password itself. Valid values are:  
- None  
- DoesNotContainPassword. This value is the default value. |

#### SessionSettings

Indicates whether the session timeout warning popup is disabled (true) or enabled (false).

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>disableTimeoutWarning</td>
<td>boolean</td>
<td>Indicates whether the session timeout warning popup is disabled (true) or enabled (false).</td>
</tr>
<tr>
<td>enableCSPOnEmail</td>
<td>boolean</td>
<td>Indicates whether a content security policy is enabled for the email template. A content security policy helps prevent cross-site scripting attacks by whitelisting sources of images and other content.</td>
</tr>
<tr>
<td>enableCSRFOnGet</td>
<td>boolean</td>
<td>Indicates whether Cross-Site Request Forgery (CSRF) protection on GET requests on non-setup pages is enabled (true) or disabled (false).</td>
</tr>
<tr>
<td>enableCSRFOnPost</td>
<td>boolean</td>
<td>Indicates whether Cross-Site Request Forgery (CSRF) protection on POST requests on non-setup pages is enabled (true) or disabled (false).</td>
</tr>
<tr>
<td>enableCacheAndAutocomplete</td>
<td>boolean</td>
<td>Indicates whether the user’s browser is allowed to store usernames and auto-fill the User Name field on the login page (true) or not (false).</td>
</tr>
<tr>
<td>enableClickjackNonsetupSFDC</td>
<td>boolean</td>
<td>Indicates whether clickjack protection for non-setup Salesforce pages is enabled (true) or disabled (false).</td>
</tr>
<tr>
<td>enableClickjackNonsetupUser</td>
<td>boolean</td>
<td>Indicates whether clickjack protection for customer Visualforce pages with standard headers turned on is enabled (true) or disabled (false).</td>
</tr>
<tr>
<td>enableClickjackNonsetupUserHeaderless</td>
<td>boolean</td>
<td>Indicates whether clickjack protection for customer Visualforce pages with standard headers turned off is enabled (true) or disabled (false). Available in API version 34.0 and later.</td>
</tr>
<tr>
<td>enableClickjackSetup</td>
<td>boolean</td>
<td>Indicates whether clickjack protection for setup pages is enabled (true) or disabled (false).</td>
</tr>
<tr>
<td>enableContentSniffingProtection</td>
<td>boolean</td>
<td>Indicates if the browser is prevented from inferring the MIME type from the document content and from executing malicious files (JavaScript, Stylesheet) as dynamic content.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>enablePostForSessions</td>
<td>boolean</td>
<td>Indicates whether cross-domain session information is exchanged using a POST request instead of a GET request, such as when a user is using a Visualforce page. In this context, POST requests are more secure than GET requests. Available in API version 31.0 and later.</td>
</tr>
<tr>
<td>enableSMSIdentity</td>
<td>boolean</td>
<td>Indicates whether users can receive a one-time PIN delivered via SMS (true) or not (false).</td>
</tr>
<tr>
<td>enableUpgradeInsecureRequests</td>
<td>boolean</td>
<td>Indicates if HTTPS is required for connecting to third-party domains. Note: This setting is enabled by default on accounts created after the Summer '17 release.</td>
</tr>
<tr>
<td>enableXssProtection</td>
<td>boolean</td>
<td>Indicates if protection against reflected cross-site scripting attacks is enabled. If a reflected cross-site scripting attack is detected and XSS protection is enabled, the browser shows a blank page with no content. This field is available in API version 39.0 and later.</td>
</tr>
<tr>
<td>enforceIpRangesEveryRequest</td>
<td>boolean</td>
<td>If true, the IP addresses in Login IP Ranges are enforced when a user accesses Salesforce (on every page request), including access from a client application. If false, the IP addresses in Login IP Ranges are enforced only when a user logs in. This field affects all user profiles that have login IP restrictions. Available in API version 34.0 and later.</td>
</tr>
<tr>
<td>forceLogoutOnSessionTimeout</td>
<td>boolean</td>
<td>Indicates that when sessions time out for inactive users, current sessions become invalid. The browser refreshes and returns to the login page. To access the org, the user must log in again. Enabled (true) or not (false). Available in API version 31.0 and later.</td>
</tr>
<tr>
<td>forceRelogin</td>
<td>boolean</td>
<td>If true, an administrator who is logged in as another user must log in again to their original session, after logging out as the secondary user. If false, the administrator is not required to log in again.</td>
</tr>
<tr>
<td>hstsOnForcecomSites</td>
<td>boolean</td>
<td>Indicates whether Visualforce, Salesforce sites, or Communities must use HTTPS. This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>identityConfirmationOnEmailChange</td>
<td>boolean</td>
<td>Indicates if a user's identity is confirmed when changing their email address, instead of requiring a relogin.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>identityConfirmationOnTwoFactorRegistrationEnabled</td>
<td>boolean</td>
<td>This field is available in API version 42.0 and later. Indicates if users are required to confirm their identities to add a two-factor authentication method, such as Salesforce Authenticator, instead of requiring a relogin. This field is available in API version 40.0 and later.</td>
</tr>
<tr>
<td>lockSessionsToDomain</td>
<td>boolean</td>
<td>Indicates whether the current UI session for a user, such as a community user, is associated with a specific domain. This check helps prevent unauthorized use of the session ID in another domain. The value is <code>true</code> by default for organizations created with the Spring ’15 release or later. Available in API version 33.0 and later.</td>
</tr>
<tr>
<td>lockSessionsToIp</td>
<td>boolean</td>
<td>Indicates whether user sessions are locked to the IP address from which the user logged in (<code>true</code>) or not (<code>false</code>).</td>
</tr>
<tr>
<td>logoutURL</td>
<td>string</td>
<td>The URL to which users are redirected when they log out of Salesforce. If no value is specified, the default is <code>https://login.salesforce.com</code> unless MyDomain is enabled. If MyDomain is enabled, the default is <code>https://customdomain.my.salesforce.com</code>. Available in API version 34.0 and later.</td>
</tr>
<tr>
<td>redirectionWarning</td>
<td>boolean</td>
<td>Indicates whether users see an alert when they click a link in a web tab that redirects them outside the salesforce.com domain. Available in API version 42.0 and later.</td>
</tr>
<tr>
<td>referrerPolicy</td>
<td>boolean</td>
<td>Indicates whether the referrer header hides sensitive information that could be present in the full URL. If <code>true</code>, then the referrer header displays only <code>salesforce.com</code>. If <code>false</code>, then the header displays the entire URL. For a Visualforce user, if <code>referrerPolicy</code> is set to <code>true</code>, then the referrer header displays only <code>force.com</code>. If <code>false</code>, then the header displays the entire URL. Available in API version 42.0 and later.</td>
</tr>
<tr>
<td>requireHttpOnly</td>
<td>boolean</td>
<td>Sets the <code>HttpOnly</code> attribute on session cookies, making them inaccessible via JavaScript. If <code>true</code>, session ID cookie access is restricted. If <code>false</code>, access is not restricted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> When a request is made from a domain within salesforce.com or force.com, the referrer header displays the entire URL.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> If you have a custom or packaged application that uses JavaScript to access session ID cookies, your application breaks if <code>requireHttpOnly</code> is set to <code>true</code>. The application can't access the cookie. This field is available in API version 40.0 and later.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>requireHttps</td>
<td>boolean</td>
<td>Determines whether HTTPS is required to log in to or access Salesforce. This option is enabled by default for security reasons. To change to HTTP, contact Salesforce Customer Support. This field is available in API version 40.0 and later.</td>
</tr>
</tbody>
</table>
| sessionTimeout| SessionTimeout (enumeration of type string) | The length of time after which users without activity are prompted to log out or continue working. Valid values are:  
- FifteenMinutes  
- ThirtyMinutes  
- SixtyMinutes  
- TwoHours  
- FourHours  
- EightHours  
- TwelveHours |

**Declarative Metadata Sample Definition**

The following is a sample `security.settings` metadata file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SecuritySettings xmlns="http://soap.sforce.com/2006/04/metadata">
  <networkAccess>
    <ipRanges>
      <end>127.0.0.1<end>
      <start>127.0.0.1<start>
    </ipRanges>
  </networkAccess>
  <passwordPolicies>
    <apiOnlyUserHomePageURL>http://www.altPage.com</apiOnlyUserHomePageURL>
    <complexity>SpecialCharacters</complexity>
    <expiration>OneYear</expiration>
    <passwordAssistanceURL>http://www.acme.com/forgotpassword</passwordAssistanceURL>
    <passwordAssistanceMessage>Forgot your password? Reset it here.</passwordAssistanceMessage>
    <historyRestriction>3</historyRestriction>
    <lockoutInterval>ThirtyMinutes</lockoutInterval>
    <maxLoginAttempts>ThreeAttempts</maxLoginAttempts>
    <minimumPasswordLength>10</minimumPasswordLength>
    <questionRestriction>None</questionRestriction>
  </passwordPolicies>
  <sessionSettings>
    <disableTimeoutWarning>true</disableTimeoutWarning>
    <enableCSRFOnGet>false</enableCSRFOnGet>
    <enableCSRFOnPost>false</enableCSRFOnPost>
    <enableCacheAndAutocomplete>false</enableCacheAndAutocomplete>
</SecuritySettings>
```
Metadata Types

SocialCustomerServiceSettings

Wildcard Support in the Manifest File
The wildcard character * (asterisk) in the package.xml manifest file doesn't apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:

Settings

SocialCustomerServiceSettings
Represents which Social Business Rules option has been saved to format inbound content from social posts to cases.

File Suffix and Directory Location
SocialCustomerServiceSettings components have the suffix settings and are stored in the settings folder.

Version
SocialCustomerServiceSettings is available in API version 41.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>caseSubjectOption</td>
<td>CaseSubjectOption</td>
<td>Required. Specifies an option from which inbound social content is formatted to appear in case records. CaseSubject field. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• SocialPostSource</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SocialPostContent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BuildCustom</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

This is a sample of a SocialCustomerServiceSettings.settings file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>Settings</name>
  </types>
  <version>41.0</version>
</Package>
```

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn't apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Territory2Settings

Represents the metadata for the default settings for Territory Management 2.0 users to access and modify records associated with sales territories. The standard record access settings apply to accounts and opportunities. If your Salesforce org uses Private default internal access for contacts or cases, you can also set access for those records. This type extends the Metadata metadata type and inherits its fullName field. Available only if Territory Management 2.0 has been enabled for your org.

File Suffix and Directory Location

Territory2Settings components have the suffix settings and are stored in the Settings folder.

Version

Territory2Settings components are available in API version 32.0 and later.

Special Access Rules

The Territory2Model object has a State field in the SOAP API. States include Planning, Active, Archived, and a number of other states, such as Cloning, that indicate that a process is underway. Users who do not have the "Manage Territories" permission can access only territories that belong to the model in Active state. The "Manage Territories" permission is required for deploy() calls for all territory management entities, in addition to the "Modify All Data" permission required by Metadata API. Using retrieve() without the "Manage Territories" permission will return only entities that belong to a Territory2Model in Active state. We recommend against retrieving without the "Manage Territories" permission because the call will retrieve only partial data.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultAccountAccessLevel</td>
<td>string</td>
<td>The default level of access users will have to account records in territories: <code>view</code> and <code>edit</code> accounts assigned to territories or <code>view</code>, <code>edit</code>, <code>transfer</code>, and <code>delete</code> accounts assigned to territories.</td>
</tr>
<tr>
<td>defaultCaseAccessLevel</td>
<td>string</td>
<td>The default level of access users will have to case records in territories: <code>view</code> and <code>edit</code> accounts assigned to territories or <code>view</code>, <code>edit</code>, <code>transfer</code>, and <code>delete</code> accounts assigned to territories.</td>
</tr>
<tr>
<td>defaultContactAccessLevel</td>
<td>string</td>
<td>The default level of access users will have to contact records in territories: <code>view</code> and <code>edit</code> accounts assigned to territories or <code>view</code>, <code>edit</code>, <code>transfer</code>, and <code>delete</code> accounts assigned to territories.</td>
</tr>
<tr>
<td>defaultOpportunityAccessLevel</td>
<td>string</td>
<td>The default level of access users will have to opportunity records in territories: <code>view</code> and <code>edit</code> accounts assigned to territories or <code>view</code>, <code>edit</code>, <code>transfer</code>, and <code>delete</code> accounts assigned to territories.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following example shows the definition of a Territory2Settings component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Territory2Settings xmlns="http://soap.sforce.com/2006/04/metadata">
  <defaultAccountAccessLevel>Owner</defaultAccountAccessLevel>
  <defaultOpportunityAccessLevel>Read</defaultOpportunityAccessLevel>
  <defaultCaseAccessLevel>Read</defaultCaseAccessLevel>
  <defaultCaseAccessLevel>None</defaultCaseAccessLevel>
  <defaultContactAccessLevel>Edit</defaultContactAccessLevel>
</Territory2Settings>
```

Usage

Territory Management 2.0 components don’t support packaging or change sets and aren’t supported in CRUD calls.

Wildcard Support in the Manifest File

The wildcard character `*` (asterisk) in the `package.xml` manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

UserManagementSettings

Represents a selection of user management options.

This type extends the Metadata metadata type and inherits its `fullName` field.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.
Declarative Metadata File Suffix and Directory Location

User Management Settings are stored in the UserManagement.settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

Version

User Management Settings are available in API version 46.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableEnhancedPermsetMgmt</td>
<td>boolean</td>
<td>Indicates whether Enhanced Permission Set Component Views is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableEnhancedProfileMgmt</td>
<td>boolean</td>
<td>Indicates whether Enhanced Profile Lists Views is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableNewProfileUI</td>
<td>boolean</td>
<td>Indicates whether Enhanced Profile User Interface is enabled (true) or not (false).</td>
</tr>
<tr>
<td>enableUserSelfDeactivate</td>
<td>boolean</td>
<td>Indicates whether Self Deactivation is enabled (true) or not (false).</td>
</tr>
</tbody>
</table>

Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn't apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SharedTo

SharedTo defines the sharing access for a list view or a folder. It can be used to specify the target and source for owner-based sharing rules. See “Sharing Considerations” and “What Is a Group?” in Salesforce Help.

Note: SharedTo is included in the metadata for shared and private list views. SharedTo isn't in the metadata for public list views.

Declarative Metadata File Suffix and Directory Location

SharedTo is used with ListView, Folder, and SharingRules.

Version

SharedTo is available in API version 17.0 and later.
### Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allCustomerPortalUsers</td>
<td>string</td>
<td>A group containing all customer portal users. This field is available in API version 24.0 and later.</td>
</tr>
<tr>
<td>allInternalUsers</td>
<td>string</td>
<td>A group containing all internal and nonportal users. This field is available in API version 24.0 and later.</td>
</tr>
<tr>
<td>allPartnerUsers</td>
<td>string</td>
<td>A group containing all partner users. This field is available in API version 24.0 and later.</td>
</tr>
<tr>
<td>channelProgramGroup</td>
<td>string</td>
<td>A system-managed group with sharing access containing all partner members of the corresponding channel program or level. This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>channelProgramGroups</td>
<td>string[]</td>
<td>A list of system-managed groups with sharing access containing all partner members of the corresponding channel programs or levels. This field is available in API version 41.0 and later.</td>
</tr>
<tr>
<td>group</td>
<td>string[]</td>
<td>A list of groups with sharing access. Use this field instead of the groups field. This field is available in API version 22.0 and later.</td>
</tr>
<tr>
<td>groups</td>
<td>string[]</td>
<td>A list of groups with sharing access. Use the group field instead for API version 22.0 and later.</td>
</tr>
<tr>
<td>managerSubordinates</td>
<td>string[]</td>
<td>A list of users whose direct and indirect subordinates receive sharing access. This field is available in API version 24.0 and later.</td>
</tr>
<tr>
<td>managers</td>
<td>string[]</td>
<td>A list of users whose direct and indirect managers receive sharing access. This field is available in API version 24.0 and later.</td>
</tr>
<tr>
<td>portalRole</td>
<td>string[]</td>
<td>A list of groups with sharing access containing all users in a portal role. This field is available in API version 24.0 and later.</td>
</tr>
<tr>
<td>portalRoleandSubordinates</td>
<td>string[]</td>
<td>A list of groups with sharing access containing all users in a portal role or those under that role. This field is available in API version 24.0 and later.</td>
</tr>
<tr>
<td>role</td>
<td>string[]</td>
<td>A list of roles with sharing access. Use this field instead of the roles field. This field is available in API version 22.0 and later.</td>
</tr>
<tr>
<td>Field</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>roleAndSubordinates</td>
<td>string[]</td>
<td>A list of roles with sharing access. All roles below each of these roles in the role hierarchy also have sharing access. If portal accounts are enabled, then all roles and portal accounts below each of these roles in the role hierarchy also have sharing access. Use this field instead of the <code>rolesAndSubordinates</code> field. Today, this field is available in API version 22.0 and later.</td>
</tr>
<tr>
<td>roleAndSubordinatesInternal</td>
<td>string[]</td>
<td>A list of roles with sharing access. All roles below each of these roles in the role hierarchy also have sharing access.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 22.0 and later.</td>
</tr>
<tr>
<td>roles</td>
<td>string[]</td>
<td>A list of roles with sharing access. Use the <code>role</code> field instead for API version 22.0 and later.</td>
</tr>
<tr>
<td>rolesAndSubordinates</td>
<td>string[]</td>
<td>A list of roles with sharing access. All roles below each of these roles in the role hierarchy also have sharing access. If portal accounts are enabled, then all roles and portal accounts below each of these roles in the role hierarchy also have sharing access. Use the <code>roleAndSubordinates</code> field instead for API version 22.0 and later.</td>
</tr>
<tr>
<td>territories</td>
<td>string[]</td>
<td>A list of territories with sharing access. Use the <code>territory</code> field instead for API version 22.0 and later.</td>
</tr>
<tr>
<td>territoriesAndSubordinates</td>
<td>string[]</td>
<td>A list of territories with sharing access. All territories below each of these territories in the territory hierarchy also have sharing access. Use the <code>territoryAndSubordinates</code> field instead for API version 22.0 and later.</td>
</tr>
<tr>
<td>territory</td>
<td>string[]</td>
<td>A list of territories with sharing access. Use this field instead of the <code>territories</code> field. If you're using Enterprise Territory Management, use <code>modelName.territoryName</code> for the shared-to and shared-from territory values, where:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>modelName</code> equals the name of the active territory model in the API.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>territoryName</code> equals the territory's name in the API.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 22.0 and later.</td>
</tr>
<tr>
<td>territoryAndSubordinates</td>
<td>string[]</td>
<td>A list of territories with sharing access. All territories below each of these territories in the territory hierarchy also have sharing access.</td>
</tr>
</tbody>
</table>
### SharingBaseRule

Represents sharing rule settings such as access level and to whom access is granted.

This type extends the Metadata metadata type and inherits its `fullName` field.

**Note:** You can't create a SharingBaseRule component directly. Use the components under `SharingRules` instead.

#### Version

SharingBaseRule replaces BaseSharingRule and is available in API version 33.0 and later.

#### Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessLevel</td>
<td>string</td>
<td>Required. The access level that the sharing rule grants.</td>
</tr>
<tr>
<td>accountSettings</td>
<td>AccountSharingRuleSettings[]</td>
<td>The access level for the account's children (case, contact, and opportunity).</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Describes the sharing rule. Maximum of 1000 characters.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Label for the sharing rule.</td>
</tr>
<tr>
<td>sharedTo</td>
<td>SharedTo</td>
<td>Required. Specifies who the record should be shared with.</td>
</tr>
</tbody>
</table>
AccountSharingRuleSettings

Defines the access level for the case, contact, and opportunity associated with the account.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>caseAccessLevel</td>
<td>string</td>
<td>Required. The access level that the user or group has to cases associated with the account. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td>contactAccessLevel</td>
<td>string</td>
<td>Required. The access level that the user or group has to contacts associated with the account. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td>opportunityAccessLevel</td>
<td>string</td>
<td>Required. The access level that the user or group has to opportunities associated with the account. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
</tbody>
</table>

Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SharingRules

Represents the base container for sharing rules, which can be criteria-based, ownership-based, or territory-based. SharingRules enables you to share records with a set of users, using rules that specify the access level for the target user group.

This type extends the Metadata metadata type and inherits its fullName field. For more information, see “Sharing Rules” in the Salesforce online help.

In API version 33.0 and later, retrieving, deleting, or deploying of all sharing rules in an organization is available. Wildcard support is also available. You can’t retrieve, delete, or deploy manual sharing rules or sharing rules by their type (owner, criteria-based, or territory).
Declarative Metadata File Suffix and Directory Location

In API version 33.0 and later, components are stored in the sharingRules folder and their file name matches the object name with the suffix `.sharingRules`. Criteria-based, owner-based, and territory-based sharing rules are all contained in a `object.sharingRule` file.

Prior to API version 33.0, SharingRules components are stored in their corresponding object directory and the file name matches the object name. For example, the `accountSharingRules` directory contains an `Account.sharingRules` file for account sharing rules. SharingRules for custom objects are stored in the `customObjectSharingRules` directory, which contains files with the `.sharingRules` extension such as `ObjA__c.sharingRules`, where `ObjA` refers to the developer name of a custom object type.

Version

SharingRules components are available in API version 24.0 and later, but these components are no longer available in API version 33.0 and later: AccountSharingRules, CampaignSharingRules, CaseSharingRules, ContactSharingRules, LeadSharingRules, OpportunitySharingRules, AccountTerritorySharingRules, CustomObjectSharingRules, UserSharingRules.

In API version 33.0 and later, use `SharingCriteriaRule`, `SharingOwnerRule` and `SharingTerritoryRule`.

Fields

The following information assumes that you are familiar with implementing sharing rules for standard objects and custom objects. For more information on these fields, see “Sharing Settings” in the Salesforce online help.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sharingCriteriaRules</td>
<td>SharingCriteriaRule[]</td>
<td>An array of criteria-based sharing rules. Available in API version 33.0 and later.</td>
</tr>
<tr>
<td>sharingOwnerRules</td>
<td>SharingOwnerRule[]</td>
<td>An array of ownership-based sharing rules. Available in API version 33.0 and later.</td>
</tr>
<tr>
<td>sharingTerritoryRules</td>
<td>SharingTerritoryRule[]</td>
<td>An array of territory-based sharing rules. Available in API version 33.0 and later.</td>
</tr>
</tbody>
</table>

SharingCriteriaRule

Defines a criteria-based sharing rule. It extends `SharingBaseRule` and inherits all its fields. Available in API version 33.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Advanced filter conditions that are specified for the sharing rule.</td>
</tr>
<tr>
<td>criteriaItems</td>
<td>FilterItem[]</td>
<td>An array of the boolean criteria (conditions) for the sharing rule.</td>
</tr>
</tbody>
</table>
SharingOwnerRule

Defines a ownership-based sharing rule. It extends SharingBaseRule and inherits all its fields. Available in API version 33.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| sharedFrom  | SharedTo    | Required. Specifies the record owners. If you’re using Enterprise Territory Management, use `modelName.territoryName` for the shared-to and shared-from territory and `territoryAndSubordinates` values on the `SharedTo` type, where:

- `modelName` equals the name of the active territory model in the API.
- `territoryName` equals the territory’s name in the API. |

SharingTerritoryRule

Defines a territory-based sharing rule. It extends SharingOwnerRule and inherits all its fields. Available in API version 33.0 and later.

AccountSharingRules

Represents the sharing rules for accounts. It extends the SharingRules metadata type and inherits its `fullName` field. Only available in API version 32.0 and earlier.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteriaBasedRules</td>
<td>AccountCriteriaBasedSharingRule[]</td>
<td>List that defines user criteria-based rules.</td>
</tr>
<tr>
<td>ownerRules</td>
<td>AccountOwnerSharingRule[]</td>
<td>List that defines user membership-based rules.</td>
</tr>
</tbody>
</table>

CampaignSharingRules

Represents the sharing rules for campaigns. It extends the SharingRules metadata type and inherits its `fullName` field. Only available in API version 32.0 and earlier.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteriaBasedRules</td>
<td>CampaignCriteriaBasedSharingRule[]</td>
<td>List that defines user criteria-based rules.</td>
</tr>
<tr>
<td>ownerRules</td>
<td>CampaignOwnerSharingRule[]</td>
<td>List that defines user membership-based rules.</td>
</tr>
</tbody>
</table>

CaseSharingRules

Represents the sharing rules for cases. It extends the SharingRules metadata type and inherits its `fullName` field. Only available in API version 32.0 and earlier.
### ContactSharingRules

Represents the sharing rules for contacts. It extends the `SharingRules` metadata type and inherits its `fullName` field. Only available in API version 32.0 and earlier.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteriaBasedRules</td>
<td>ContactCriteriaBasedSharingRule[]</td>
<td>List that defines user criteria-based rules.</td>
</tr>
<tr>
<td>ownerRules</td>
<td>ContactOwnerSharingRule[]</td>
<td>List that defines user membership-based rules.</td>
</tr>
</tbody>
</table>

### LeadSharingRules

Represents the sharing rules for leads. It extends the `SharingRules` metadata type and inherits its `fullName` field. Only available in API version 32.0 and earlier.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteriaBasedRules</td>
<td>LeadCriteriaBasedSharingRule[]</td>
<td>List that defines user criteria-based rules.</td>
</tr>
<tr>
<td>ownerRules</td>
<td>LeadOwnerSharingRule[]</td>
<td>List that defines user membership-based rules.</td>
</tr>
</tbody>
</table>

### OpportunitySharingRules

Represents the sharing rules for opportunities. It extends the `SharingRules` metadata type and inherits its `fullName` field. Only available in API version 32.0 and earlier.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>criteriaBasedRules</td>
<td>OpportunityCriteriaBasedSharingRule[]</td>
<td>List that defines user criteria-based rules.</td>
</tr>
<tr>
<td>ownerRules</td>
<td>OpportunityOwnerSharingRule[]</td>
<td>List that defines user membership-based rules.</td>
</tr>
</tbody>
</table>

### AccountTerritorySharingRules

Represents the sharing rules for account territories in the original territory management feature. It extends the `SharingRules` metadata type and inherits its `fullName` field. Only available in API version 32.0 and earlier.
Metadata Types

**Field** | **Field Type** | **Description**
--- | --- | ---
rules | AccountTerritorySharingRule[] | List that defines user membership-based rules. The list of acceptable values for the `sharedFrom` fields are:
- `territory`
- `territoryAndSubordinates`

### CustomObjectSharingRules

Represents the sharing rules for custom objects. It extends the `SharingRules` metadata type and inherits its `fullName` field. Only available in API version 32.0 and earlier.

**Field** | **Field Type** | **Description**
--- | --- | ---
criteriaBasedRules | CustomObjectCriteriaBasedSharingRule[] | List that defines user criteria-based rules.
ownerRules | CustomObjectOwnerSharingRule[] | List that defines user membership-based rules.

### UserSharingRules

Represents the sharing rules for users. With user sharing rules, you can share members of a group with members of another group. It extends the `SharingRules` metadata type and inherits its `fullName` field. Only available in API version 32.0 and earlier.

**Field** | **Field Type** | **Description**
criteriaBasedRules | UserCriteriaBasedSharingRule[] | List that defines user criteria-based rules.
membershipRules | UserMembershipSharingRule[] | List that defines user membership-based rules.

### Declarative Metadata Sample Definition

For retrieving sharing rules, see `package.xml` sample at [Sharing Rules](#).

The following sample XML definition represents a criteria-based sharing rule in API version 33.0.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SharingRules xmlns="http://soap.sforce.com/2006/04/metadata">
  <sharingCriteriaRules>
    <fullName>AccountCriteriaShareWithCEO</fullName>
    <accessLevel>Edit</accessLevel>
    <accountSettings>
      <caseAccessLevel>Read</caseAccessLevel>
      <contactAccessLevel>Edit</contactAccessLevel>
      <opportunityAccessLevel>Edit</opportunityAccessLevel>
    </accountSettings>
    <criteriaItems>
      <field>Name</field>
      <operation>startsWith</operation>
      <value>Test</value>
    </criteriaItems>
  </sharingCriteriaRules>
</SharingRules>
```
The following sample XML definition represents an ownership-based sharing rule in API version 33.0.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SharingRules xmlns="http://soap.sforce.com/2006/04/metadata">
  <sharingOwnerRules>
    <fullName>MyCase</fullName>
    <accessLevel>Edit</accessLevel>
    <description>my case test owner sharing rule desc</description>
    <label>MyCase</label>
    <sharedFrom>
      <role>COO</role>
    </sharedFrom>
    <sharedTo>
      <role>CEO</role>
    </sharedTo>
  </sharingOwnerRules>
</SharingRules>
```

The following sample XML definition represents a territory-based sharing rule in API version 33.0.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SharingRules xmlns="http://soap.sforce.com/2006/04/metadata">
  <sharingTerritoryRules>
    <fullName>MyAccountTerritoryRule</fullName>
    <accessLevel>Read</accessLevel>
    <accountSettings>
      <caseAccessLevel>None</caseAccessLevel>
      <contactAccessLevel>Read</contactAccessLevel>
      <opportunityAccessLevel>None</opportunityAccessLevel>
    </accountSettings>
    <description>MyAccountTerritoryRule desc</description>
    <label>MyAccountTerritoryRule</label>
    <sharedFrom>
      <territory>My_territory</territory>
    </sharedFrom>
    <sharedTo>
      <role>CEO</role>
    </sharedTo>
  </sharingTerritoryRules>
</SharingRules>
```

The following is the definition of two account owner-based sharing rules in API version 32.0 and earlier. The file name corresponds to Account.sharingRules under the accountSharingRules directory. In this definition, ownerRules corresponds to AccountOwnerSharingRule.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<AccountSharingRules xmlns="http://soap.sforce.com/2006/04/metadata">
  <ownerRules>
    <fullName>OwnerRule</fullName>
    <accessLevel>Edit</accessLevel>
    <description>OwnerRule desc</description>
    <label>OwnerRule</label>
    <sharedFrom>
      <role>COO</role>
    </sharedFrom>
    <sharedTo>
      <role>CEO</role>
    </sharedTo>
  </ownerRules>
</AccountSharingRules>
```
The following is the definition of a user criteria-based sharing rule and a user membership-based sharing rule in API version 32.0 and earlier. The file name corresponds to `User.sharingRules` under the `userSharingRules` directory.
The following shows a sample package.xml file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>ObjA__c.*</members>
    <name>SharingCriteriaRule</name>
  </types>
  <types>
    <members>ObjA__c.*</members>
    <name>SharingOwnerRule</name>
  </types>
  <version>46.0</version>
</Package>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### BaseSharingRule

This component is removed as of API version 33.0 and is available in earlier versions only. Use SharingBaseRule instead. Represents the base container for criteria-based and owner-based sharing rules.

This type extends the Metadata metadata type and inherits its fullName field.

**Note:** You can't create a BaseSharingRule component directly. Use the components under the CriteriaBasedSharingRule or OwnerSharingRule metadata types instead.

### Version

BaseSharingRule components are available in API version 24.0 and later.

### Fields

For more information on these fields, see “Sharing Settings” in the Salesforce online help.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sharedTo</td>
<td>String</td>
<td>Required. Specifies who the record should be shared with.</td>
</tr>
<tr>
<td>fullName</td>
<td>String</td>
<td>The unique identifier for API access. The fullName can contain only underscores and alphanumeric characters. It must be unique,</td>
</tr>
</tbody>
</table>
Description
Field Type
Field

description
begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the Metadata component.

Wildcard Support in the Manifest File
This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

CriteriaBasedSharingRule
This component is removed as of API version 33.0 and is available in earlier versions only. Use SharingRules instead. Represents a criteria-based sharing rule. CriteriaBasedSharingRule enables you to share records based on specific criteria.

It extends the BaseSharingRule metadata type and inherits its sharedTo field. For more information, see “Criteria-Based Sharing Rules Overview” in the Salesforce online help.

Note: You can’t create a CriteriaBasedSharingRule component directly. Use the child components instead.

Declarative Metadata File Suffix and Directory Location
CriteriaBasedSharingRule components are stored within the SharingRules component in the criteriaBasedRules field.

Version
CriteriaBasedSharingRule components are available in API version 24.0 and later.

Fields
The following information assumes that you are familiar with implementing sharing rules for standard objects and custom objects. For more information on these fields, see “Sharing Settings” in the Salesforce online help.

Field
Field Type
Description

criteriaItems
FilterItem[]
List that represents the criteria for the sharing rule. The possible values are:
• field
• operation
• value

AccountCriteriaBasedSharingRule
Represents a criteria-based sharing rule for accounts. It extends the CriteriaBasedSharingRule metadata type and inherits its criteriaItems field.
AccountCriteriaBasedSharingRule is used by the `criteriaBasedRules` field in `AccountSharingRules`.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| accountAccessLevel     | ShareAccessLevelNoNone (enumeration of type string) | Required. A value that represents the level of access that the user or group has to the account. The possible values are:  
  • Read  
  • Edit  
  • All |
| booleanFilter          | string                                  | Represents the filter logic of the sharing rule.                           |
| caseAccessLevel        | ShareAccessLevelNoAll (enumeration of type string) | Required. A value that represents the level of access that the user or group has to cases associated with the account. The possible values are:  
  • None  
  • Read  
  • Edit |
| contactAccessLevel     | ShareAccessLevelNoAll (enumeration of type string) | Required. A value that represents the level of access that the user or group has to contacts associated with the account. The possible values are:  
  • None  
  • Read  
  • Edit |
| description            | string                                  | Represents the description of the sharing rule. Maximum of 1000 characters. This field is available in API version 29.0 and later. |
| name                   | string                                  | Required. Name for the sharing rule. Corresponds to `Label` in the user interface. |
| opportunityAccessLevel | ShareAccessLevelNoAll (enumeration of type string) | Required. A value that represents the level of access that a target group is granted for any associated opportunity. The possible values are:  
  • None  
  • Read  
  • Edit |

CampaignCriteriaBasedSharingRule

Represents a criteria-based sharing rule for campaigns. It extends the `CriteriaBasedSharingRule` metadata type and inherits its `criteriaItems` field.

CampaignCriteriaBasedSharingRule is used by the `criteriaBasedRules` field in `CampaignSharingRules`. 
<table>
<thead>
<tr>
<th><strong>Field</strong></th>
<th><strong>Field Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Represents the filter logic of the sharing rule.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Represents the description of the sharing rule. Maximum of 1000 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>campaignAccessLevel</td>
<td>ShareAccessLevel</td>
<td>Required. A value that represents the level of access that a target group is granted for a campaign. The possible values are:</td>
</tr>
<tr>
<td></td>
<td>NONE</td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td>enumeration</td>
<td>• Edit</td>
</tr>
<tr>
<td></td>
<td>of type string</td>
<td>• All</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Name for the sharing rule. Corresponds to <strong>Label</strong> in the user interface.</td>
</tr>
</tbody>
</table>

**CaseCriteriaBasedSharingRule**

Represents a criteria-based sharing rule for cases. It extends the **CriteriaBasedSharingRule** metadata type and inherits its `criteriaItems` field.

CaseCriteriaBasedSharingRule is used by the `criteriaBasedRules` field in **CaseSharingRules**.

<table>
<thead>
<tr>
<th><strong>Field</strong></th>
<th><strong>Field Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Represents the filter logic of the sharing rule.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Represents the description of the sharing rule. Maximum of 1000 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>caseAccessLevel</td>
<td>ShareAccessLevel</td>
<td>Required. A value that represents the level of access being granted for a case. The possible values are:</td>
</tr>
<tr>
<td></td>
<td>READ</td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td>EDIT</td>
<td>• Edit</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Name for the sharing rule. Corresponds to <strong>Label</strong> in the user interface.</td>
</tr>
</tbody>
</table>

**ContactCriteriaBasedSharingRule**

Represents a criteria-based sharing rule for contacts. It extends the **CriteriaBasedSharingRule** metadata type and inherits its `criteriaItems` field.

ContactCriteriaBasedSharingRule is used by the `criteriaBasedRules` field in **ContactSharingRules**.
### LeadCriteriaBasedSharingRule

Represents a criteria-based sharing rule for leads. It extends the CriteriaBasedSharingRule metadata type and inherits its criteriaItems field.

LeadCriteriaBasedSharingRule is used by the criteriaBasedRules field in LeadSharingRules.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Represents the filter logic of the sharing rule.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Represents the description of the sharing rule. Maximum of 1000 characters. This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>contactAccessLevel</td>
<td>ShareAccessLevelReadEdit (enumeration of type string)</td>
<td>Required. A value that represents the level of access being granted to the target group, role, or user for a contact. The possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Name for the sharing rule. Corresponds to Label in the user interface.</td>
</tr>
</tbody>
</table>

### OpportunityCriteriaBasedSharingRule

Represents a criteria-based sharing rule for opportunities. It extends the CriteriaBasedSharingRule metadata type and inherits its criteriaItems field.

OpportunityCriteriaBasedSharingRule is used by the criteriaBasedRules field in OpportunitySharingRules.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Represents the filter logic of the sharing rule.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Represents the description of the sharing rule. Maximum of 1000 characters. This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>leadAccessLevel</td>
<td>ShareAccessLevelReadEdit (enumeration of type string)</td>
<td>Required. A value that represents the level of allowed access. The possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Name for the sharing rule. Corresponds to Label in the user interface.</td>
</tr>
</tbody>
</table>
### CriteriaBasedSharingRule

Represents a criteria-based sharing rule for custom objects. It extends the CriteriaBasedSharingRule metadata type and inherits its criteriaItems field.

CustomObjectCriteriaBasedSharingRule is used by the criteriaBasedRules field in CustomObjectSharingRules.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>Represents the filter logic of the sharing rule.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Represents the description of the sharing rule. Maximum of 1000 characters. This field is available in API version 29.0 and later.</td>
</tr>
</tbody>
</table>
| opportunityAccessLevel | ShareAccessLevelReadEdit (enumeration of type string) | Required. A value that represents the level of allowed access. The possible values are:  
  - Read  
  - Edit |
| name                | string                          | Required. Name for the sharing rule. Corresponds to Label in the user interface. |

### CustomObjectCriteriaBasedSharingRule

Represents a criteria-based sharing rule for custom objects. It extends the CriteriaBasedSharingRule metadata type and inherits its criteriaItems field.

CustomObjectCriteriaBasedSharingRule is used by the criteriaBasedRules field in CustomObjectSharingRules.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| accessLevel         | string                          | Required. A value that represents the type of allowed sharing. The possible values are:  
  - Read  
  - Edit  
  - All |
| booleanFilter       | string                          | Represents the filter logic of the sharing rule.                             |
| description         | string                          | Represents the description of the sharing rule. Maximum of 1000 characters. This field is available in API version 29.0 and later. |
| name                | string                          | Required. Name for the sharing rule. Corresponds to Label in the user interface. |

### UserCriteriaBasedSharingRule

Represents a criteria-based sharing rule for users. It extends the CriteriaBasedSharingRule metadata type and inherits its criteriaItems field.

UserCriteriaBasedSharingRule is used by the criteriaBasedRules field in UserSharingRules.
### Field | Field Type | Description
--- | --- | ---
booleanFilter | string | Represents the filter logic of the sharing rule.
description | string | Represents the description of the sharing rule. Maximum of 1000 characters. This field is available in API version 29.0 and later.
name | string | Required. Name for the sharing rule. Corresponds to **Label** in the user interface.
userAccessLevel | ShareAccessLevelReadEdit (enumeration of type string) | Required. A value that represents the type of allowed sharing. The possible values are:
  - Read
  - Edit

### Declarative Metadata Sample Definition

The following is the definition of two owner-based sharing rules and one criteria-based sharing rule containing two criteria items. The file name corresponds to the `Account.sharingRules` file under the `accountSharingRules` directory.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<AccountSharingRules xmlns="http://soap.sforce.com/2006/04/metadata">
  <ownerRules>
    <fullName>G1Dev_G2New</fullName>
    <sharedTo>
      <group>G2New</group>
    </sharedTo>
    <sharedFrom>
      <group>G1Dev</group>
    </sharedFrom>
    <accountAccessLevel>Read</accountAccessLevel>
    <caseAccessLevel>None</caseAccessLevel>
    <contactAccessLevel>Read</contactAccessLevel>
  </ownerRules>
  <fullName>G2New_R1New</fullName>
  <sharedTo>
    <roleAndSubordinates>R1New</roleAndSubordinates>
  </sharedTo>
  <sharedFrom>
    <group>G2New</group>
  </sharedFrom>
  <accountAccessLevel>Edit</accountAccessLevel>
  <caseAccessLevel>Read</caseAccessLevel>
  <contactAccessLevel>Edit</contactAccessLevel>
  <name>G2New_R1New</name>
  <opportunityAccessLevel>None</opportunityAccessLevel>
</ownerRules>
  <criteriaBasedRules>
    <fullName>AccountCriteria</fullName>
    <sharedTo>
      <group>G1</group>
    </sharedTo>
  </criteriaBasedRules>
</AccountSharingRules>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

OwnerSharingRule

Represents an ownership-based sharing rule. OwnerSharingRule enables you to share records owned by a set of users with another set, using rules that specify the access level of the target user group. This component is removed as of API version 33.0 and is available in earlier versions only.

OwnerSharingRule extends the BaseSharingRule metadata type and inherits its SharedTo field. For more information, see “Sharing Rules” in the Salesforce online help.

⚠️ Note: You can’t create a OwnerSharingRule component directly. Use the child components instead.

Declarative Metadata File Suffix and Directory Location

OwnerSharingRules components are stored within the SharingRules component in the ownerRules field.

Version

OwnerSharingRules components are available in API version 24.0 and later.

Fields

The following information assumes that you are familiar with implementing sharing rules for standard objects and custom objects. For more information on these fields, see “Sharing Settings” in the Salesforce online help.
## AccountOwnerSharingRule

Represents a sharing rule for an account with users other than the owner. It extends the `OwnerSharingRule` metadata type and inherits its `fullName`, `sharedFrom`, and `sharedTo` fields.

AccountOwnerSharingRule is used by the `ownerRules` field in `AccountSharingRules`.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accountAccessLevel</td>
<td><code>ShareAccessLevelNoNone</code></td>
<td>Required. A value that represents the level of access that a group or role has to the account. The possible values are:</td>
</tr>
<tr>
<td></td>
<td>(<code>enumeration</code> of type string)</td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• All</td>
</tr>
<tr>
<td>caseAccessLevel</td>
<td><code>ShareAccessLevelNoAll</code></td>
<td>Required. A value that represents the level of access that a group or role has to cases associated with the account. The possible values are:</td>
</tr>
<tr>
<td></td>
<td>(<code>enumeration</code> of type string)</td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td>contactAccessLevel</td>
<td><code>ShareAccessLevelNoAll</code></td>
<td>Required. A value that represents the level of access that a group or role has to contacts associated with the account. The possible values are:</td>
</tr>
<tr>
<td></td>
<td>(<code>enumeration</code> of type string)</td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td>description</td>
<td><code>string</code></td>
<td>Represents the description of the sharing rule. Maximum of 1000 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 29.0 and later.</td>
</tr>
</tbody>
</table>
**CampaignOwnerSharingRule**

Represents a sharing rule for a campaign with users other than the owner. It extends the `OwnerSharingRule` metadata type and inherits its `fullName`, `sharedFrom`, and `sharedTo` fields.

CampaignOwnerSharingRule is used by the `ownerRules` field in `CampaignSharingRules`.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>campaignAccessLevel</td>
<td>ShareAccessLevelNone</td>
<td>A value that represents the level of access that a group or role is granted for a campaign. The possible values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• All</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Represents the description of the sharing rule. Maximum of 1000 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Name for the sharing rule. Corresponds to <code>Label</code> in the user interface.</td>
</tr>
</tbody>
</table>

**CaseOwnerSharingRule**

Represents a sharing rule for a case with users other than the owner. It extends the `OwnerSharingRule` metadata type and inherits its `fullName`, `sharedFrom`, and `sharedTo` fields.

CaseOwnerSharingRule is used by the `ownerRules` field in `CaseSharingRules`. All the following fields are required.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>caseAccessLevel</td>
<td>ShareAccessLevelReadEdit</td>
<td>Required. A value that represents the level of access that a group or role is granted for a case. The possible values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
</tbody>
</table>
ContactOwnerSharingRule

Represents a sharing rule for a contact with users other than the owner. It extends the OwnerSharingRule metadata type and inherits its fullName, sharedFrom, and sharedTo fields.

ContactOwnerSharingRule is used by the ownerRules field in ContactSharingRules.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| contactAccessLevel  | ShareAccessLevelReadEdit  | Required. A value that represents the level of access that a group or role is granted for a contact. The possible values are:  
|                     | (enumeration of type string) | • Read  
|                     |                           | • Edit  
| description         | string                    | Represents the description of the sharing rule. Maximum of 1000 characters.  
|                     |                           | This field is available in API version 29.0 and later.                      |
| name                | string                    | Required. Name for the sharing rule. Corresponds to Label in the user interface. |

LeadOwnerSharingRule

Represents a sharing rule for a lead with users other than the owner. It extends the OwnerSharingRule metadata type and inherits its fullName, sharedFrom, and sharedTo fields.

LeadOwnerSharingRule is used by the ownerRules field in LeadSharingRules.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| leadAccessLevel     | ShareAccessLevelReadEdit  | Required. A value that represents the level of access that a group or role is granted for a lead. The possible values are:  
|                     | (enumeration of type string) | • Read  
|                     |                           | • Edit  
| description         | string                    | Represents the description of the sharing rule. Maximum of 1000 characters.  
|                     |                           | This field is available in API version 29.0 and later.                      |
### Metadata Types

#### OwnerSharingRule

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Name for the sharing rule. Corresponds to <strong>Label</strong> in the user interface.</td>
</tr>
</tbody>
</table>

**OpportunityOwnerSharingRule**

Represents a sharing rule for an opportunity with users other than the owner. It extends the **OwnerSharingRule** metadata type and inherits its `fullName`, `sharedFrom`, and `sharedTo` fields.

OpportunityOwnerSharingRule is used by the `ownerRules` field in **OpportunitySharingRules**.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Name for the sharing rule. Corresponds to <strong>Label</strong> in the user interface.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Represents the description of the sharing rule. Maximum of 1000 characters.</td>
</tr>
</tbody>
</table>
| opportunityAccessLevel | ShareAccessLevel (enumeration of type string) | Required. A value that represents the level of access that a group or role is granted for an opportunity. The possible values are:  
- Read
- Edit

#### AccountTerritorySharingRule

Represents a rule for sharing an account within a territory. It extends the **OwnerSharingRule** metadata type and inherits its `fullName`, `sharedFrom`, and `sharedTo` fields.

AccountTerritorySharingRule is used by the `ownerRules` field in **AccountTerritorySharingRules**.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| accountAccessLevel  | ShareAccessLevelNoNone (enumeration of type string) | Required. A value that represents the level of access that a Territory or TerritoryAndSubordinates group is granted for an account territory. The possible values are:  
- Read
- Edit
- All

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| caseAccessLevel     | ShareAccessLevelNoAll (enumeration of type string) | Required. A value that represents the level of access that a Territory or TerritoryAndSubordinates group is granted for all child cases to an account. The possible values are:  
- None
- Read

---

864
<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| contactAccessLevel| ShareAccessLevelNoAll (enumeration of type string) | Required. A value that represents the level of access that a Territory or TerritoryAndSubordinates group is granted for all related contacts on an account. The possible values are:  
  - None  
  - Read  
  - Edit |
| description       | string                                         | Represents the description of the sharing rule. Maximum of 1000 characters.  
  This field is available in API version 29.0 and later. |
| name              | string                                         | Required. Name for the sharing rule. Corresponds to Label in the user interface.                                                          |
| opportunityAccessLevel | ShareAccessLevelNoAll (enumeration of type string) | Required. A value that represents the level of access that a Territory or TerritoryAndSubordinates group is granted for all opportunities associated with an account. The possible values are:  
  - None  
  - Read  
  - Edit |

**CustomObjectOwnerSharingRule**

Represents a sharing rule for custom objects. It extends the OwnerSharingRule metadata type and inherits its fullName, sharedFrom, and sharedTo fields.

CustomObjectOwnerSharingRule is used by the ownerRules field in CustomObjectSharingRules.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| accessLevel       | string     | Required. A value that represents the level of access that a group or role is granted to a custom object. The possible values are:  
  - Read  
  - Edit  
  - All |
| description       | string     | Represents the description of the sharing rule. Maximum of 1000 characters.  
  This field is available in API version 29.0 and later. |
| name              | string     | Required. Name for the sharing rule. Corresponds to Label in the user interface.                                                          |
UserMembershipSharingRule

Represents a sharing rule to share members of a group with another group of users. It extends the OwnerSharingRule metadata type and inherits its fullName, sharedFrom, and sharedTo fields.

UserMembershipSharingRule is used by the ownerRules field in UserSharingRules on page 850.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Represents the description of the sharing rule. Maximum of 1000 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is available in API version 29.0 and later.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Name for the sharing rule. Corresponds to Label in the user interface.</td>
</tr>
<tr>
<td>userAccessLevel</td>
<td>ShareAccessLevelReadEdit</td>
<td>Required. A value that represents the level of access that a group or role is granted for a user. The possible values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
</tbody>
</table>

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SharingSet

Represents a sharing set. A sharing set defines an access mapping that grants portal or community users access to objects that are associated with their accounts or contacts. This type extends the Metadata metadata type and inherits its fullName field.

For example, you can grant portal or community users access to all cases related to their account record. Similarly, you can grant portal or community users access to all cases related to a parent account that is identified on the user’s account record. For more information, see “Giving Access to Records Using Sharing Sets” in the Salesforce Help.

File Suffix and Directory Location

SharingSet components have the suffix .sharingSet and are stored in the sharingSets folder.

Version

SharingSet components are available in API version 30.0 and later.

Special Access Rules

To create or update sharing sets, you need the “Customize Application” permission.

Sharing sets are available with these licenses.
• Authenticated Website
• Customer Community Login
• Customer Community Plus
• Partner Community
• Customer Community User
• High Volume Customer Portal
• High Volume Portal
• Overage Authenticated Website User
• Overage High Volume Customer Portal User

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessMappings</td>
<td>AccessMapping[]</td>
<td>A list of access mappings on a sharing set.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The sharing set description. Limit: 255 characters.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The unique identifier for API access. Corresponds to Sharing Set Name on the user interface.</td>
</tr>
<tr>
<td>profiles</td>
<td>string[]</td>
<td>The profiles of users that are granted access to the target objects. Profiles must be associated with a license that can use sharing sets. See Special Access Rules for more information.</td>
</tr>
</tbody>
</table>

AccessMapping

AccessMapping represents an access mapping in the sharing set, which grants access to a target object by looking up to an account or contact associated with the user.

You can grant portal users access to a target object, or to both a target object and its associated objects, such as an account and its contacts and cases.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessLevel</td>
<td>string</td>
<td>Required. The target object access level granted to the portal user. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td>objectField</td>
<td>string</td>
<td>Required. A lookup to the target object, which supports standard or custom fields, or an ID. For accounts or cases associated with entitlements, use Entitlement.Account or Entitlement.Case.</td>
</tr>
<tr>
<td>object</td>
<td>string</td>
<td>Required. The target object to which the portal user is gaining access, and refers to one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Account</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Campaign</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Contact</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Case</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Custom Objects (for example, ObjA__c)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Opportunity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Order</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• ServiceContract</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• User</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• WorkOrder</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Portal users gain access to all order entitlements and order items under an account to which they have access.

<table>
<thead>
<tr>
<th>userField</th>
<th>string</th>
<th>Required. The user’s lookup to an account, contact, or a standard or custom field derived from an account or contact. Either the user or the user’s manager can be used in the lookup. Valid values are:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Account</td>
<td></td>
<td>Account</td>
</tr>
<tr>
<td>• Account.Field</td>
<td></td>
<td>Contact</td>
</tr>
<tr>
<td>• Contact.Field</td>
<td></td>
<td>Contact.RelatedAccount</td>
</tr>
<tr>
<td>• Manager.Account</td>
<td></td>
<td>Manager.Contact</td>
</tr>
</tbody>
</table>

**Field** refers to a standard or custom field based on an account or contact.

### Declarative Metadata Sample Definition

The following is an example of a SharingSet component that grants users access to all contacts whose ReportsTo fields match the users' contacts.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SharingSet xmlns="http://soap.sforce.com/2006/04/metadata">
  <accessMappings>
    <accessLevel>Read</accessLevel>
    <objectField>ReportsTo</objectField>
    <object>Contact</object>
    <userField>Contact</userField>
  </accessMappings>
  <description>User Access Mapping</description>
  <name>User</name>
  <profiles>customer community user</profiles>
</SharingSet>
```
The following is an example of a SharingSet component that grants users access to all cases that are related to an entitlement, which is associated with the user's account.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SharingSet xmlns="http://soap.sforce.com/2006/04/metadata">
  <name>Case</name>
  <accessMappings>
    <accessLevel>Edit</accessLevel>
    <objectField>Entitlement.Account</objectField>
    <object>Case</object>
    <userField>Account</userField>
  </accessMappings>
</SharingSet>
```

The following is an example of a SharingSet component with a list of access mappings.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SharingSet xmlns="http://soap.sforce.com/2006/04/metadata">
  <description>This is a basic sharing set with several access mappings.</description>
  <name>Basic</name>
  <profiles>customer community user</profiles>
  <accessMappings>
    <accessLevel>Read</accessLevel>
    <objectField>Id</objectField>
    <object>Account</object>
    <userField>Account</userField>
  </accessMappings>
  <accessMappings>
    <accessLevel>Edit</accessLevel>
    <objectField>Account</objectField>
    <object>Contact</object>
    <userField>Account</userField>
  </accessMappings>
  <accessMappings>
    <accessLevel>Edit</accessLevel>
    <objectField>Contact</objectField>
    <object>Case</object>
    <userField>Contact</userField>
  </accessMappings>
  <accessMappings>
    <accessLevel>Read</accessLevel>
    <objectField>AccountLookup__c</objectField>
    <object>HVPUAccessible__c</object>
    <userField>Account</userField>
  </accessMappings>
</SharingSet>
```

The following is an example `package.xml` that references the previous definition.

```xml
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <fullName>SharingSetBasic</fullName>
  <types>
    <members>HVPUAccessible__c.AccountLookup__c</members>
    <members>HVPUAccessible__c.ContactLookup__c</members>
    <name>CustomField</name>
  </types>
</Package>
```
Metadata Types

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SiteDotCom

Represents a site for deployment.

SiteDotCom extends the MetadataWithContent type and inherits its fullName and content fields.

Declarative Metadata File Suffix and Directory Location

SiteDotCom components are stored in the siteDotComSites directory of the corresponding package directory.

The file name for the metadata .xml file is [sitename]1.site-meta.xml. The file name for the site file is [sitename]1.site.

When a Lightning community is created, two sites are actually made behind the scenes: CustomSite (of type ChatterNetwork) and SiteDotComSite (of type ChatterNetworkPicasso). These are named, respectively, <community_name> and <community_name>1. The corresponding MDAPI file names are <community_name>.site-meta.xml and <community_name>1.site-meta.xml. 1 is appended to the SiteDotComSite type to keep the name unique from the corresponding CustomSite site.

Note: There is a file size limitation when using the Metadata API to deploy a site from sandbox to production. The assets in the .site file can’t be larger than 40 MB. The site gets created, but the assets show in the new site as broken. To fix the assets, export the assets from the sandbox environment separately and then import them into your new site.

Version

SiteDotCom components are available in API version 30.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>The name of the site you are deploying.</td>
</tr>
<tr>
<td>siteType</td>
<td>(enumeration of type string)</td>
<td>Required. Identifies whether the site is a ChatterNetworkPicasso site for Salesforce</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

Sample XML definitions for SiteDotCom are shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SiteDotCom xmlns="http://soap.sforce.com/2006/04/metadata">
  <label>testsite</label>
  <siteType>Siteforce</siteType>
</SiteDotCom>

<?xml version="1.0" encoding="UTF-8"?>
<SiteDotCom xmlns="http://soap.sforce.com/2006/04/metadata">
  <label>testCommunity</label>
  <siteType>ChatterNetworkPicasso</siteType>
</SiteDotCom>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Skill

Represents the settings for a skill used for field service or to route chats to agents in Chat, such as the name of the skill and which agents the skills are assigned to. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

Skill values are stored in the `<developer_name>.skill` file in the skills directory.

Version

Skill is available in API version 28.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignments</td>
<td>SkillAssignments</td>
<td>Specifies how skills are assigned to Chat users. Skills can be assigned to sets of users or sets of profiles.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Specifies the description of the skill. This field is available in API version 38.0 and later.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Specifies the name of the skill.</td>
</tr>
</tbody>
</table>

**SkillAssignments**

Represents which users and user profiles to whom specific skills are assigned.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>profiles</td>
<td>SkillProfileAssignments</td>
<td>Specifies the profiles that are associated with a specific skill.</td>
</tr>
<tr>
<td>users</td>
<td>SkillUserAssignments</td>
<td>Specifies the users that are associated with a specific skill.</td>
</tr>
</tbody>
</table>

**SkillProfileAssignments**

Represents the profiles that are associated with a specific skill.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>profile</td>
<td>string</td>
<td>Specifies the custom name of the profile associated with a specific skill.</td>
</tr>
</tbody>
</table>

**SkillUserAssignments**

Represents the users that are associated with a specific skill.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>string</td>
<td>Specifies the username of the user associated with a specific skill.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

This is a sample of a skill file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Skill xmlns="http://soap.sforce.com/2006/04/metadata">
    <label>My Skill 1</label>
    <assignments>
        <profiles>
            <profile>LiveAgentOperator</profile>
            <profile>LiveAgentSupervisor</profile>
        </profiles>
        <users>
            <user>jdoe@acme.com</user>
        </users>
    </assignments>
</Skill>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

StandardValueSet

Represents the set of values in a standard picklist field. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

StandardValueSet components have the suffix .standardValueSet and are stored in the standardValueSets folder.

Version

StandardValueSet components are available in API version 38.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupingStringEnum</td>
<td>string</td>
<td>Groups picklist and enumerated values. For example, for the picklist values of the Status field on the Service Appointment object, Done and Finished can both have a grouping string of Completed. Available in API version 41.0 and later.</td>
</tr>
<tr>
<td>sorted</td>
<td>boolean</td>
<td>Required. Indicates whether a global value set is sorted in alphabetical order. By default, this value is false.</td>
</tr>
</tbody>
</table>
### Declarative Metadata Sample Definition

The following example shows a StandardValueSet component that’s defined as the Stage standard picklist on a customized opportunity object.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<StandardValueSet xmlns="http://soap.sforce.com/2006/04/metadata">
    <fullName>OpportunityStage</fullName> <!-- Enum name -->
    <standardValue>
        <fullName>Closed Abandoned</fullName>
    </standardValue>
    <standardValue>
        <fullName>Closed Won</fullName>
    </standardValue>
    <standardValue>
        <fullName>Closed Lost</fullName>
    </standardValue>
</StandardValueSet>

<CustomObject>
    <fullName>Opportunity</fullName>
    <fields>
        <fullName>StageName</fullName> <!-- field name -->
        <label>Stage</label>
        <type>Picklist</type>
    </fields>
</CustomObject>
```

For a list of standard value set names for standard picklists, see [StandardValueSet Names and Standard Picklist Fields](#).

### Wildcard Support in the Manifest File

This metadata type doesn’t support the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see [Deploying and Retrieving Metadata with the Zip File](#).
StandardValueSetTranslation

Contains details for a standard picklist translation. It returns a translated standard value set. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

StandardValueSetTranslation components have the suffix .standardValueSetTranslation and are stored in the standardValueSetTranslations folder.

Translations are stored in a file with a format of ValueSetName-lang.standardValueSetTranslation, where ValueSetName is the global value set’s name, and lang is the translation language.

Version

StandardValueSetTranslation components are available in API version 38.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>valueTranslation</td>
<td>ValueTranslation[]</td>
<td>A list of values from global value sets to be translated.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a StandardValueSetTranslation component. When a value isn’t translated, its translation becomes a comment that’s paired with its masterLabel.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<StandardValueSetTranslation xmlns="http://soap.sforce.com/2006/04/metadata">
    <valueTranslation>
        <masterLabel>Cold</masterLabel>
        <translation><!-- Cold --></translation>
    </valueTranslation>
    <valueTranslation>
        <masterLabel>Hot</masterLabel>
        <translation><!-- Hot --></translation>
    </valueTranslation>
    <valueTranslation>
        <masterLabel>Warm</masterLabel>
        <translation><!-- Warm --></translation>
    </valueTranslation>
</StandardValueSetTranslation>
```

The following is an example package.xml that references the StandardValueSetTranslation definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>AccountRating-fr</members>
    </types>
</Package>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
Translations

**StaticResource**

Represents a static resource file, often a code library in a ZIP file. Static resources allow you to upload content that you can reference in a Visualforce page, including archives (such as .zip and .jar files), images, style sheets, JavaScript, and other files.

This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.

File Suffix and Directory Location

The file suffix is .resource for the template file. The accompanying metadata file is named resource-meta.xml.

Static resource components are stored in the staticresources folder in the corresponding package directory.

Version

Static resources are available in API version 12.0 and later.

Fields

This metadata type contains the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cacheControl</td>
<td>StaticResourceCacheControl</td>
<td>Required. Indicates whether the static resource is marked with a public caching tag so that a third-party delivery client can cache the content. This is a new field in API version 14.0. The valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• Private</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Public</td>
</tr>
<tr>
<td>content</td>
<td>base64Binary</td>
<td>The static resource content. Base 64-encoded binary data. Prior to making an API call, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to binary. This conversion is usually handled for you by a SOAP client. This field is inherited from the MetadataWithContent component.</td>
</tr>
</tbody>
</table>
**Metadata Types**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contentType</td>
<td>string</td>
<td>Required. The content type of the file, for example text/plain.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the static resource.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The static resource name. The name can only contain characters, letters, and the underscore (_) character, must start with a letter, and cannot end with an underscore or contain two consecutive underscore characters.</td>
</tr>
</tbody>
</table>

Inherited from the Metadata component, this field is not defined in the WSDL for this component. It must be specified when creating, updating, or deleting. See `create()` to see an example of this field specified for a call.

### Declarative Metadata Sample Definition

```xml
<?xml version="1.0" encoding="UTF-8"?>
<StaticResource xmlns="http://soap.sforce.com/2006/04/metadata">
  <contentType>text/plain</contentType>
  <description>Test Resource</description>
</StaticResource>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### SynonymDictionary

Represents a set of synonym groups, which are groups of words or phrases that are treated as equivalent in users’ searches. You can define synonym groups to optimize search results for acronyms, variations of product names, and other terminology unique to your organization.

Synonyms are available in Service Cloud features such as Salesforce Knowledge. This type extends the Metadata metadata type and inherits its `fullName` field.

### File Suffix and Directory Location

SynonymDictionary components have the suffix `.synonymDictionary` and are stored in the `synonymDictionaries` folder.

### Version

SynonymDictionary components are available in API version 29.0 and later.

### Special Access Rules

Synonyms must be enabled in your organization. Only users with the “Manage Synonyms” permission can access this object.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groups</td>
<td>SynonymGroup</td>
<td>The synonym groups defined in this dictionary.</td>
</tr>
<tr>
<td>isProtected</td>
<td>boolean</td>
<td>Indicates whether this component is protected (true) or not (false). Protected components cannot be linked to or referenced by components created in the installing organization.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. Specifies the display name of the synonym dictionary.</td>
</tr>
</tbody>
</table>

**SynonymGroup**

Represents a group of synonymous words or phrases.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>languages</td>
<td>Language</td>
<td>Required. Specifies the languages the synonym group applies to. If synonyms are specific to a single language, specify only that language. If the synonyms apply to multiple languages, specify multiple languages for one synonym group.</td>
</tr>
<tr>
<td>terms</td>
<td>string</td>
<td>Required. A word or phrase synonymous with other terms in the group. Maximum of 50 characters. Minimum of two terms per group. Synonym groups are symmetric, which means that if oranges and apples are defined in a synonym group, a search for oranges will return a match for apples, and vice versa for a search for apples.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is an example of a SynonymDictionary component:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SynonymDictionary xmlns="http://soap.sforce.com/2006/04/metadata">
<groups>
    <languages>en_US</languages>
    <terms>Salesforce</terms>
    <terms>salesforce.com</terms>
    <terms>The Customer Company</terms>
    <terms>SFDC</terms>
</groups>
<groups>
    <languages>fr</languages>
    <terms>renault</terms>
    <terms>clio</terms>
</groups>
<label>Sample Dictionary</label>
</SynonymDictionary>
```
The following is an example `package.xml` that references the SynonymDictionary component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>Sample Dictionary</members>
    <name>SynonymDictionary</name>
  </types>
  <version>46.0</version>
</Package>
```

**Usage**

If you have existing synonym groups defined before API version 29.0, your existing groups are associated with a default dictionary called `_Default`.

If you have a set of synonyms that require frequent updates, we recommend assigning the synonym group or groups to a dedicated dictionary with a small number of groups. Each time you deploy an existing dictionary, all of its synonym groups are overwritten. We don’t support deploying updates to only a single synonym group within a dictionary.

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**Territory**

Represents a territory in your organization.

**Declarative Metadata File Suffix and Directory Location**

The file suffix for territory components is `.territory` and components are stored in the `territories` directory of the corresponding package directory.

**Version**

Territory components are available in API version 24.0 and later.

**Fields**

This metadata type extends to subtype `RoleOrTerritory`.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| accountAccessLevel  | string     | Specifies whether users in this territory can access accounts that are assigned to this territory and are otherwise inaccessible. Valid values are:  
  - Read  
  - Edit  |
If your organization’s sharing model for accounts is Public Read/Write, valid values are only Edit and All.

If no value is set for this field, this field value uses the default access level that is specified in the Manage Territory page in Setup.

This field is available in API version 31.0 and later.

**fullName** string

The unique identifier for API access. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the Metadata component. Corresponds to ** Territory Name** in the user interface.

**parentTerritory** string

The territory above this territory in the territory hierarchy.

### Declarative Metadata Sample Definition

The following is the definition of a territory.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Territory xmlns="http://soap.sforce.com/2006/04/metadata">
  <accountAccessLevel>Edit</accountAccessLevel>
  <caseAccessLevel>Edit</caseAccessLevel>
  <contactAccessLevel>Edit</contactAccessLevel>
  <description>Sample Territory</description>
  <mayForecastManagerShare>false</mayForecastManagerShare>
  <name>T22name</name>
  <opportunityAccessLevel>Read</opportunityAccessLevel>
</Territory>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

#### Territory2

Represents the metadata associated with a sales territory in Territory Management 2.0. This type extends the Metadata metadata type and inherits its fullName field. Available only if Territory Management 2.0 has been enabled for your organization.

**File Suffix and Directory Location**

Territory2 components have the suffix territory2 and are stored in the territories folder under the folder for the corresponding Territory2Model.
Version

Territory2 components are available in API version 32.0 and later.

Special Access Rules

The Territory2Model object has a State field in the SOAP API. States include Planning, Active, Archived, and a number of other states, such as Cloning, that indicate that a process is underway. Users who do not have the “Manage Territories” permission can access only territories that belong to the model in Active state. The “Manage Territories” permission is required for deploy() calls for all territory management entities, in addition to the “Modify All Data” permission required by Metadata API. Using retrieve() without the “Manage Territories” permission will return only entities that belong to a Territory2Model in Active state. We recommend against retrieving without the “Manage Territories” permission because the call will retrieve only partial data.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| accountAccessLevel | string     | Specifies whether users in this territory can access accounts that are assigned to this territory and are otherwise inaccessible. Valid values are:  
  - Read  
  - Edit  
  - All  
  If your organization’s sharing model for accounts is Public Read/Write, valid values are only Edit and All. If no value is set for this field, this field value uses the default access level that is specified in Territory2Settings as permitted by the organization’s sharing settings. |
| caseAccessLevel  | string     | Specifies whether users in this territory can access cases that are assigned to this territory and are otherwise inaccessible. Valid values are:  
  - None  
  - Read  
  - Edit  
  No value should be specified if your organization’s sharing model for cases/opportunities is Public Read/Write. If no value is set for this field, this field value uses the default access level that is specified in Territory2Settings as permitted by the organization’s sharing settings. |
| contactAccessLevel | string     | Specifies whether users in this territory can access contacts that are assigned to this territory and are otherwise inaccessible. Valid values are:  
  - None  
  - Read  
  - Edit |
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customFields</td>
<td>FieldValue</td>
<td>Values for custom fields defined on the Territory2 object and used by this territory. Their metadata is captured separately in CustomObject. Note the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Territory2 and Territory2Model objects do not handle values for Text Area (Long), Text Area (Rich), and text-encrypted custom fields.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Fields are referenced using their API names. Compound field types like Location will appear as their constituent column fields. For example, nnn_Latitude__s, nnn_Longitude__s where “nnn” is the field name and the suffixes are the geolocation components.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Values of required custom fields are enforced during the deploy() operation.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the territory.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The user interface label for the territory.</td>
</tr>
<tr>
<td>opportunityAccessLevel</td>
<td>string</td>
<td>Specifies whether users in this territory can access opportunities that are assigned to this territory and are otherwise inaccessible. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Edit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No value should be specified if your organization’s sharing model for cases/opportunities is Public Read/Write. If no value is set for this field, this field value uses the default access level that is specified in Territory2Settings as permitted by the organization’s sharing settings.</td>
</tr>
<tr>
<td>parentTerritory</td>
<td>string</td>
<td>The name of the territory’s parent. When you specify the parent territory, use the developer name. Do not use the “fully qualified” name. Custom fields with no values are retrieved with values of type: &lt;value xsi:nil=”true”/&gt;. You can also use &lt;value xsi:nil=”true”/&gt; syntax to remove existing values in custom fields.</td>
</tr>
<tr>
<td>ruleAssociations</td>
<td>Territory2RuleAssociation</td>
<td>Represents an object assignment rule and its association to a territory. Use the developer name of the rule.</td>
</tr>
<tr>
<td>territory2Type</td>
<td>string</td>
<td>Required. The territory type that the territory belongs to.</td>
</tr>
</tbody>
</table>

**FieldValue**

Represents the values of custom fields on the Territory2 object. Available in API version 32.0 and later.
### Territory2RuleAssociation

Represents the association of an object assignment rule to a territory. Available in API version 32.0 and later.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inherited</td>
<td>boolean</td>
<td>Required. Indicates whether the rule is inherited from a parent territory (true) or local to the current territory (false). Rule inheritance flows from the parent territory where the rule is created to the rule's descendent territories (if any) in the territory model hierarchy. A local rule is created within a single territory and affects that territory only.</td>
</tr>
<tr>
<td>ruleName</td>
<td>string</td>
<td>Required. The name of a rule associated with the territory. ruleName doesn't need to be fully qualified because Metadata API assumes that the rule belongs to the same model as the territory.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following example shows the definition of a Territory2 component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <name>USA</name>
  <description>United States sales</description>
  <accountAccessLevel>Edit</accountAccessLevel>
  <opportunityAccessLevel>Read</opportunityAccessLevel>
  <caseAccessLevel>Edit</caseAccessLevel>
  <contactAccessLevel>Edit</contactAccessLevel>
  <parentTerritory>Worldwide_Sales</parentTerritory>
  <territory2Type>Geo</territory2Type>
  <ruleAssociations>
    <ruleName>AccRule1</name>
    <inherited>True</inherited>
  </ruleAssociations>
  <ruleAssociations>
    <ruleName>AccRule2</name>
    <inherited>False</inherited>
  </ruleAssociations>
  <customFields>
    <name>Activation_DateTime__c</name>
    <value xsi:type="xsd:dateTime">2014-07-16T05:05:00.000Z</value>
  </customFields>
</Territory2>
```
The following is a package.xml sample. FY13 and FY14 represent the names of territory models and demonstrate that rules can have identical developer names within different models. A wildcard character (*) in place of the model name can be used to retrieve all rules in all models in an organization.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>FY13</members>
    <members>FY14</members>
    <name>Territory2Model</name>
  </types>

  <types>
    <members>FY13.USA</members>
    <members>FY13.Worldwide_Sales</members>
    <members>FY14.APAC</members>
    <members>FY14.USA</members>
    <name>Territory2</name>
  </types>

  <version>46.0</version>
</Package>
```

Usage

- Triggers defined on Territory2 will not fire during a `deploy()` operation.
- Territory Management 2.0 components don’t support packaging or change sets and aren’t supported in CRUD calls.

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
Territory2Model

Represents the metadata associated with a territory model in Territory Management 2.0. This type extends the Metadata metadata type and inherits its fullName field. Available only if Territory Management 2.0 has been enabled for your Salesforce org.

File Suffix and Directory Location

Territory2Model components have the suffix territory2Model and are stored in the territory2Models folder.

Version

Territory2Model components are available in API version 32.0 and later.

Special Access Rules

The Territory2Model object has a State field in the SOAP API. States include Planning, Active, Archived, and a number of other states, such as Cloning, that indicate that a process is underway. Users who do not have the “Manage Territories” permission can access only models in Active state. The “Manage Territories” permission is required for deploy() calls for all territory management entities, in addition to the “Modify All Data” permission required by Metadata API. Using retrieve() without the “Manage Territories” permission will return only entities that belong to a Territory2Model in Active state. We recommend against retrieving without the “Manage Territories” permission because the call will retrieve only partial data.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customFields</td>
<td>FieldValue</td>
<td>Custom fields defined on the Territory2Model object and used by this model. Their metadata is captured separately.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Territory2 and Territory2Model objects do not handle values for Text Area (Long), Text Area (Rich), and text-encrypted custom fields.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Fields are referenced using their API names. Compound field types like Location will appear as their constituent column fields. For example, nnn_Latitude__s, nnn_Longitude__s where “nnn” is the field name and the suffixes are the geolocation components.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Values of required custom fields are enforced during the deploy() operation.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description for the territory model.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The user interface label for the territory model.</td>
</tr>
</tbody>
</table>
Declarative Metadata Sample Definition

The following example shows the definition of a Territory2Model component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Territory2Model xmlns="http://soap.sforce.com/2006/04/metadata"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <name>FY13</name>
  <description>Geographic allocation</description>
  <customFields>
    <name>Activation_DateTime__c</name>
    <value xsi:type="xsd:dateTime">2014-07-16T05:05:00.000Z</value>
  </customFields>
  <customFields>
    <name>AutoNumber__c</name>
    <value xsi:type="xsd:string">M# 000001</value>
  </customFields>
  <customFields>
    <name>DeactivationDate__c</name>
    <value xsi:type="xsd:date">2016-07-12</value>
  </customFields>
  <customFields>
    <name>External_Id__c</name>
    <value xsi:nil="true"/>
  </customFields>
</Territory2Model>
```

Usage

- The `retrieve()` call will not return models in these four states: Cloning, Cloning Failed, Deleting, and Deletion Failed.
- Whenever a model is created, its initial state is Planning. You can only do a `deploy()` operation for models in Planning or Active state. The same requirement applies to territories and rules associated with those models. For example, you might have a model in Planning state on a sandbox org, and a model with the same developer name in Archived state on your production org. The `deploy()` operation on production will fail because that model’s state is Archived and that state prevents changes to the model.
- Because of the state restrictions, if you have territory models in different orgs with identical developer names and you attempt a `deploy()` operation, Metadata API will attempt to create new models, but that operation will fail because of the developer name conflict. For example, you might have a model in Planning state on a sandbox org, and a model with the same developer name in Archived state on your production org. The `deploy()` operation on production will fail because that model’s state is Archived and that state prevents changes to the model.
- If you try to delete a model that has territories, then the `delete()` call will change the model’s state to Deleting and cascade delete all territories, rules, and user associations in the model. Deleting may take some time depending on the number of territories in the model.
- Whenever a model is created, its initial state is Planning. If a model with the same developer name already exists, it will already have a state, so we do not include the State field in Territory2.
- Territory Management 2.0 components don’t support packaging or change sets and aren’t supported in CRUD calls.
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Territory2Rule

Represents the metadata associated with a territory assignment rule associated with an object, such as Account, in Territory Management 2.0. This type extends the `Metadata` metadata type and inherits its `fullName` field. Available only if Territory Management 2.0 has been enabled for your Salesforce org.

File Suffix and Directory Location

Territory2Rule components have the suffix `territory2Rule` and are stored in the `rules` folder under the folder for the corresponding Territory2Model.

Version

Territory2Rule components are available in API version 32.0 and later.

Special Access

The Territory2Model object has a `State` field in the SOAP API. States include `Planning`, `Active`, `Archived`, and a number of other states, such as `Cloning`, that indicate that a process is underway. Users who do not have the “Manage Territories” permission can access only rules that belong to the model in `Active` state. The “Manage Territories” permission is required for `deploy()` calls for all territory management entities, in addition to the “Modify All Data” permission required by Metadata API. Using `retrieve()` without the “Manage Territories” permission will return only entities that belong to a Territory2Model in `Active` state. We recommend against retrieving without the “Manage Territories” permission because the call will retrieve only partial data. The SOAP API and the user interface require that a user attempting to create or edit a rules has field-level security access to the fields referenced in the rule item. This restriction is relaxed for Metadata API `deploy()` operations, as those require “Modify All Data” and “Manage Territories” permissions. “Modify All Data” is the base permission requirement for all Metadata API operations.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Indicates whether the rule is active (true) or inactive (false). Via the API, active rules run automatically when object records are created and edited. The exception is when the value of the <code>IsExcludedFromRealign</code> field on an object record is true, which prevents record assignment rules from evaluating that record.</td>
</tr>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>An advanced filter condition. For example: (1 AND 2) OR 3. Numbering must start at 1 and must be contiguous.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The user interface label for the rule.</td>
</tr>
</tbody>
</table>
**Field Name** | **Field Type** | **Description**
--- | --- | ---
objectType | string | Required. The object that the rule is defined for. For API version 32.0, the only available object is Account.
ruleItems | Territory2RuleItem on page 888 | The items that define a rule's the selection criteria, such as Billing State equals California.

### Territory2RuleItem

Represents the association of a rule item to a rule. Available in API version 32.0 and later.

**Field Name** | **Field Type** | **Description**
--- | --- | ---
field | string | The standard or custom object field that the rule item operates on.
operation | FilterOperation (enumeration of type string) | The criterion to apply for the rule item. For example: equals or starts with.
value | string | The field value or values to evaluate. For example: if the field is Billing ZIP/Postal Code, a value might be 94105.

### Declarative Metadata Sample Definition

The following example shows the definition of a Territory2RuleItem component.

```xml
<Territory2Rule xmlns="http://soap.sforce.com/2006/04/metadata">
  <label>Northern CA</label>
  <description>To capture northern CA based accounts</description>
  <objectType>Account</objectType>
  <active>True</active>
  <ruleItems>
    <field>BillingZip</field>
    <operation>contains</operation>
    <value>94105,94404,94536/value>
  </ruleItems>
  <ruleItems>
    <field>Industry</field>
    <operation>equals</operation>
    <value>IT</value>
  </ruleItems>
  <ruleItems>
    <field>someCustomField__c</field>
    <operation>greater_than</operation>
    <value>50000</value>
  </ruleItems>
  <booleanFilter>(1 OR 2) AND 3</booleanFilter>
</Territory2Rule>
```
The following is a `package.xml` sample. `FY13` and `FY14` represent names of territory models and demonstrate that rules can have identical developer names within different models. A wildcard character (*) in place of the model name can be used to retrieve all rules in all models in an org.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">

  <types>
    <members>FY13</members>
    <members>FY14</members>
    <name>Territory2Model</name>
  </types>

  <types>
    <members>FY13.AccRule1</members>
    <members>FY14.AccRule1</members>
    <name>Territory2Rule</name>
  </types>

  <version>46.0</version>
</Package>
```

**Usage**

- A territory rule can have up to 10 rule items.
- The sort order of rule items is implicitly derived from the position of the rule items in the XML.
- Rules can't be run via Metadata API.
- Territory Management 2.0 components don't support packaging or change sets and aren't supported in CRUD calls.

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**Territory2Type**

Represents the metadata for a category of territories in Territory Management 2.0. Every Territory2 must have a Territory2Type. This type extends the Metadata metadata type and inherits its `fullName` field. Available only if Enterprise Territory Management has been enabled for your Salesforce org.

**File Suffix and Directory Location**

Territory2Type components have the suffix `territory2Type` and are stored in the `territory2Types` folder.

**Version**

Territory2Type components are available in API version 32.0 and later.
**Special Access Rules**

Users without the “Manage Territories” permission will be able to retrieve all the Territory2Types in the org. “Manage Territories” permission is required for the deploy() operation, in addition to the “Modify All Data” permission required by the Metadata API.

**Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the territory type.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The user interface label for the territory type.</td>
</tr>
<tr>
<td>priority</td>
<td>int</td>
<td>Required. Used for Filter-Based Opportunity Territory Assignment (Pilot in Spring '15 / Metadata API version 33). Lets you specify a priority for a territory type. For opportunity assignments, the filter examines all territories assigned to the account that the opportunity is assigned to. The account-assigned territory whose territory type priority is highest is then assigned to the opportunity. The priority field value on each territory type must be unique. Further, if there are multiple territories with the same territory type (and therefore the same priority) assigned to the account, no territory is not assigned to the opportunity.</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following example shows the definition of a Territory2Type component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Territory2Type xmlns="http://soap.sforce.com/2006/04/metadata">
  <name>Geo</name>
  <description>Geographic allocation</description>
</Territory2Type>
```

**Usage**

Territory Management 2.0 components don’t support packaging or change sets and aren’t supported in CRUD calls.

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**TimeSheetTemplate**

Represents a template for creating time sheets in Field Service Lightning. This type extends the Metadata metadata type and inherits its fullName field.
File Suffix and Directory Location

TimeSheetTemplate components have the suffix `timeSheetTemplate` and are stored in the `timeSheetTemplates` folder.

Version

TimeSheetTemplate components are available in API version 46.0 and later.

Special Access Rules

Field Service Lightning must be enabled. Users must have the Customize Application and Time Sheet Template permissions.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Indicates whether the time sheet template is active (true) or not (false).</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The time sheet template's description.</td>
</tr>
</tbody>
</table>
| frequency           | TimeSheetFrequency (enumeration of type string) | Required. Defines the frequency of the time sheet creation period. One of the following values:  
  - Daily  
  - Weekly  
  - EveryTwoWeeks  
  - TwiceAMonth  
  - Monthly |
| masterLabel         | string                      | Required. The name of the time sheet template.                              |
| startDate           | date                        | Required. The date when the time sheet takes effect.                        |
| timeSheetTemplateAssignments | TimeSheetTemplateAssignment | A list of profiles that the template is assigned to.                        |
| workWeekEndDay      | DaysOfWeek (enumeration of type string) | Required. The end day of the template's work week. One of the following values:  
  - Monday  
  - Tuesday  
  - Wednesday  
  - Thursday  
  - Friday  
  - Saturday  
  - Sunday |
TimeSheetTemplateAssignment

Returns a quick action that’s associated with an EmbeddedServiceLiveAgent setup. The quick action includes the pre-chat form fields that the embedded chat window displays and shows the order in which the fields are displayed.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignedTo</td>
<td>string</td>
<td>The user profiles that a time sheet template is assigned to.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a TimeSheetTemplate file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<TimeSheetTemplate xmlns="http://soap.sforce.com/2006/04/metadata">
  <active>true</active>
  <description>Time Sheet Template description</description>
  <frequency>Daily</frequency>
  <masterLabel>label</masterLabel>
  <startDate>2018-10-18</startDate>
  <timeSheetTemplateAssignments>
    <assignedTo>admin</assignedTo>
  </timeSheetTemplateAssignments>
  <timeSheetTemplateAssignments>
    <assignedTo>standard</assignedTo>
  </timeSheetTemplateAssignments>
  <workWeekEndDay>Tuesday</workWeekEndDay>
  <workWeekStartDay>Monday</workWeekStartDay>
</TimeSheetTemplate>
```

The following is an example package.xml that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>TimeSheetTemplate</name>
  </types>
</Package>
```
**Metadata Types**

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### TopicsForObjects

Represents the ability to assign topics to objects or to remove topic assignments.

### File Suffix and Directory Location

TopicsForObjects components have the suffix `.topicsForObjects` and are stored in the `topicsForObjects` folder of the corresponding package directory.

### Version

TopicsForObjects components are available in API version 41.0 and later.

### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableTopics</td>
<td>boolean</td>
<td>Required. When true, indicates whether users can assign topics or remove topic assignments. When false, users can’t assign or remove topics. Upon org creation, this value is true for the following objects:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Account</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Asset</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Campaign</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Case</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Contact</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Content Document</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Contract</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Event</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Lead</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Opportunity</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Order</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Solution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Task</td>
</tr>
</tbody>
</table>
## Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>For all remaining standard objects and custom objects, the default is false.</td>
</tr>
<tr>
<td>entityApiName</td>
<td>string</td>
<td>Required. Indicates the object’s API name for enabling topics.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of a TopicsForObjects component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<TopicsForObjects xmlns="http://soap.sforce.com/2006/04/metadata">
  <enableTopics>false</enableTopics>
  <entityApiName>Account</entityApiName>
</TopicsForObjects>
```

The following is an example `package.xml` that references the previous definition.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>TopicsForObjects</name>
  </types>
  <version>41.0</version>
</Package>
```

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character `*` (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### TransactionSecurityPolicy

Represents a transaction security policy definition. Transaction Security policies give you a way to look through events in your organization and specify actions to take when certain combinations occur.

This type extends the `Metadata` metadata type and inherits its `fullName` field.

### File Suffix and Directory Location

`TransactionSecurityPolicy` components have the suffix `.transactionSecurityPolicy` and are stored in the `transactionSecurityPolicies` folder.

### Version

`TransactionSecurityPolicy` components are available in API version 35.0 and later.
## Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>TransactionSecurityAction</td>
<td>Required. Describes the action to take when the matching Transaction Security policy is triggered.</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. If true, the policy is enabled and is actively monitoring its event.</td>
</tr>
<tr>
<td>apexClass</td>
<td>string</td>
<td>Required for Apex-based policies, and optional for all other policies. The name of the class that implements the TxnSecurity.PolicyCondition interface for this policy. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the policy.</td>
</tr>
<tr>
<td>developerName</td>
<td>string</td>
<td>This unique name prevents conflicts with other policies that have the same masterLabel. This name can contain only underscores and alphanumeric characters, and must be unique in your org. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores.</td>
</tr>
</tbody>
</table>
| eventName    | TransactionSecurityEventName (enumeration of type string) | Used in Real-Time Event Monitoring (Beta) only. Indicates the name of the event the policy monitors. This field is available in API 45.0 and later. Valid values are:  
  - AdminSetupEvent—Tracks user operations for managing encryption keys in your org.  
  - ApiEvent—Tracks user API calls in your org.  
  - ListViewEvent—Tracks when users access data with list views using Salesforce Classic or the API. It doesn’t track list views of Setup entities.  
  - LoginEvent—Tracks user login events in your org.  
  - ReportEvent—Tracks when reports are run in your org. |
| eventType    | MonitoredEvents (enumeration of type string) | Required for Apex-based policies, and optional for all other policies. Indicates which type of event is being monitored. Valid values are:  
  - AccessResource—Notifies you when the selected resource has been accessed.  
  - AuditTrail—Reserved for future use.  
  - DataExport—Notifies you when the selected object type has been exported using the Data Loader API client.  
  - Entity—Notifies you on use of an object type such as an authentication provider or Chatter comment.  
  - Login—Notifies you when a user logs in. |
| executionUser| string                      | Required for policies of type CustomApexPolicy. The name of the user to notify when the policy is triggered, if any notifications |
### TransactionSecurityPolicyMetadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flow</td>
<td>string</td>
<td>Required only for policies of type CustomConditionBuilderPolicy. The ID of the Flow object that contains the logic the Condition Builder transaction security policy. Available in API version 46.0 and later.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>The master label for this object. This display value is the internal label that is not translated.</td>
</tr>
<tr>
<td>resourceName</td>
<td>string</td>
<td>Required for Apex-based policies, and optional for all other policies. A resource used to narrow down the conditions under which the policy triggers. For example, with a DataExport event, you can select a resource Lead to specifically monitor export activity occurring on your Lead entities. The resources available depend on the Event Type field. Valid resources are grouped below by event type.</td>
</tr>
<tr>
<td>1.</td>
<td>AccessResource</td>
<td>ConnectedApplication, Reports</td>
</tr>
<tr>
<td>2.</td>
<td>DataExport</td>
<td>Account, Case, Contact, Lead, Opportunity</td>
</tr>
<tr>
<td>3.</td>
<td>Entity</td>
<td>AuthProvider, ChatterMessage, FeedComment, FeedItem, Idea, Question</td>
</tr>
<tr>
<td>4.</td>
<td>Login</td>
<td>LoginHistory</td>
</tr>
<tr>
<td>type</td>
<td>TxnSecurityPolicyType (enumeration of type string)</td>
<td>The type of validation that the policy uses. The valid values are: CustomApexPolicy—Created with Apex editor. CustomConditionBuilderPolicy—Created with Condition Builder. The default value is CustomApexPolicy.</td>
</tr>
</tbody>
</table>

**TransactionSecurityAction**

Describes the action to take when the matching Transaction Security policy is triggered.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>boolean</td>
<td>Required. If true, the requested operation is blocked. This action only applies to Login and AccessResource events.</td>
</tr>
<tr>
<td>endSession</td>
<td>boolean</td>
<td>Required. If true, a current session must be closed before a new session can be started. This action only applies to Login events.</td>
</tr>
<tr>
<td>freezeUser</td>
<td>boolean</td>
<td>Required. If true, the user that triggered the policy is frozen. This action only applies to Chatter resources for Entity events.</td>
</tr>
<tr>
<td>notifications</td>
<td>TransactionSecurityNotification[]</td>
<td>Specifies how to notify the system administrator when the action is triggered. There can be none, one, or multiple notifications.</td>
</tr>
</tbody>
</table>

```markdown

896
```
TransactionSecurityPolicy

Describes who to notify and how to notify them when the matching Transaction Security policy is triggered.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>twoFactorAuthentication</td>
<td>boolean</td>
<td>Required. If true, two-factor authentication is required for a higher level of access before the requested operation can continue. This action only applies to Login and AccessResource events.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a TransactionSecurityPolicy component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <action>
    <block>true</block>
    <endSession>false</endSession>
    <notifications>
      <inApp>false</inApp>
      <sendEmail>true</sendEmail>
      <user>admin@your.org</user>
    </notifications>
    <twoFactorAuthentication>false</twoFactorAuthentication>
  </action>
</TransactionSecurityPolicy>
```

The following is an example package manifest used to deploy or retrieve the transaction security metadata for an organization.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>MySecurityPolicy</members>
    <name>TransactionSecurityPolicy</name>
  </types>
  <version>35.0</version>
</Package>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

Translations

This metadata type allows you to work with translations for various supported languages. The ability to translate component labels is part of the Translation Workbench. For more information, see "Enable and Disable the Translation Workbench" in the Salesforce online help.

This type extends the Metadata metadata type and inherits its fullName field.

Language

A two-character language code identifies each language, such as en, or a five-character locale code, such as en_US.

Note: Setting a default locale is different from setting a default language.

Salesforce offers full support for the following languages.

- Chinese (Simplified): zh_CN
- Chinese (Traditional): zh_TW
- Danish: da
- Dutch: nl_NL
- English: en_US
- Finnish: fi
- French: fr
- German: de
- Italian: it
- Japanese: ja
- Korean: ko
- Norwegian: no
- Portuguese (Brazil): pt_BR
- Russian: ru
- Spanish: es
- Spanish (Mexico): es_MX Spanish (Mexico) falls back to Spanish for customer-defined translations.
- Swedish: sv
- Thai: th

The Salesforce user interface is fully translated to Thai, but Help is in English.

End-user languages are useful if you have a multilingual organization or partners who speak languages other than your company’s default language. For end-user languages, Salesforce provides translated labels for all standard objects and pages, except admin pages, Setup, and Help. When you specify an end-user language, labels and Help that aren’t translated appear in English. End-user languages are intended only for personal use by end users. Don’t use end-user languages as corporate languages. Salesforce doesn’t provide customer support in end-user languages.

End-user languages include:
Note: Salesforce provides limited support for right-to-left (RTL) languages—Arabic and Hebrew—only for the following features.

- Chat
- Cases
- Accounts
- Lightning Knowledge

These features aren’t supported in Lightning Experience, the Salesforce app, any other mobile app or mobile browser, or any user interface, except Salesforce Classic. Salesforce doesn’t guarantee that RTL languages function correctly with any other Salesforce feature. There are no plans to expand the list of supported features.

Features that aren’t supported for RTL languages include, but are not limited to, the following.

- Report Builder
- Generating quote PDFs
- Customizable forecasting
- Emails
- Knowledge in Salesforce Classic
- Feeds
- Communities
- Certain search features, including lemmatization and synonym groups

The absence of a feature from this list does not imply support.

In situations where Salesforce doesn’t provide default translations, use platform-only languages to localize apps and custom functionality that you’ve built on the Salesforce App Cloud. You can translate items such as custom labels, custom objects, and field names. You can also rename most standard objects, labels, and fields. Informational text and non-field label text aren’t translatable.

Platform-only languages are available in all places where you can select a language in the application. However, when you select a platform-only language, all standard Salesforce labels default to English or, in select cases, to an end-user or fully supported language.
Note: Language support is closely tied to the API version. For example, we introduced support for Belgian Dutch (nl_BE) in API version 40.0. To take advantage of this language, you must use API version 40.0 or later. In general, we recommend using the most recent version of the API to make the most of our language features.

- Albanian: sq
- Afrikaans (South Africa): af_ZA
- Arabic (Algeria): ar_DZ
- Arabic (Bahrain): ar_BH
- Arabic (Egypt): ar_EG
- Arabic (Iraq): ar_IQ
- Arabic (Jordan): ar_JO
- Arabic (Kuwait): ar_KW
- Arabic (Lebanon): ar_LB
- Arabic (Libya): ar_LY
- Arabic (Morocco): ar_MA
- Arabic (Oman): ar_OM
- Arabic (Qatar): ar_QA
- Arabic (Saudi Arabia): ar_SA
- Arabic (Sudan): ar_SD
- Arabic (Syria): ar_SY
- Arabic (Tunisia): ar_TN
- Arabic (United Arab Emirates): ar_AE
- Arabic (Yemen): ar_YE
- Armenian: hy
- Basque: eu
- Bosnian: bs
- Bengali (Bangladesh): bn_BD
- Bengali (India): bn_IN
- Burmese: my
- Catalan: ca
- Chinese (Simplified—Singapore): zh_SG
- Chinese (Traditional—Hong Kong): zh_HK
- Dutch (Belgium): nl_BE
- English (Australia): en_AU
- English (Canada): en_CA
- English (Hong Kong): en_HK
- English (India): en_IN
- English (Ireland): en_IE
- English (Malaysia): en_MY
- English (New Zealand): en_NZ
- English (Philippines): en_PH
- English (Singapore): en_SG
- English (South Africa): en_ZA
- Estonian: et
- French (Belgium): fr_BE
- French (Canada): fr_CA
- French (Luxembourg): fr卢
- French (Switzerland): fr_CH
- Georgian: ka
- German (Austria): de_AT
- German (Belgium): de_BE
- German (Luxembourg): de卢
- German (Switzerland): de_CH
- Gujarati (India): gu_IN
- Hindi: hi
- Icelandic: is
- Irish: ga
- Italian (Switzerland): it_CH
- Kannada (India): kn_IN
- Latvian: lv
- Lithuanian: lt
- Luxembourgish: lb
- Macedonian: mk
- Malay: ms
- Malayalam (India): ml_IN
- Maltese: mt
- Maori: mi
- Maori (New Zealand): mi_NZ
- Marathi (India): mr_IN
- Romanian (Moldova): ro_MD
- Montenegrin: sh_ME
- Romansh: rm
- Russian (Kazakhstan): ru_KZ
- Serbian (Cyrillic): sr
- Serbian (Latin): sh
- Spanish (Argentina): es_AR
- Spanish (Bolivia): es_BO
- Spanish (Chile): es_CL
- Spanish (Colombia): es_CO
- Spanish (Costa Rica): es_CR
- Spanish (Dominican Republic): es_DO
Translations are stored in a file with a format of `localeCode.translation`, where `localeCode` is the locale code of the translation language. For example, the file name for German translations is `de.translation`. The supported locale codes are listed in Language.

Custom object translations are stored in the `translations` folder in the corresponding package directory.

Version
Translations components are available in API version 14.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customApplications</td>
<td>CustomApplicationTranslation[]</td>
<td>A list of custom application translations.</td>
</tr>
<tr>
<td>customLabels</td>
<td>CustomLabelTranslation[]</td>
<td>A list of custom label translations.</td>
</tr>
<tr>
<td>customPageWebLinks</td>
<td>CustomPageWebLinkTranslation[]</td>
<td>A list of translations for web links defined in a home page component.</td>
</tr>
<tr>
<td>customTabs</td>
<td>CustomTabTranslation[]</td>
<td>A list of custom tab translations.</td>
</tr>
</tbody>
</table>
### Description

A list of flow translations. Only Flow and AutolaunchedFlow types are supported for translation. This field is available in API version 41.0 and later.

### Required. The language code; for example, de for German.

Inherited from Metadata, this field is not defined in the WSDL for this metadata type. It must be specified when creating, updating, or deleting. See [createMetadata()](#) to see an example of this field specified for a call.

### A list of global picklist translations. A global picklist's value set is inherited by all the custom picklist fields that are based on it.

This field is available in API version 37.0 only and is removed from later versions.

### A list of global (rather than object-specific) quick actions.

### A list of report type translations.

### A list of s-control translations.

---

### CustomApplicationTranslation

CustomApplicationTranslation contains details for a custom application translation. For more details, see CustomApplication.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The translated custom application name. Maximum of 765 characters.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the custom application.</td>
</tr>
</tbody>
</table>

### CustomLabelTranslation

CustomLabelTranslation contains details for a custom label translation. For more details, see CustomLabels.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The translated custom label name. Maximum of 765 characters.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The custom label name.</td>
</tr>
</tbody>
</table>
CustomPageWebLinkTranslation

CustomPageWebLinkTranslation contains details for a translation of a web link defined in a home page component. For more details, see CustomPageWebLink.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The translated web link.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the web link.</td>
</tr>
</tbody>
</table>

CustomTabTranslation

CustomTabTranslation contains details for a translation of a custom tab. For more details, see CustomTab.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The translated custom tab name.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The custom tab name.</td>
</tr>
</tbody>
</table>

FlowDefinitionTranslation

FlowDefinitionTranslation contains details for a translation of a flow definition. For more details, see FlowDefinition. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flows</td>
<td>FlowTranslation[]</td>
<td>A list of flow version translations for the flow definition.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The API name for the flow definition.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>A translated label for the flow definition.</td>
</tr>
</tbody>
</table>

By default, flow definitions inherit the label of the active flow version. If you provide a label here, the definition label no longer inherits changes to the active version label.

FlowTranslation

FlowTranslation contains details for a translation of a flow version. For more details, see Flow. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>choices</td>
<td>FlowChoiceTranslation[]</td>
<td>A list of choice translations for the flow version.</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Description</th>
<th>Field Type</th>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required. The API name for the flow version.</td>
<td></td>
<td><strong>fullName</strong></td>
<td>string</td>
</tr>
<tr>
<td>A unique name for the flow that contains only underscores and alphanumeric characters. The name must be unique across the org, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. To deploy or retrieve a version, you can specify the version number. For example, sampleFlow-3 specifies version 3 of the flow whose unique name is sampleFlow. If you don’t specify a version number, the flow is the latest version. In API version 43.0 and earlier, this field included the version number. In API version 44 and later, this field no longer includes the version number.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A translated label for the flow version.</td>
<td></td>
<td><strong>label</strong></td>
<td>string</td>
</tr>
<tr>
<td>A list of screen translations for the flow version. Available in API version 43.0 and later.</td>
<td></td>
<td><strong>screens</strong></td>
<td>FlowScreenTranslation[]</td>
</tr>
<tr>
<td>A list of stage translations for the flow version. Available in API version 43.0 and later.</td>
<td></td>
<td><strong>stages</strong></td>
<td>FlowStageTranslation on page 906[]</td>
</tr>
</tbody>
</table>

### FlowChoiceTranslation

FlowChoiceTranslation contains details for a translation of a choice in a flow version. For more details, see FlowChoice in Flow. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Description</th>
<th>Field Type</th>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>A translated label for the choice.</td>
<td></td>
<td><strong>choiceText</strong></td>
<td>string</td>
</tr>
<tr>
<td>Required. A unique name for the choice.</td>
<td></td>
<td><strong>name</strong></td>
<td>string</td>
</tr>
<tr>
<td>A translated choice input for the choice.</td>
<td></td>
<td><strong>userInput</strong></td>
<td>FlowChoiceUserInputTranslation</td>
</tr>
</tbody>
</table>

### FlowChoiceUserInputTranslation

FlowChoiceUserInputTranslation contains details for a translation of a choice input. For more details, see FlowChoiceUserInput in Flow. Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Description</th>
<th>Field Type</th>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>A translated label for the choice input.</td>
<td></td>
<td><strong>promptText</strong></td>
<td>string</td>
</tr>
<tr>
<td>A translated validation rule for the choice input.</td>
<td></td>
<td><strong>validationRule</strong></td>
<td>FlowInputValidationRuleTranslation</td>
</tr>
</tbody>
</table>
FlowInputValidationRuleTranslation

FlowInputValidationRuleTranslation contains details for a translation of a validation rule. For more details, see FlowInputValidationRule in Flow.

Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorMessage</td>
<td>string</td>
<td>A translated error message for the validation rule.</td>
</tr>
</tbody>
</table>

FlowScreenTranslation

FlowScreenTranslation contains details for a translation of a screen. For more details, see FlowScreen in Flow.

Available in API version 41.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>FlowScreenField[]</td>
<td>A list of screen component translations for the screen.</td>
</tr>
<tr>
<td>helpText</td>
<td>string</td>
<td>Translated help text for the screen.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. An API name for the screen.</td>
</tr>
<tr>
<td>pausedText</td>
<td>string</td>
<td>A translated pause confirmation message for the screen.</td>
</tr>
</tbody>
</table>

FlowScreenFieldTranslation

FlowScreenFieldTranslation contains details for a translation of a screen component. For more details, see FlowScreenField in Flow.

Available in API version 41.0 and later.

Note: Translation isn't supported for screen components that require Lightning runtime.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldText</td>
<td>string</td>
<td>A translated label for the screen component.</td>
</tr>
<tr>
<td>helpText</td>
<td>string</td>
<td>Translated help text for the screen component.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. An API name for the screen component.</td>
</tr>
<tr>
<td>validationRule</td>
<td>FlowInputValidationRuleTranslation</td>
<td>Translated validation rule for the screen component.</td>
</tr>
</tbody>
</table>

FlowStageTranslation

FlowStageTranslation contains details for a translation of a stage in a flow version. For more details, see FlowStage in Flow.

Available in API version 43.0 and later.
**FlowTextTemplateTranslation**

FlowTextTemplateTranslation is available only in flows created via Salesforce Surveys and represents the translation details for the text on all the pages of a survey.

Available in API version 45.0 and later.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Unique name for the text template.</td>
</tr>
<tr>
<td>text</td>
<td>string</td>
<td>Translated text for the text template.</td>
</tr>
</tbody>
</table>

**GlobalPicklistTranslation**

*Note:* GlobalPicklistTranslation is available in API version 37.0 only and is removed from later versions.

GlobalPicklistTranslation contains details for a global picklist translation.

Translations are stored in a file with a format of `globalPicklistName__e-lang.objectTranslation`, where `globalPicklistName__e` is the global picklist name, and `lang` is the translation language. To reference a global picklist translation value, use `globalPicklistName__e.value1`, where `value1` is the translated value for the user interface.

Here’s what translations look like for a global picklist.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Translations xmlns="http://soap.sforce.com/2006/04/metadata">
<globalPicklists>
  <name>transpicklist</name>
  <picklistValues>
    <masterLabel>Three</masterLabel>
    <translation>Trois</translation>
  </picklistValues>
  <picklistValues>
    <masterLabel>Four</masterLabel>
    <translation>Quatre</translation>
  </picklistValues>
</globalPicklists>
</Translations>
```

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. Represents the name of a global picklist to be translated.</td>
</tr>
<tr>
<td>picklistValues</td>
<td>PicklistValueTranslation[]</td>
<td>A list of picklist values from global picklists to be translated.</td>
</tr>
</tbody>
</table>
### GlobalQuickActionTranslation
GlobalQuickActionTranslation contains details for the translation of a quick action, globally. For more information, see QuickAction.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The translated quick action name, globally.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The quick action name.</td>
</tr>
</tbody>
</table>

### ReportTypeTranslation
ReportTypeTranslation contains details for a translation of a custom report type. For more details, see ReportType.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The translated report type description.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The translated report type name.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the report type.</td>
</tr>
<tr>
<td>sections</td>
<td>ReportTypeSectionTranslation[]</td>
<td>A list of report type section translations.</td>
</tr>
</tbody>
</table>

### ReportTypeSectionTranslation
ReportTypeSectionTranslation contains details for a report type section translation.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns</td>
<td>ReportTypeColumnTranslation[]</td>
<td>A list of report type column translations.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The translated report type section name.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the report type section.</td>
</tr>
</tbody>
</table>

### ReportTypeColumnTranslation
ReportTypeColumnTranslation contains details for a report type column translation.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The translated report type column name.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The report type column name.</td>
</tr>
</tbody>
</table>

### ScontrolTranslation
Important: Visualforce pages supersede s-controls. Organizations that haven’t previously used s-controls can’t create them. Existing s-controls are unaffected, and can still be edited.
ScontrolTranslation contains details for a translation of an s-control. For more information, see “About S-Controls” in the Salesforce online help.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The translated s-control name.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the s-control.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

A sample XML definition of a translations component is shown below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Translations xmlns="http://soap.sforce.com/2006/04/metadata">
  <customApplications>
    <label>Angebot-Manager</label>
    <name>Quote Manager</name>
  </customApplications>
  <customLabels>
    <label>Dieses ist ein manuelles Angebot</label>
    <name>quoteManual</name>
  </customLabels>
</Translations>
```

### Usage

When you use the `retrieve()` call to get translations in your organization, the files returned in the `.translations` folder only include translations for the other metadata types referenced in `package.xml`. For example, the following `package.xml` file contains `types` elements that match all custom applications, custom labels, web links defined in home page components, custom tabs, report types, and s-controls. Translations for all these metadata types are returned because each metadata type is explicitly listed in `package.xml`.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*</members>
    <name>CustomApplication</name>
  </types>
  <types>
    <members>*</members>
    <name>CustomLabels</name>
  </types>
  <types>
    <members>*</members>
    <name>CustomPageWebLink</name>
  </types>
  <types>
    <members>*</members>
    <name>CustomTab</name>
  </types>
</Package>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

SEE ALSO:
  - CustomLabels

UserCriteria

Represents the member criteria to use in community moderation rules. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

UserCriteria components have the suffix community_name.user_criteria_developer_name.userCriteria and are stored in the UserCriteria folder.

Version

UserCriteria components are available in API version 39.0 and later.

Special Access Rules

To view, create, edit, and delete moderation rules, you need the Manage Communities or Create and Set Up Communities permission.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>creationAgeInSeconds</td>
<td>int</td>
<td>If specified, includes only users that were created within a specific time frame.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the user criteria.</td>
</tr>
<tr>
<td>lastChatterActivityAgeInSeconds</td>
<td>int</td>
<td>If specified, includes only members that have posted or commented in the community within a specific time frame.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Name of the user criteria.</td>
</tr>
</tbody>
</table>
| userTypes                  | NetworkUserType (enumeration of type string) | The member type to use in moderation rules. Valid values:  
  - Internal  
  - Customer  
  - Partner |

**Declarative Metadata Sample Definition**

The following is an example of a UserCriteria component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<UserCriteria xmlns="http://soap.sforce.com/2006/04/metadata">
  <masterLabel>Customer and Partner Members</masterLabel>
  <description>Member criteria matches customer and partner member</description>
  <userTypes>Partner</userTypes>
  <userTypes>Customer</userTypes>
</UserCriteria>
```

**UserInterfaceSettings**

Represents the settings that modify the behavior of the org's user interface. This type extends the Metadata metadata type and inherits its fullName field.

In the package manifest, all organization settings metadata types are accessed using the “Settings” name. See Settings for more details.

**File Suffix and Directory Location**

A UserInterfaceSettings component file has the suffix .settings and is stored in the settings directory. The .settings files are different from other named components because there is only one settings file for each settings component.

**Version**

UserInterfaceSettings components are available in API version 46.0 and later.
## Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableHoverDetails</td>
<td>boolean</td>
<td>Indicates whether an interactive overlay containing record details is displayed (true) or not (false). The default is true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> To view hover details for a record, users need the appropriate sharing access and field-level security access for the fields in the mini page layout.</td>
</tr>
<tr>
<td>enableInlineEdit</td>
<td>boolean</td>
<td>Indicates whether users are allowed to edit field values on a record’s detail page (true) or not (false). The default is true.</td>
</tr>
<tr>
<td>enableQuickCreate</td>
<td>boolean</td>
<td>Indicates whether an area displays on a tab home page (corresponds to the Show Quick Create setting), allowing users to create a record quickly with minimal information (true) or not (false). The Quick Create area displays by default on the tab home pages for leads, accounts, contacts, and opportunities. You can control whether the Quick Create area is displayed on all relevant tab home pages.</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of a UserInterfaceSettings component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<UserInterfaceSettings xmlns="http://soap.sforce.com/2006/04/metadata">
    <enableInlineEdit>true</enableInlineEdit>
    <enableHoverDetails>false</enableHoverDetails>
    <enableQuickCreate>true</enableQuickCreate>
</UserInterfaceSettings>
```

### Example Package Manifest

The following is an example package manifest used to deploy or retrieve the user interface settings metadata for an organization:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
        <members>UserInterface</members>
        <name>Settings</name>
    </types>
    <version>46.0</version>
</Package>
```

### Wildcard Support in the Manifest File

The wildcard character * (asterisk) in the package.xml manifest file doesn’t apply to metadata types for feature settings. The wildcard applies only when retrieving all settings, not for an individual setting. For details, see Settings. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.
WaveApplication

Represents the Analytics application. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

WaveApplication components have the suffix .wapp and are stored in the wave folder.

Version

WaveApplication components are available in API version 37.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assetIcon</td>
<td>string</td>
<td>The icon that represents the Analytics application.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description that appears in the user interface.</td>
</tr>
<tr>
<td>folder</td>
<td>string</td>
<td>The internal api name of the folder or application.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>The user interface label name of the folder or application.</td>
</tr>
<tr>
<td>shares</td>
<td>FolderShare</td>
<td>The folder sharing rules.</td>
</tr>
<tr>
<td>templateOrigin</td>
<td>string</td>
<td>The internal (unique) name of the template used to create the application.</td>
</tr>
<tr>
<td>templateVersion</td>
<td>string</td>
<td>The version assigned to the application template by the template's creator.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a WaveApplication component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<WaveApplication xmlns="http://soap.sforce.com/2006/04/metadata">
  <assetIcon>/analytics/wave/web/proto/images/app/icons/11.png</assetIcon>
  <description>Application that shows my sales</description>
  <folder>edit</folder>
  <masterLabel>Sales Application</masterLabel>
  <shares>
    <accessLevel>EditAllContents</accessLevel>
    <sharedTo>shareswith@org.ee</sharedTo>
  </shares>
</WaveApplication>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

WaveDataflow

Represents the WaveDataflow object in the Analytics application. This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.

File Suffix and Directory Location

WaveDataflow components have the suffix .wdf and are stored in the wave folder.

Version

WaveDataflow components are available in API version 37.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The dataflow description that appears in the user interface.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The dataflow name that appears in the user interface.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a WaveDataflow component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <content xsi:nil="true"/>
  <description>flow1</description>
  <masterLabel>flow1</masterLabel>
</WaveDataflow>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

WaveDashboard

Represents the WaveDashboard object in the Analytics application. This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.
File Suffix and Directory Location

WaveDashboard components have the suffix `.wdash` and are stored in the `wave` folder.

Version

WaveDashboard components are available in API version 37.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>string</td>
<td>Required. The internal name of the application.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The dashboard description that appears in the user interface.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The dashboard name that appears in the user interface.</td>
</tr>
<tr>
<td>templateAssetSourceName</td>
<td>string</td>
<td>Links the dashboard to the template used to create it. Null for assets not</td>
</tr>
<tr>
<td></td>
<td></td>
<td>created from a template.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a WaveDashboard component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<WaveDashboard xmlns="http://soap.sforce.com/2006/04/metadata"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <application>dev__app</application>
  <masterLabel>Dashboard1</masterLabel>
  <description>somedesc</description>
</WaveDashboard>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

WaveDataset

Represents the WaveDataset object in the Analytics application. This type extends the Metadata metadata type and inherits its `fullName` field.

File Suffix and Directory Location

WaveDataset components have the suffix `.wds` and are stored in the `wave` folder.
Version

WaveDataset components are available in API version 37.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>string</td>
<td>Required. The internal name of the application.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The dataset description that appears in the user interface.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The user interface label name of the Dataset.</td>
</tr>
<tr>
<td>templateAssetSourceName</td>
<td>string</td>
<td>Links the dataset to the template used to create it. Null for assets not created from a template.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a WaveDataset component.

```xml
<WaveDataset>
  <application>SharedApp</application>
  <description>description</description>
  <masterLabel>dataset1</masterLabel>
</WaveDataset>
```

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

WaveLens

Represents the WaveLens object in the Analytics application. This type extends the MetadataWithContent metadata type and inherits its content and fullName fields.

File Suffix and Directory Location

WaveLens components have the suffix .wlens and are stored in the wave folder.

Version

WaveLens components are available in API version 37.0 and later.
## Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>string</td>
<td>Required. The internal name of the application.</td>
</tr>
<tr>
<td>datasets</td>
<td>string</td>
<td>A reference to the dataset used to create this lens.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The dashboard description that appears in the user interface.</td>
</tr>
<tr>
<td>masterLabel</td>
<td>string</td>
<td>Required. The user interface label name of the dashboard.</td>
</tr>
<tr>
<td>templateAssetSourceName</td>
<td>string</td>
<td>Links the lens to the template used to create it. Null for assets not created from a template.</td>
</tr>
<tr>
<td>visualizationType</td>
<td>string</td>
<td>Required. The visualization type to be used for this lens. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• calheatmap—Calendar heat map</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• comparisontable—Comparison table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• heatmap—Heat map</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• hbar—Horizontal bar</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• hbarhdot—Horizontal dot plot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• matrix—Matrix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• parallelcoords—Parallel coordinates</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• pie—Donut</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• pivottable—Pivot table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• scatter—Scatter plot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• stackhbar—Stacked horizontal bar</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• stackvbar—Stacked vertical bar</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• time—Time line</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• valuestable—Values table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• vbar—Vertical bar</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• vdot—Vertical dot plot</td>
</tr>
</tbody>
</table>

### Declarative Metadata Sample Definition

The following is an example of a WaveLens component.

```xml
<?xml version='1.0' encoding='UTF-8'?>
<WaveLens xmlns='http://soap.sforce.com/2006/04/metadata'
  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'>
  <application>dev__app</application>
  <datasets>dev__abc</datasets>
  <masterLabel>lens1</masterLabel>
  <description>lens in shared app</description>
  <visualizationType>hbar</visualizationType>
</WaveLens>
```
Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

WaveTemplateBundle

Represents an Analytics template bundle, which can be used to create Analytics apps. A bundle contains an Analytics template definition and all its related resources. This type extends the Metadata metadata type and inherits its fullName field.

File Suffix and Directory Location

An Analytics template bundle is a folder that contains definition files for a template. Unlike other metadata components, a WaveTemplateBundle component isn’t represented by a single component file, but instead by a collection of JSON and CSV definition files. Each definition file represents a resource in a template, such as lenses, dashboards, dataflows, and comma-separated values. For example, this directory structure shows the hierarchy of the folders and files for one Analytics Template definition, template1.

```
waveTemplates
    template1
    template-info.json
    variables.json
    ui.json
    extFiles
    PostalCodes.csv
```

Analytics template bundles must be under a top-level folder that’s named waveTemplates. Each bundle must have its own subfolder under the waveTemplates folder and be named with the template’s fully qualified API name. The bundle folder must contain a template-info.json file to specify the metadata about the template and the references to other definition files. An entire bundle doesn’t have a suffix and definition files can have one of the following suffixes.

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Component Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>.json</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>.csv</td>
<td>Comma-Separated Values</td>
</tr>
</tbody>
</table>

Version

WaveTemplateBundle components are available in API version 35.0 and later.

Special Access Rules

Definitions can be created in both managed and unmanaged packages.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assetIcon</td>
<td>string</td>
<td>The icon to use by default for new Analytics apps based on this template. Valid values are 1.png through 20.png.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The specification of the template.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Required. The label of the template.</td>
</tr>
<tr>
<td>templateType</td>
<td>string</td>
<td>Required. The type of the template. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• App</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Dashboard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Lens</td>
</tr>
</tbody>
</table>

Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

WaveXmd

Represents the WaveXmd object in the Analytics application. This type extends the Metadata metadata type and inherits its `fullName` field.

File Suffix and Directory Location

WaveXmd components have the suffix `.xmd` and are stored in the `wave` folder.

Version

WaveXmd components are available in API version 39.0 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataset</td>
<td>string</td>
<td>Specifies the dataset associated with this Xmd.</td>
</tr>
<tr>
<td>datasetConnector</td>
<td>string</td>
<td>The name of the connector source for the dataset.</td>
</tr>
<tr>
<td>datasetFullyQualifiedName</td>
<td>string</td>
<td>Specifies the fully qualified name of the dataset version associated with this Xmd.</td>
</tr>
<tr>
<td>dates</td>
<td>WaveXmdDate</td>
<td>List of dates, with formatting information.</td>
</tr>
<tr>
<td>dimensions</td>
<td>WaveXmdDimension</td>
<td>List of dimensions, with formatting information.</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>measures</td>
<td>WaveXmdMeasure</td>
<td>List of measures, with formatting information.</td>
</tr>
<tr>
<td>organizations</td>
<td>WaveXmdOrganization</td>
<td>List of organizations, for multi-organization support.</td>
</tr>
<tr>
<td>origin</td>
<td>string</td>
<td>The origin of the dataset version.</td>
</tr>
</tbody>
</table>

#### WaveXmdDate

WaveXmdDate represents a date.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>string</td>
<td>Alias of the Date column.</td>
</tr>
<tr>
<td>compact</td>
<td>boolean</td>
<td>Whether the date should be displayed in compact form.</td>
</tr>
<tr>
<td>dateFieldDay</td>
<td>string</td>
<td>The day field.</td>
</tr>
<tr>
<td>dateFieldEpochDay</td>
<td>string</td>
<td>The epoch day field.</td>
</tr>
<tr>
<td>dateFieldEpochSecond</td>
<td>string</td>
<td>The epoch second field.</td>
</tr>
<tr>
<td>dateFieldFiscalMonth</td>
<td>string</td>
<td>The fiscal month field.</td>
</tr>
<tr>
<td>dateFieldFiscalQuarter</td>
<td>string</td>
<td>The fiscal quarter field.</td>
</tr>
<tr>
<td>dateFieldFiscalWeek</td>
<td>string</td>
<td>The fiscal week field.</td>
</tr>
<tr>
<td>dateFieldFiscalYear</td>
<td>string</td>
<td>The fiscal year field.</td>
</tr>
<tr>
<td>dateFieldFullYear</td>
<td>string</td>
<td>The full year field.</td>
</tr>
<tr>
<td>dateFieldHour</td>
<td>string</td>
<td>The hour field.</td>
</tr>
<tr>
<td>dateFieldMinute</td>
<td>string</td>
<td>The minute field.</td>
</tr>
<tr>
<td>dateFieldMonth</td>
<td>string</td>
<td>The month field.</td>
</tr>
<tr>
<td>dateFieldQuarter</td>
<td>string</td>
<td>The quarter field.</td>
</tr>
<tr>
<td>dateFieldSecond</td>
<td>string</td>
<td>The second field.</td>
</tr>
<tr>
<td>dateFieldWeek</td>
<td>string</td>
<td>The week field.</td>
</tr>
<tr>
<td>dateFieldYear</td>
<td>string</td>
<td>The year field.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the date column.</td>
</tr>
<tr>
<td>firstDayOfWeek</td>
<td>int</td>
<td>Represents the first day of the week.</td>
</tr>
<tr>
<td>fiscalMonthOffset</td>
<td>int</td>
<td>Offset number of months for the fiscal year in relation to the calendar year.</td>
</tr>
<tr>
<td>isYearEndFiscalYear</td>
<td>boolean</td>
<td>Whether the year end is the fiscal year.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label of the date column.</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showInExplorer</td>
<td>boolean</td>
<td>Whether or not the date should be shown in the explorer.</td>
</tr>
<tr>
<td>sortIndex</td>
<td>int</td>
<td>Whether or not to sort.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>The type of date. Values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Date — A legacy date type. Available when the time zone is not enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- DateOnly — A date type without an associated time. Available when the time zone is enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- DateTime — A date type that contains both date and time parts. Available when the time zone is enabled.</td>
</tr>
</tbody>
</table>

### WaveXmdDimension

WaveXmdDimension represents a dimension.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customActions</td>
<td>WaveXmdDimensionCustomAction</td>
<td>Custom actions linked to this dimension.</td>
</tr>
<tr>
<td>customActionsEnabled</td>
<td>boolean</td>
<td>Indicates whether the dimension has custom actions enabled.</td>
</tr>
<tr>
<td>dateFormat</td>
<td>string</td>
<td>The format used for a date that is a dimension.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the dimension.</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>The field name of the dimension (used in queries).</td>
</tr>
<tr>
<td>fullyQualifiedName</td>
<td>string</td>
<td>The fully qualified name of the dimension.</td>
</tr>
<tr>
<td>imageTemplate</td>
<td>string</td>
<td>The image template.</td>
</tr>
<tr>
<td>isDerived</td>
<td>boolean</td>
<td>Whether this is a derived dimension.</td>
</tr>
<tr>
<td>isMultiValue</td>
<td>boolean</td>
<td>Indicates whether the dimension is multi-value.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label for the dimension.</td>
</tr>
<tr>
<td>linkTemplate</td>
<td>string</td>
<td>The template for formatting a link.</td>
</tr>
<tr>
<td>linkTemplateEnabled</td>
<td>boolean</td>
<td>Indicates whether the dimension has link templates enabled.</td>
</tr>
<tr>
<td>linkTooltip</td>
<td>string</td>
<td>The tooltip to be displayed for links.</td>
</tr>
<tr>
<td>members</td>
<td>WaveXmdDimensionMember</td>
<td>The member overrides for a dimension.</td>
</tr>
<tr>
<td>origin</td>
<td>string</td>
<td>The origin of this dimension.</td>
</tr>
<tr>
<td>recordDisplayFields</td>
<td>WaveXmdRecordDisplayLookup</td>
<td>Ordered list of dimensions. The list defines the default order in which to display the dimensions in the user interface.</td>
</tr>
<tr>
<td>recordIdField</td>
<td>string</td>
<td>The record ID for this dimension.</td>
</tr>
<tr>
<td>recordOrganizationIdField</td>
<td>string</td>
<td>The record organization ID for this dimension.</td>
</tr>
</tbody>
</table>
### WaveXmdDimension

WaveXmdDimension represents a dimension.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>salesforceActions</td>
<td>WaveXmdDimensionSalesforceAction</td>
<td>Salesforce actions linked to this dimension.</td>
</tr>
<tr>
<td>salesforceActionsEnabled</td>
<td>boolean</td>
<td>Indicates whether the dimension has Salesforce actions enabled.</td>
</tr>
<tr>
<td>showDetailsDefaultFieldIndex</td>
<td>int</td>
<td>Default order in which to show the dimensions in the user interface.</td>
</tr>
<tr>
<td>showInExplorer</td>
<td>boolean</td>
<td>Indicates whether the dimension is displayed in the explorer.</td>
</tr>
<tr>
<td>sortIndex</td>
<td>int</td>
<td>Whether or not to sort.</td>
</tr>
</tbody>
</table>

### WaveXmdDimensionCustomAction

WaveXmdDimensionCustomAction represents a custom action in a dimension.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customActionName</td>
<td>string</td>
<td>The name of this custom action.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Indicates whether the action is enabled for a specific dimension.</td>
</tr>
<tr>
<td>icon</td>
<td>string</td>
<td>The icon for the action.</td>
</tr>
<tr>
<td>method</td>
<td>string</td>
<td>The method for the action.</td>
</tr>
<tr>
<td>sortIndex</td>
<td>int</td>
<td>Whether or not to sort.</td>
</tr>
<tr>
<td>target</td>
<td>string</td>
<td>The target for the action.</td>
</tr>
<tr>
<td>tooltip</td>
<td>string</td>
<td>The tooltip for the action.</td>
</tr>
<tr>
<td>url</td>
<td>string</td>
<td>The URL for the action.</td>
</tr>
</tbody>
</table>

### WaveXmdDimensionMember

WaveXmdDimensionMember represents a dimension.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>string</td>
<td>The color for the member.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label for the member.</td>
</tr>
<tr>
<td>member</td>
<td>string</td>
<td>The member value.</td>
</tr>
<tr>
<td>sortIndex</td>
<td>int</td>
<td>Whether or not to sort.</td>
</tr>
</tbody>
</table>

### WaveXmdRecordDisplayLookup

WaveXmdDimensionRecordDisplayLookup represents a record display field.
**WaveXmdDimensionSalesforceAction**

WaveXmdDimensionSalesforceAction represents an action in a dimension.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Indicates whether the action is enabled for a specific dimension.</td>
</tr>
<tr>
<td>salesforceActionName</td>
<td>string</td>
<td>The name of the action.</td>
</tr>
<tr>
<td>sortIndex</td>
<td>int</td>
<td>Whether or not to sort.</td>
</tr>
</tbody>
</table>

**WaveXmdMeasure**

WaveXmdMeasure represents a measure.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateFormat</td>
<td>string</td>
<td>The format used for a date that is a measure.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the measure.</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>The field name of the measure (used in queries).</td>
</tr>
<tr>
<td>formatCustomFormat</td>
<td>string</td>
<td>The original (XMD 1.1) format array as a single string.</td>
</tr>
<tr>
<td>formatDecimalDigits</td>
<td>int</td>
<td>The number of digits displayed after the decimal place.</td>
</tr>
<tr>
<td>formatIsNegativeParens</td>
<td>boolean</td>
<td>Indicates whether to display negative numbers with parentheses, rather than a minus sign.</td>
</tr>
<tr>
<td>formatPrefix</td>
<td>string</td>
<td>The prefix placed before the field value.</td>
</tr>
<tr>
<td>formatSuffix</td>
<td>string</td>
<td>The suffix placed after the field value.</td>
</tr>
<tr>
<td>formatUnit</td>
<td>string</td>
<td>The unit string for the measure. For example, 'cm'.</td>
</tr>
<tr>
<td>formatUnitMultiplier</td>
<td>double</td>
<td>The multiplier for the unit.</td>
</tr>
<tr>
<td>fullyQualifiedName</td>
<td>string</td>
<td>The fully qualified name of the measure.</td>
</tr>
<tr>
<td>isDerived</td>
<td>boolean</td>
<td>Whether this is a derived measure.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label for the measure.</td>
</tr>
<tr>
<td>origin</td>
<td>string</td>
<td>The origin of the measure.</td>
</tr>
<tr>
<td>showDetailsDefaultFieldIndex</td>
<td>int</td>
<td>Default order in which to show the measures in the user interface.</td>
</tr>
<tr>
<td>showInExplorer</td>
<td>boolean</td>
<td>Indicates whether the measure is displayed in the explorer.</td>
</tr>
</tbody>
</table>
### WaveXmdOrganization

WaveXmdOrganization represents a Salesforce organization.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>instanceUrl</td>
<td>string</td>
<td>The instance URL for the organization.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>The label for the organization.</td>
</tr>
<tr>
<td>organizationIdentifier</td>
<td>string</td>
<td>The ID of the organization.</td>
</tr>
<tr>
<td>sortIndex</td>
<td>int</td>
<td>Whether or not to sort.</td>
</tr>
</tbody>
</table>

### Wildcard Support in the Manifest File

This metadata type supports the wildcard character * (asterisk) in the package.xml manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

### Workflow

Represents the metadata associated with a workflow rule. A workflow rule sets workflow actions into motion when its designated conditions are met. You can configure workflow actions to execute immediately when a record meets the conditions in your workflow rule, or set time triggers that execute the workflow actions on a specific day. Use this metadata type to create, update, or delete workflow rule definitions.

For more information, see “Workflow” in the Salesforce Help. This type extends the Metadata metadata type and inherits its fullName field.

When using a manifest file, retrieve all workflow components using the following code:

```xml
<types>
  <members>*</members>
  <name>Workflow</name>
</types>
```

### Declarative Metadata File Suffix and Directory Location

Workflow files have the suffix .workflow. There is one file per standard or custom object that has workflow. These files are stored in the workflows directory of the corresponding package.

### Version

Workflow rules are available in API version 13.0 and later.
Workflow

This metadata type represents the valid types of workflow rules and actions associated with a standard or custom object.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alerts</td>
<td>WorkflowAlert[]</td>
<td>An array of all alerts for the object associated with the workflow.</td>
</tr>
<tr>
<td>fieldUpdates</td>
<td>WorkflowFieldUpdate[]</td>
<td>An array of all field updates for the object associated with the workflow.</td>
</tr>
<tr>
<td>flowActions</td>
<td>WorkflowFlowAction[]</td>
<td>An array of flow triggers for the object associated with the workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The pilot program for flow trigger workflow actions is closed. If you've</td>
</tr>
<tr>
<td></td>
<td></td>
<td>already enabled the pilot in your org, you can continue to create and edit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>flow trigger workflow actions. If you didn’t enable the pilot in your org,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>use the Flows action in Process Builder instead.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>The developer name used as a unique identifier for API access. The fullName</td>
</tr>
<tr>
<td></td>
<td></td>
<td>can contain only underscores and alphanumeric characters. It must be unique,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>begin with a letter, not include spaces, not end with an underscore, and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>not contain two consecutive underscores. This field is inherited from the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Metadata component.</td>
</tr>
<tr>
<td>knowledgePublishes</td>
<td>WorkflowKnowledgePublish[]</td>
<td>An array of Salesforce Knowledge Workflow Publishes associated with the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>workflow. Available in API version 27.0 and later.</td>
</tr>
<tr>
<td>outboundMessages</td>
<td>WorkflowOutboundMessage[]</td>
<td>An array of all of the outbound messages for the object associated with the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>workflow.</td>
</tr>
<tr>
<td>rules</td>
<td>WorkflowRule[]</td>
<td>An array of all the objects associated with the workflow.</td>
</tr>
<tr>
<td>tasks</td>
<td>WorkflowTask[]</td>
<td>An array of all the tasks for the object associated with the workflow.</td>
</tr>
</tbody>
</table>

WorkflowActionReference

WorkflowActionReference represents one of the workflow actions.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. The name of the workflow action.</td>
</tr>
<tr>
<td>type</td>
<td>WorkflowActionType (enumeration of type string)</td>
<td>Required. Available types of workflow actions:</td>
</tr>
</tbody>
</table>
The pilot program for flow trigger workflow actions is closed. If you've already enabled the pilot in your org, you can continue to create and edit flow trigger workflow actions. If you didn't enable the pilot in your org, use the Flows action in Process Builder instead.

### WorkflowAlert

WorkflowAlert represents an email alert associated with a workflow rule.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ccEmails</td>
<td>string[]</td>
<td>Additional email addresses. This field is similar to the CC field in email clients. For the email to be sent successfully, set a value for ccEmails or recipients. You can set values for both fields. The value of ccEmails can include up to five different email addresses.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The developer name used as a unique identifier for API access. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the Metadata component.</td>
</tr>
<tr>
<td>protected</td>
<td>boolean</td>
<td>Required. Indicates whether this component is protected (true) or not (false). Protected components cannot be linked to or referenced by components created in the installing organization.</td>
</tr>
<tr>
<td>recipients</td>
<td>WorkflowEmailRecipient[]</td>
<td>The recipients for the email. For the email to be sent successfully, set a value for ccEmails or recipients. You can set values for both fields.</td>
</tr>
<tr>
<td>senderAddress</td>
<td>string</td>
<td>The address in the From field for the email alert. This allows you to use a standard global email address for your organization (such as <a href="mailto:support@company.com">support@company.com</a>) instead of the default From field, which is the email address of the person who updates the record. You can only specify a value in this field if the senderType is set to OrgWideEmailAddress. See “Organization-Wide Email Addresses” in the Salesforce Help.</td>
</tr>
</tbody>
</table>
The email used as the sender’s From and Reply-To addresses. The following values are valid:

- **CurrentUser**—The email address of the person updating the record. This is the default setting.
- **DefaultWorkflowUser**—The email address of the default workflow user.
- **OrgWideEmailAddress**—A verified global email address for your organization, such as support@company.com.

---

**WorkflowEmailRecipient**

WorkflowEmailRecipient represents a recipient for an email alert associated with a workflow rule.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>Name of the field referenced in type. The field named should be of the type specified in type.</td>
</tr>
<tr>
<td>recipient</td>
<td>string</td>
<td>The recipients for the email. Depending on the type selected, this may be required.</td>
</tr>
<tr>
<td>type</td>
<td>ActionEmailRecipientTypes (enumeration of type string)</td>
<td>Named reference to an EmailTemplate component. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>accountOwner</td>
<td>The email is sent to the record’s account owner (for example, the Account owner for an Opportunity).</td>
</tr>
<tr>
<td></td>
<td>accountTeam</td>
<td>Only applicable on the Account object. The email is sent to everyone on that Account’s account team.</td>
</tr>
<tr>
<td></td>
<td>campaignMemberDerivedOwner</td>
<td>Emails are sent to lead and contact owners when contacts are added to a campaign or in response to a campaign.</td>
</tr>
<tr>
<td></td>
<td>contactLookup</td>
<td>The email is sent to a contact whose value is looked up from a field on the record. For this value, the field field must reference a Contact.</td>
</tr>
<tr>
<td></td>
<td>creator</td>
<td>The email is sent to the record’s creator.</td>
</tr>
<tr>
<td></td>
<td>customerPortalOwner</td>
<td>The email is sent to a specific self-service portal user. For this value, the recipient</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the field update. This information is useful to track the reasoning for initially configuring the field update.</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>Required. The field (on the object for the workflow) to be updated.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>formula</td>
<td>string</td>
<td>If the <code>operation</code> field value is <code>Formula</code>, this is set to a formula used to compute the new field value.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The developer name used as a unique identifier for API access. The <code>fullName</code> can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the Metadata component.</td>
</tr>
<tr>
<td>literalValue</td>
<td>string</td>
<td>If the <code>operation</code> field value is <code>Literal</code>, this is the literal value for the field.</td>
</tr>
<tr>
<td>lookupValue</td>
<td>string</td>
<td>If the <code>operation</code> field value is <code>lookupValue</code>, this is the lookup value that is referenced.</td>
</tr>
<tr>
<td>lookupValueType</td>
<td>LookupValueType</td>
<td>The type of object that the <code>lookupValue</code> field value is referencing. The valid values are:</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. A name for the component. Available in version API 16.0 and later.</td>
</tr>
<tr>
<td>notifyAssignee</td>
<td>boolean</td>
<td>Required. Notify the assignee when the field is updated.</td>
</tr>
<tr>
<td>operation</td>
<td>FieldUpdateOperation</td>
<td>Required. The operation that computes the value with which to update the field. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>(enumeration of type string)</td>
<td>• Formula - Indicates the field will be set to a formula. If set, the formula must be a valid formula.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Literal - Indicates the field will be set to a literal value. If set, the literalValue must be a valid literal value for this field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LookupValue - Similar to Literal, but for an object reference, such as a contact, user, account, etc. If set, the lookupValue element must be set. Only User is supported in the current API.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NextValue - Indicates that the field will be set to its next value; this is only allowed when the field update references a picklist.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Null - Indicates the field will be set to null.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PreviousValue - Indicates that the field will be set to its previous value; this is only allowed when the field update references a picklist.</td>
</tr>
</tbody>
</table>
### Workflow

**WorkflowFlowAction**

Represents a flow trigger, which is a workflow action that launches a flow. Available in API version 30.0 and later. For more information, see the following topics in the Salesforce Help.

- Define a Flow Trigger for Workflow (Pilot)
- Flow Trigger Considerations (Pilot)

**Note:**

- The pilot program for flow trigger workflow actions is closed. If you've already enabled the pilot in your org, you can continue to create and edit flow trigger workflow actions. If you didn't enable the pilot in your org, use the Flows action in Process Builder instead.
- Test mode for flow triggers isn't supported in the Metadata API. If you want a flow trigger to run the latest flow version when an administrator causes the workflow rule to fire, enable test mode via the user interface after deployment.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>Describes the flow trigger.</td>
</tr>
<tr>
<td>flow</td>
<td>string</td>
<td>Required. API name of the flow that this workflow action launches.</td>
</tr>
<tr>
<td>flowInputs</td>
<td>WorkflowFlowActionParameter[]</td>
<td>An array of values to pass into flow variables when launching the flow.</td>
</tr>
</tbody>
</table>
WorkflowFlowActionParameter

Represents a value specified in the flow trigger that is passed into a variable when launching the flow.

**Note:** The pilot program for flow trigger workflow actions is closed. If you’ve already enabled the pilot in your org, you can continue to create and edit flow trigger workflow actions. If you didn’t enable the pilot in your org, use the Flows action in Process Builder instead.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Required. API name of the flow variable. The flow variable must have isInput set to True.</td>
</tr>
</tbody>
</table>
| value      | string     | Required. Value to assign to the flow variable when launching the flow. If the variable’s data type is sObject, value must be a merge field that identifies a record—or a lookup relationship field that references a record—of the same object type as the variable. For example:
  - {!this}—identifies the record that fired the workflow rule.
  - {!Contact}—identifies the contact associated with the record that fired the workflow rule.
  - {!Asset.Account}—identifies the account associated with the asset that is associated with the record that fired the workflow rule.
  - {!SomeObject__r}—uses a lookup relationship field to identify a custom object record associated with the record that fired the workflow rule.
For variables of other data types, you can enter a merge field or a literal value. Manually enter a literal value when the variable should have the same value every time the flow is launched, regardless of which record fired the workflow rule. For example, you can enter true or false for a variable of type Boolean.
Supported merge fields identify a global variable or a field of the same data type as the flow variable. For example:
  - {!Id}—ID of the record that fired the workflow rule.
  - {!Account.Owner.Email}—email address of the account owner for the account associated with the record that fired the workflow rule.
  - {$Organization.Country}—country of the organization.

WorkflowKnowledgePublish

WorkflowKnowledgePublish represents Salesforce Knowledge article publishing actions and information. Available in API version 27.0 and later.
### Field Name | Field Type | Description
--- | --- | ---
action | KnowledgeWorkflowAction (enumeration of type string) | The article publishing actions available when this rule fires. Valid values are:
• PublishAsNew: Publishes the article as a new article.
• Publish: Publishes the article as a version of a previously published article.
description | string | A brief article description.
label | string | Label that represents the article throughout the Salesforce user interface.
language | string | The language of the article.
protected | boolean | Required. Indicates whether this component is protected (true) or not (false). Protected components cannot be linked to or referenced by components created in the installing organization.

---

### WorkflowOutboundMessage

WorkflowOutboundMessage represents an outbound message associated with a workflow rule. Outbound messages are workflow and approval actions that send the information you specify to an endpoint you designate, such as an external service. An outbound message sends the data in the specified fields in the form of a SOAP message to the endpoint. For more information, see “Outbound Message Actions” in the Salesforce Help.

| Field Name | Field Type | Description |
--- | --- | ---
apiVersion | double | Required. The API version of the outbound message. This is automatically set to the current API version when the outbound message is created. Valid API versions for outbound messages are 8.0 and 18.0 or later. This API version is used in API calls back to Salesforce using the enterprise or partner WSDLs. The API Version can only be modified by using Metadata API. It can’t be modified using the Salesforce user interface. This field is available in API version 18.0 and later.
| | | **Warning:** If you change the apiVersion to a version that doesn’t support one of the fields configured for the outbound message, messages will fail until you update your outbound message listener to consume the updated WSDL. You can monitor the status of outbound messages from Setup by entering Outbound Messages in the Quick Find box, then selecting Outbound Messages in Salesforce.
description | string | Describes the outbound message.
endpointUrl | string | Required. The endpoint URL to which the outbound message is sent.
## Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>string[]</td>
<td>The named references to the fields that are to be sent.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The developer name used as a unique identifier for API access.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The fullName can contain only underscores and alphanumeric characters. It</td>
</tr>
<tr>
<td></td>
<td></td>
<td>must be unique, begin with a letter, not include spaces, not end with an</td>
</tr>
<tr>
<td></td>
<td></td>
<td>underscore, and not contain two consecutive underscores. This field is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>inherited from the Metadata component.</td>
</tr>
<tr>
<td>includeSessionId</td>
<td>boolean</td>
<td>Required. Set if you want the Salesforce session ID included in the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>outbound message. Useful if you intend to make API calls and you do not</td>
</tr>
<tr>
<td></td>
<td></td>
<td>want to include a username and password.</td>
</tr>
<tr>
<td>integrationUser</td>
<td>string</td>
<td>Required. The named reference to the user under which this message is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sent.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Required. A name for the component. Available in version API 16.0 and later.</td>
</tr>
<tr>
<td>protected</td>
<td>boolean</td>
<td>Required. Indicates whether this component is protected (true) or not</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(false). Protected components cannot be linked to or referenced by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>components created in the installing organization.</td>
</tr>
<tr>
<td>useDeadLetterQueue</td>
<td>boolean</td>
<td>This field is only available for organizations with dead letter queue</td>
</tr>
<tr>
<td></td>
<td></td>
<td>permissions turned on. If set, this outbound message will use the dead</td>
</tr>
<tr>
<td></td>
<td></td>
<td>letter queue if normal delivery fails.</td>
</tr>
</tbody>
</table>

### WorkflowRule

This metadata type represents a workflow rule. This type extends the Metadata metadata type and inherits its fullName field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actions</td>
<td>WorkflowActionReference[]</td>
<td>An array of references for the actions that should happen when this rule</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fires.</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>Required. Determines if this rule is active.</td>
</tr>
<tr>
<td>booleanFilter</td>
<td>string</td>
<td>For advanced criteria filter, the boolean formula, for example, (1 AND 2) OR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.</td>
</tr>
<tr>
<td>criteriaItems</td>
<td>FilterItem[]</td>
<td>An array of the boolean criteria (conditions) under which this rule fires.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note that either this or formula must be set.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the workflow rule</td>
</tr>
<tr>
<td>formula</td>
<td>string</td>
<td>The formula condition under which this rule first (either this or criteriaItems) must be set</td>
</tr>
</tbody>
</table>
### Metadata Types

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fullName</td>
<td>string</td>
<td>The developer name used as a unique identifier for API access. The <code>fullName</code> can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the Metadata component.</td>
</tr>
</tbody>
</table>

| triggerType | WorkflowTriggerTypes (enumeration of type string) | Under what conditions the trigger fires. Valid values are:  
- `onAllChanges` - The workflow rule is considered on all changes.  
- `onCreateOnly` - The workflow rule is considered only on create.  
- `onCreateOrTriggeringUpdate` - The workflow rule is considered on create and triggering updates. |

| workflowTimeTriggers | WorkflowTimeTrigger | Represents a set of Workflow actions (Field Updates, Email Alerts, Outbound Messages, and Tasks) that should execute before or after a specified interval of time. |

### WorkflowTask

This metadata type references an assigned workflow task.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignedTo</td>
<td>string</td>
<td>Specifies the user, role, or team to which the workflow rule or action is assigned. The field corresponding to the value specified here must be the same as the specified <code>assignedToType</code>.</td>
</tr>
</tbody>
</table>

| assignedToType | ActionTaskAssignedToTypes (enumeration of type string) | Valid string values for this type are:  
- `accountCreator` - When set, the task is assigned to the record’s account’s creator.  
- `accountOwner` - When set, the task is assigned to the record’s account’s owner (Opportunity).  
- `accountTeam` - Same as WorkflowAlert type  
- `creator` - When set, the task is assigned to the record’s creator.  
- `opportunityTeam` - Same as WorkflowAlert type |
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>- When set, the task is assigned to the record’s owner.</td>
<td></td>
</tr>
<tr>
<td>partnerUser</td>
<td>- When set, the assignedTo field references a User (by username), a partner user.</td>
<td></td>
</tr>
<tr>
<td>portalRole</td>
<td>- When set, the assignedTo field references a Role (by role name), a portal role.</td>
<td></td>
</tr>
<tr>
<td>role</td>
<td>- When set, the assignedTo field references a Role (by role name)</td>
<td></td>
</tr>
<tr>
<td>user</td>
<td>- When set, the assignedTo field references a User (by username)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>description</th>
<th>string</th>
<th>The description of this workflow task.</th>
</tr>
</thead>
<tbody>
<tr>
<td>dueDateOffset</td>
<td>int</td>
<td>Required. The offset, in days, from either the trigger date, or the date specified in the (optional) offsetFromField. This can be a negative number.</td>
</tr>
<tr>
<td>fullName</td>
<td>string</td>
<td>Required. The developer name used as a unique identifier for API access. The fullName can contain only underscores and alphanumeric characters. It must be unique, begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is inherited from the Metadata component.</td>
</tr>
<tr>
<td>notifyAssignee</td>
<td>boolean</td>
<td>Required. Set to send an email notification when the task is assigned.</td>
</tr>
<tr>
<td>offsetFromField</td>
<td>string</td>
<td>Optional field reference of the date field from which the dueDate should be computed.</td>
</tr>
<tr>
<td>priority</td>
<td>string</td>
<td>Required. The priority to assign the created task.</td>
</tr>
<tr>
<td>protected</td>
<td>boolean</td>
<td>Required. Indicates whether this component is protected (true) or not (false). Protected components cannot be linked to or referenced by components created in the installing organization.</td>
</tr>
<tr>
<td>status</td>
<td>string</td>
<td>Required. The status to assign the created task.</td>
</tr>
<tr>
<td>subject</td>
<td>string</td>
<td>Required. A subject for the workflow task. It is used if an email notification is sent when the task is assigned. Available in API version 16.0 and later.</td>
</tr>
</tbody>
</table>

**WorkflowTimeTrigger**

Represents a set of Workflow actions (Field Updates, Email Alerts, Outbound Messages, and Tasks) that should execute before or after a specified interval of time.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actions</td>
<td>WorkflowActionReference[]</td>
<td>An array of references for the actions that should happen when this trigger fires.</td>
</tr>
<tr>
<td>offsetFromField</td>
<td>string</td>
<td>The date type field name that the time-based workflow triggers off of, i.e. Created Date, Last Modified Date, Rule Trigger Date or a custom date field on the object for which the workflow rule is defined.</td>
</tr>
<tr>
<td>timeLength</td>
<td>string</td>
<td>The numeric value of the time after/before the workflow triggers. A negative value represents the time length before the trigger will fire. The timeLength is measured in either hours or days, as specified by workflowTimeTriggerUnit.</td>
</tr>
<tr>
<td>workflowTimeTriggerUnit</td>
<td>WorkflowTimeUnits  (enumeration of type string)</td>
<td>The unit of time before or after which the time-based workflow will trigger. Valid string values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Days</td>
</tr>
</tbody>
</table>

**Declarative Metadata Sample Definition**

The following is the definition of a workflow rule:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Workflow xmlns="http://soap.sforce.com/2006/04/metadata">
  <alerts>
    <fullName>Another_alert</fullName>
    <description>Another alert</description>
    <protected>false</protected>
    <recipients>
      <type>accountOwner</type>
    </recipients>
    <recipients>
      <field>Contact__c</field>
      <type>contactLookup</type>
    </recipients>
    <recipients>
      <field>Email__c</field>
      <type>email</type>
    </recipients>
    <template>TestEmail/Email Test</template>
  </alerts>
  <fieldUpdates>
    <fullName>Enum_Field_Update</fullName>
    <description>Blah</description>
    <field>EnumField__c</field>
    <name>Enum Field Update</name>
    <notifyAssignee>true</notifyAssignee>
    <operation>NextValue</operation>
    <protected>false</protected>
  </fieldUpdates>
</Workflow>
```
<fieldUpdates>
  <fullName>Enum_Field_Update2</fullName>
  <description>Blah</description>
  <field>EnumField__c</field>
  <literalValue>PLX2</literalValue>
  <name>Enum Field Update2</name>
  <notifyAssignee>true</notifyAssignee>
  <operation>Literal</operation>
  <protected>false</protected>
</fieldUpdates>

<fieldUpdates>
  <fullName>Field_Update</fullName>
  <description>TestField update desc</description>
  <field>Name</field>
  <formula>Name &quot;Updated&quot;</formula>
  <name>Field Update</name>
  <notifyAssignee>false</notifyAssignee>
  <operation>Formula</operation>
  <protected>false</protected>
</fieldUpdates>

<fieldUpdates>
  <fullName>Lookup_On_Contact</fullName>
  <field>RealOwner__c</field>
  <lookupValue>admin@acme.com</lookupValue>
  <name>Lookup On Contact</name>
  <notifyAssignee>false</notifyAssignee>
  <operation>LookupValue</operation>
  <protected>false</protected>
</fieldUpdates>

<outboundMessages>
  <fullName>Another_Outbound_message</fullName>
  <description>Another Random outbound.</description>
  <endpointUrl>http://www.test.com</endpointUrl>
  <fields>Email__c</fields>
  <fields>Id</fields>
  <fields>Name</fields>
  <includeSessionId>true</includeSessionId>
  <integrationUser>admin@acme.com</integrationUser>
  <name>Another Outbound message</name>
  <protected>false</protected>
</outboundMessages>

<rules>
  <fullName>BooleanFilter</fullName>
  <active>false</active>
  <booleanFilter>1 AND 2 OR 3</booleanFilter>
  <criteriaItems>
    <field>CustomObjectForWorkflow__c.CreatedById</field>
    <operation>notEqual</operation>
  </criteriaItems>
  <criteriaItems>
    <field>CustomObjectForWorkflow__c.CreatedById</field>
    <operation>notEqual</operation>
    <value>abc</value>
  </criteriaItems>
</rules>
<criteriaItems>
  <field>CustomObjectForWorkflow__c.CreatedById</field>
  <operation>equals</operation>
  <value>xyz</value>
</criteriaItems>
<triggerType>onCreateOrTriggeringUpdate</triggerType>
</rules>

<rules>
  <fullName>Custom Rule1</fullName>
  <actions>
    <name>Another_alert</name>
    <type>Alert</type>
  </actions>
  <actions>
    <name>Enum_Field_Update2</name>
    <type>FieldUpdate</type>
  </actions>
  <actions>
    <fullName>Field_Update</fullName>
    <type>FieldUpdate</type>
  </actions>
  <actions>
    <name>Another_Outbound_message</name>
    <type>OutboundMessage</type>
  </actions>
  <actions>
    <name>Role_task_was_completed</name>
    <type>Task</type>
  </actions>
  <active>true</active>
  <criteriaItems>
    <field>CustomObjectForWorkflow__c.Name</field>
    <operation>startsWith</operation>
    <value>ABC</value>
  </criteriaItems>
  <description>Custom Rule1 desc</description>
  <triggerType>onCreateOrTriggeringUpdate</triggerType>
</rules>

<rules>
  <fullName>IsChangedFunctionRule</fullName>
  <active>true</active>
  <description>IsChangedDesc</description>
  <formula>ISCHANGED(Name)</formula>
  <triggerType>onAllChanges</triggerType>
</rules>

<tasks>
  <fullName>Another_task_was_completed</fullName>
  <assignedToType>owner</assignedToType>
  <description>Random Comment</description>
  <dueDateOffset>20</dueDateOffset>
  <notifyAssignee>true</notifyAssignee>
  <priority>High</priority>
  <protected>false</protected>
  <status>Completed</status>
</tasks>
<subject>Another task was completed</subject>
</tasks>
<tasks>
  <fullName>Role_task_was_completed</fullName>
  <assignedTo>R11</assignedTo>
  <assignedToType>role</assignedToType>
  <dueDateOffset>-2</dueDateOffset>
  <notifyAssignee>true</notifyAssignee>
  <offsetFromField>CustomObjectForWorkflow__c.CreatedDate</offsetFromField>
  <priority>High</priority>
  <protected>false</protected>
  <status>Completed</status>
  <subject>Role task was completed</subject>
</tasks>
<tasks>
  <fullName>User_task_was_completed</fullName>
  <assignedTo>admin@acme.com</assignedTo>
  <assignedToType>user</assignedToType>
  <dueDateOffset>-2</dueDateOffset>
  <notifyAssignee>true</notifyAssignee>
  <offsetFromField>User.CreatedDate</offsetFromField>
  <priority>High</priority>
  <protected>false</protected>
  <status>Completed</status>
  <subject>User task was completed</subject>
</tasks>
</Workflow>

**Wildcard Support in the Manifest File**

This metadata type supports the wildcard character * (asterisk) in the `package.xml` manifest file. For information about using the manifest file, see Deploying and Retrieving Metadata with the Zip File.

**WorkSkillRouting**

Represents a setup object that stores a set of `WorkSkillRoutingAttribute` objects used to route a work item to an agent who has the skills necessary to take the work.

**File Suffix and Directory Location**

`WorkSkillRouting` components have the suffix `workSkillRouting` and are stored in the `workSkillRoutings` folder.

**Version**

`WorkSkillRouting` components are available in API version 45 and later.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>boolean</td>
<td>Indicates whether assignment rules are active and can be evaluated.</td>
</tr>
<tr>
<td>attributes</td>
<td>WorkSkillRoutingAttribute[]</td>
<td>List of routing attribute assignments.</td>
</tr>
<tr>
<td>sobjectType</td>
<td>string</td>
<td>Type of Salesforce object that the attributes are associated with.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a WorkSkillRouting component.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<WorkSkillRouting xmlns="http://soap.sforce.com/2006/04/metadata">
  <active>true</active>
  <label>WorkSkillRouting0016 test</label>
  <sobjectType>Case</sobjectType>
</WorkSkillRouting>
```

WorkSkillRoutingAttribute

Represents a setup object that stores metadata related to the routing assignments between object attributes and skills. Attributes are used to route a work item to an agent who has the skills necessary to take the work.

Version

WorkSkillRoutingAttribute components are available in API version 45 and later.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>string</td>
<td>Field that this attribute applies to.</td>
</tr>
<tr>
<td>skill</td>
<td>string</td>
<td>Skill used to route the work item when the attribute maps to the value selected.</td>
</tr>
<tr>
<td>skillLevel</td>
<td>int</td>
<td>Level of the skill required. This value can range from 0 to 10.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Attribute value that is assigned to the selected skill.</td>
</tr>
</tbody>
</table>

Declarative Metadata Sample Definition

The following is an example of a WorkSkillRouting component that contains attributes.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<WorkSkillRouting xmlns="http://soap.sforce.com/2006/04/metadata">
```
The following is an example `package.xml` that references the previous definition.
CHAPTER 12  Headers

Use headers in Metadata API calls to set options for each call.

IN THIS SECTION:

**AllOrNoneHeader**
Indicates whether to roll back all metadata changes when some of the records in a call result in failures.

**CallOptions**
Specifies the API client identifier.

**DebuggingHeader**
Specifies that the deployment result will contain the debug log output, and specifies the level of detail included in the log. The debug log contains the output of Apex tests that are executed as part of a deployment.

**SessionHeader**
Specifies the session ID that the login call returns. This session ID is used to authenticate all subsequent Metadata API calls.

---

**AllOrNoneHeader**

Indicates whether to roll back all metadata changes when some of the records in a call result in failures.

---

**Version**

This header is available in API version 34.0 and later.

---

**Supported Calls**

`createMetadata(), updateMetadata(), upsertMetadata(), deleteMetadata()`

---

**Usage**

If this header isn’t used in API version 34.0 and later, by default a call can save a partial set of records (equivalent to `AllOrNoneHeader=false`)—the records that are processed successfully are saved and records that have failures aren’t saved.
Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allOrNone</td>
<td>boolean</td>
<td>Set to <code>true</code> to cause all metadata changes to be rolled back if any records in the call cause failures. Set to <code>false</code> to enable saving only the records that are processed successfully when other records in the call cause failures.</td>
</tr>
</tbody>
</table>

Sample Code—Java

Add the `AllOrNoneHeader` to the metadata connection before you perform a call as follows:

```java
metadataConnection.setAllOrNoneHeader(true);
```

This next example shows how to use the `AllOrNoneHeader` when creating two custom objects. Because the second custom object doesn't have the required `Name` field, the `create()` call can't create this custom object and rolls back the first custom object. The output is shown after this code sample.

```java
import com.sforce.soap.metadata.*;
import com.sforce.soap.metadata.Error;
import com.sforce.ws.ConnectionException;

public class CallWithHeader {

    MetadataConnection metadataConnection = null;

    public static void main(String[] args) throws ConnectionException {
        CallWithHeader samples = new CallWithHeader();
        samples.createWithHeader();
    }

    public CallWithHeader() throws ConnectionException {
        metadataConnection = MetadataLoginUtil.login();
    }

    public void createWithHeader() throws ConnectionException {
        // Define two custom objects to be inserted.
        CustomObject col = new CustomObject();
        String namel = "MyCustomObject1";
        col.setFullName(namel + "__c");
        col.setDeploymentStatus(DeploymentStatus.Deployed);
        col.setDescription("Created by the Metadata API");
        col.setEnableActivities(true);
        col.setLabel(namel + " Object");
        col.setPluralLabel(col.getLabel() + "s");
        col.setSharingModel(SharingModel.ReadWrite);

        CustomField nf = new CustomField();
        nf.setType(FieldType.Text);
        nf.setLabel(col.getFullName() + " Name");
        col.setNameField(nf);
    }
}
```
// The second custom object doesn't have a Name field
CustomObject co2 = new CustomObject();
String name2 = "MyCustomObject2";
co2.setFullName(name2 + "__c");
co2.setDeploymentStatus(DeploymentStatus.Deployed);
co2.setDescription("Created by the Metadata API");
co2.setEnableActivities(true);
co2.setLabel(name2 + " Object");
co2.setPluralLabel(co2.getLabel() + "s");
co2.setSharingModel(SharingModel.ReadWrite);

// Setting the allOrNone header to true to cause
// the call to not commit any record if one or more
// records in this call have failures.
metadataConnection.setAllOrNoneHeader(true);

// Now that the header has been set, make the create call.
SaveResult[] results = metadataConnection
   .createMetadata(new Metadata[] { co1, co2 });

// Iterate through the call results
for (SaveResult r : results) {
   if (r.isSuccess()) {
      System.out.println("Created component: " + r.getFullName());
   } else {
      System.out.println("Errors were encountered while creating "
         + r.getFullName());
      for (Error e : r.getErrors()) {
         System.out.println("Error message: " + e.getMessage());
         System.out.println("Status code: " + e.getStatusCode());
      }
   }
}

This is the output that the sample returns. The first record is rolled back and the second has a failure.

Errors were encountered while creating MyCustomObject1__c
Error message: Record rolled back because not all records were valid and the request was using AllOrNone header
Status code: ALL_OR_NONE_OPERATION_ROLLED_BACK
Errors were encountered while creating MyCustomObject2__c
Error message: Must specify a nameField of type Text or AutoNumber
Status code: FIELD_INTEGRITY_EXCEPTION

**CallOptions**

Specifies the API client identifier.
Version
This call is available in all API versions.

Supported Calls
All Metadata API calls.

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>client</td>
<td>string</td>
<td>A value that identifies an API client.</td>
</tr>
</tbody>
</table>

Sample Code—Java
To change the API client ID, add the CallOptions header to the metadata connection before you perform a call as follows:

```java
metadataConnection.setCallOptions("client ID");
```

DebuggingHeader
Specifies that the deployment result will contain the debug log output, and specifies the level of detail included in the log. The debug log contains the output of Apex tests that are executed as part of a deployment.

Version
This header is available in all API versions.

Supported Calls
deploy()

Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>categories</td>
<td>LogInfo[]</td>
<td>A list of log categories with their associated log levels.</td>
</tr>
<tr>
<td>debugLevel</td>
<td>LogType (enumeration of type string)</td>
<td>Deprecated. This field is provided only for backward compatibility. If you provide values for both <code>debugLevel</code> and <code>categories</code>, the <code>categories</code> value is used. The <code>debugLevel</code> field specifies the type of information returned in the debug log. The values are listed from the least amount of</td>
</tr>
</tbody>
</table>

945
### Field Name | Type | Description
--- | --- | ---
| LogInfo | Specifies the type and amount of information to be returned in the debug log. The categories field takes a list of these objects. LogInfo is a mapping of category to level.

#### LogInfo

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>category</td>
<td>LogCategory</td>
<td>Specify the type of information returned in the debug log. Valid values are:</td>
</tr>
<tr>
<td>level</td>
<td>LogCategoryLevel</td>
<td>Specifies the level of detail returned in the debug log. Valid log levels are (listed from lowest to highest):</td>
</tr>
</tbody>
</table>
Sample Code—Java

Add the `DebuggingHeader` to the metadata connection before you perform the `deploy()` call as follows.

```java
LogInfo[] logs = new LogInfo[1];
logs[0] = new LogInfo();
logs[0].setCategory(LogCategory.Apex_code);
logs[0].setLevel(LogCategoryLevel.Fine);
metadataConnection.setDebuggingHeader(logs);
```

The result of the `deploy()` call is obtained by calling `checkDeployStatus()`. After the deployment finishes, and if tests were run, the response of `checkDeployStatus()` contains the debug log output in the `debugLog` field of a `DebuggingInfo` output header.

---

### SessionHeader

Specifies the session ID that the login call returns. This session ID is used to authenticate all subsequent Metadata API calls.

#### Version

This header is available in all API versions.

#### Supported Calls

All Metadata API calls.

#### Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sessionId</td>
<td>string</td>
<td>The session ID that the login call returns.</td>
</tr>
</tbody>
</table>

---

Sample Code—Java

Add the `SessionHeader` to the metadata connection before you perform a call as follows:

```java
metadataConnection.setSessionHeader("<session_ID>");
```
APPENDIX A  CustomObjectTranslation Language Support: Fully Supported Languages

Not every language supports all the possible values for the fields in CustomObjectTranslation. Use this appendix to determine which field values a language supports.

Note: Salesforce offers three levels of language support: fully supported languages, end-user languages, and platform-only languages. This appendix provides information only for fully supported languages. For more information, see Supported Languages.

Chinese (Simplified)

<table>
<thead>
<tr>
<th>plural</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
</tr>
</tbody>
</table>

Chinese (Traditional)

<table>
<thead>
<tr>
<th>plural</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
</tr>
</tbody>
</table>

Danish

<table>
<thead>
<tr>
<th>article</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
<tr>
<td>Definite</td>
</tr>
<tr>
<td>Indefinite</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feminine</td>
</tr>
<tr>
<td>Neuter</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>plural</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
</tr>
<tr>
<td>false</td>
</tr>
</tbody>
</table>

Dutch

<table>
<thead>
<tr>
<th>gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feminine</td>
</tr>
</tbody>
</table>

948
Neuter

**plural**
true
false

### Finnish

caseType
Ablative
Adessive
Allative
Elative
Essive
Genitive
Illative
Inessive
Nominative
Partitive
Translative

**plural**
true
false

**possessive**
None
First
Second

### French

gender
Masculine
Feminine

**startsWith**
Consonant
Vowel

**plural**
true
false
German

caseType
- Accusative
- Dative
- Genitive
- Nominative

gender
- Masculine
- Feminine
- Neuter

plural
- true
- false

Italian

gender
- Masculine
- Feminine

startsWith
- Consonant
- Special
- Vowel

plural
- true
- false

Japanese

plural
- false

Korean

plural
- false
Norwegian

- **article**
  - Definite
  - Indefinite
  - None
- **gender**
  - Masculine
  - Feminine
  - Neuter
- **plural**
  - true
  - false

Portuguese (Brazil)

- **gender**
  - Masculine
  - Feminine
- **plural**
  - true
  - false

Russian

- **caseType**
  - Accusative
  - Dative
  - Genitive
  - Instrumental
  - Nominative
  - Prepositional
- **gender**
  - Masculine
  - Feminine
  - Neuter
  - AnimateMasculine
- **plural**
  - true
**Spanish**

<table>
<thead>
<tr>
<th>gender</th>
<th>Masculine</th>
<th>Feminine</th>
</tr>
</thead>
<tbody>
<tr>
<td>plural</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

**Spanish (Mexico)**

<table>
<thead>
<tr>
<th>gender</th>
<th>Masculine</th>
<th>Feminine</th>
</tr>
</thead>
<tbody>
<tr>
<td>plural</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

**Swedish**

<table>
<thead>
<tr>
<th>article</th>
<th>None</th>
<th>Definite</th>
<th>Indefinite</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td>Feminine</td>
<td>Neuter</td>
<td></td>
</tr>
<tr>
<td>plural</td>
<td>true</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

**Thai**

<table>
<thead>
<tr>
<th>plural</th>
<th>false</th>
</tr>
</thead>
</table>
APPENDIX B  CustomObjectTranslation Language Support: End-User Languages

Not every language supports all the possible values for the fields in CustomObjectTranslation. Use this appendix to determine which field values a language supports.

Note: Salesforce offers three levels of language support: fully supported languages, end-user languages, and platform-only languages. This appendix provides information only for end-user languages. For more information, see Supported Languages.

Arabic

<table>
<thead>
<tr>
<th>field</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>article</td>
<td>Definite</td>
</tr>
<tr>
<td></td>
<td>None</td>
</tr>
<tr>
<td>gender</td>
<td>Masculine</td>
</tr>
<tr>
<td></td>
<td>Feminine</td>
</tr>
<tr>
<td>plural</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>false</td>
</tr>
<tr>
<td>possessive</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>First</td>
</tr>
<tr>
<td></td>
<td>Second</td>
</tr>
</tbody>
</table>

Bulgarian

<table>
<thead>
<tr>
<th>field</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td>Masculine</td>
</tr>
<tr>
<td></td>
<td>Feminine</td>
</tr>
<tr>
<td></td>
<td>Neuter</td>
</tr>
<tr>
<td>plural</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>false</td>
</tr>
</tbody>
</table>
Croatian

caseType
  Accusative
  Dative
  Genitive
  Instrumental
  Locative
  Nominative
gender
  Feminine
  Masculine
  Neuter
plural
  true
  false

Czech

caseType
  Accusative
  Dative
  Genitive
  Instrumental
  Locative
  Nominative
gender
  Masculine
  Feminine
  Neuter
  AnimateMasculine
plural
  true
  false

English (UK)

plural
  false
<table>
<thead>
<tr>
<th>Language</th>
<th>caseType</th>
<th>gender</th>
<th>plural</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greek</td>
<td>caseType</td>
<td>gender</td>
<td>plural</td>
</tr>
<tr>
<td>Hebrew</td>
<td>article</td>
<td>gender</td>
<td>plural</td>
</tr>
<tr>
<td>Hungarian</td>
<td>caseType</td>
<td>gender</td>
<td>plural</td>
</tr>
</tbody>
</table>
Distributive
Elative
Essiveformal
Illative
Inessive
Instrumental
Nomminative
Sublative
Terminative
Translative
Superessive

plural
true
false

possessive
None
First
Second

startsWith
Consonant
Vowel

### Indonesian

plural
false
true

### Polish

caseType
Nomminative
Accusative
Dative
Genitive
Instrumental
Locative
gender
Masculine

956
Feminine
Neuter
AnimateMasculine

plural
ture
false

Portuguese (Portugal)

gender
Feminine
Masculine

plural
ture
false

Romanian

article
Definite
None

gender
Masculine
Feminine
Neuter

plural
ture
false

Slovak

caseType
Accusative
Dative
Genitive
Instrumental
Nominative
Locative

gender
Feminine
Slovenian

caseType
   Accusative
   Dative
   Genitive
   Instrumental
   Nominative
   Locative

gender
   Feminine
   Masculine
   Neuter
   AnimateMasculine

plural
   true
   false

Turkish

caseType
   Ablative
   Accusative
   Dative
   Genitive
   Nominative
   Locative

possessive
   None
   First
   Second
**plural**
- true
- false

**Ukrainian**

**caseType**
- Accusative
- Dative
- Genitive
- Instrumental
- Nominative
- Locative

**gender**
- Masculine
- Feminine
- Neuter
- AnimateMasculine

**plural**
- true
- false

**Vietnamese**

**plural**
- false
APPENDIX C  StandardValueSet Names and Standard Picklist Fields

In API version 38.0 and later, standard picklists are represented by the StandardValueSet type. In previous versions, standard picklists are represented by the CustomField type. This table lists the names of standard picklists as standard value sets and their corresponding field names.

Note: The names of standard value sets and picklist fields are case-sensitive.

<table>
<thead>
<tr>
<th>Standard Value Set Name (API version 38.0 and later)</th>
<th>Field Name (API version 37.0 and earlier)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccountContactMultiRoles</td>
<td>AccountContactRelation.Roles</td>
</tr>
<tr>
<td>AccountContactRole</td>
<td>AccountContactRole.Role</td>
</tr>
<tr>
<td>AccountOwnership</td>
<td>Account.Ownership</td>
</tr>
<tr>
<td>AccountRating</td>
<td>Account.Rating</td>
</tr>
<tr>
<td></td>
<td>Lead.Rating</td>
</tr>
<tr>
<td>AccountType</td>
<td>Account.Type</td>
</tr>
<tr>
<td>AssetStatus</td>
<td>Asset.Status</td>
</tr>
<tr>
<td>CampaignMemberStatus</td>
<td>Campaign.Member.Status</td>
</tr>
<tr>
<td>CampaignStatus</td>
<td>Campaign.Status</td>
</tr>
<tr>
<td>CampaignType</td>
<td>Campaign.Type</td>
</tr>
<tr>
<td>CaseContactRole</td>
<td>CaseContactRole.Role</td>
</tr>
<tr>
<td>CaseOrigin</td>
<td>Case.Origin</td>
</tr>
<tr>
<td>CasePriority</td>
<td>Case.Priority</td>
</tr>
<tr>
<td>CaseReason</td>
<td>Case.Reason</td>
</tr>
<tr>
<td>CaseStatus</td>
<td>Case.Status</td>
</tr>
<tr>
<td>CaseType</td>
<td>Case.Type</td>
</tr>
<tr>
<td>ContactRole</td>
<td>OpportunityContactRole.Role</td>
</tr>
<tr>
<td>ContractContactRole</td>
<td>ContractContactRole.Role</td>
</tr>
<tr>
<td>ContractStatus</td>
<td>Contract.Status</td>
</tr>
</tbody>
</table>
### Standard Value Set Names and Standard Picklist Fields

<table>
<thead>
<tr>
<th>Standard Value Set Name (API version 38.0 and later)</th>
<th>Field Name (API version 37.0 and earlier)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entitlement.Type</td>
<td>Entitlement.Type</td>
</tr>
<tr>
<td>EventSubject</td>
<td>Event.Subject</td>
</tr>
<tr>
<td>EventType</td>
<td>Event.Type</td>
</tr>
<tr>
<td>FiscalYearPeriodName</td>
<td>Period.PeriodLabel</td>
</tr>
<tr>
<td>FiscalYearPeriodPrefix</td>
<td>FiscalYearSettings.PeriodPrefix</td>
</tr>
<tr>
<td>FiscalYearQuarterName</td>
<td>Period.QuarterLabel</td>
</tr>
<tr>
<td>FiscalYearQuarterPrefix</td>
<td>FiscalYearSettings.QuarterPrefix</td>
</tr>
<tr>
<td>IdeaCategory¹</td>
<td>IdeaTheme.Categories¹</td>
</tr>
<tr>
<td>IdeaMultiCategory</td>
<td>Idea.Categories</td>
</tr>
<tr>
<td>IdeaStatus</td>
<td>Idea.Status</td>
</tr>
<tr>
<td>IdeaThemeStatus</td>
<td>IdeaTheme.Status</td>
</tr>
<tr>
<td>Industry</td>
<td>Account.Industry</td>
</tr>
<tr>
<td></td>
<td>Lead.Industry</td>
</tr>
<tr>
<td>LeadSource</td>
<td>Account.AccountSource</td>
</tr>
<tr>
<td></td>
<td>Lead.LeadSource</td>
</tr>
<tr>
<td></td>
<td>Opportunity.Source</td>
</tr>
<tr>
<td>LeadStatus</td>
<td>Lead.Status</td>
</tr>
<tr>
<td>OpportunityCompetitor</td>
<td>Opportunity.Competitors</td>
</tr>
<tr>
<td>OpportunityStage</td>
<td>Opportunity.StageName</td>
</tr>
<tr>
<td>OpportunityType</td>
<td>Opportunity.Type</td>
</tr>
<tr>
<td>OrderStatus</td>
<td>Not available in 37.0 and earlier</td>
</tr>
<tr>
<td>OrderType</td>
<td>Order.Type</td>
</tr>
<tr>
<td>PartnerRole</td>
<td>Account.PartnerRole</td>
</tr>
<tr>
<td>Product2Family</td>
<td>Product2.Family</td>
</tr>
<tr>
<td>QuestionOrigin¹</td>
<td>Question.Origin¹</td>
</tr>
<tr>
<td>QuickTextCategory</td>
<td>QuickText.Category</td>
</tr>
<tr>
<td>QuickTextChannel</td>
<td>QuickText.Channel</td>
</tr>
<tr>
<td>QuoteStatus</td>
<td>Quote.Status</td>
</tr>
<tr>
<td>RoleInTerritory2</td>
<td>UserTerritory2Association.RoleInTerritory2</td>
</tr>
</tbody>
</table>
## Standard Value Set Names and Standard Picklist Fields

<table>
<thead>
<tr>
<th>Standard Value Set Name (API version 38.0 and later)</th>
<th>Field Name (API version 37.0 and earlier)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SalesTeamRole</td>
<td>OpportunityTeamMember.TeamMemberRole</td>
</tr>
<tr>
<td></td>
<td>UserAccountTeamMember.TeamMemberRole</td>
</tr>
<tr>
<td></td>
<td>UserTeamMember.TeamMemberRole</td>
</tr>
<tr>
<td></td>
<td>AccountTeamMember.TeamMemberRole</td>
</tr>
<tr>
<td>Salutation</td>
<td>Contact.Salutation</td>
</tr>
<tr>
<td></td>
<td>Lead.Salutation</td>
</tr>
<tr>
<td>ServiceContractApprovalStatus</td>
<td>ServiceContract.ApprovalStatus</td>
</tr>
<tr>
<td>SocialPostClassification</td>
<td>SocialPost.Classification</td>
</tr>
<tr>
<td>SocialPostEngagementLevel</td>
<td>SocialPost.EngagementLevel</td>
</tr>
<tr>
<td>SocialPostReviewedStatus</td>
<td>SocialPost.ReviewedStatus</td>
</tr>
<tr>
<td>SolutionStatus</td>
<td>Solution.Status</td>
</tr>
<tr>
<td>TaskPriority</td>
<td>Task.Priority</td>
</tr>
<tr>
<td>TaskStatus</td>
<td>Task.Status</td>
</tr>
<tr>
<td>TaskSubject</td>
<td>Task.Subject</td>
</tr>
<tr>
<td>TaskType</td>
<td>Task.Type</td>
</tr>
<tr>
<td>WorkOrderLineItemStatus</td>
<td>WorkOrderLineItem.Status</td>
</tr>
<tr>
<td>WorkOrderPriority</td>
<td>WorkOrder.Priority</td>
</tr>
<tr>
<td>WorkOrderStatus</td>
<td>WorkOrder.Status</td>
</tr>
</tbody>
</table>

1 You can’t read or update this standard value set or picklist field.
Apex
Apex is a strongly typed, object-oriented programming language that allows developers to execute flow and transaction control statements on the Lightning platform server in conjunction with calls to the Lightning Platform API. Using syntax that looks like Java and acts like database stored procedures, Apex enables developers to add business logic to most system events, including button clicks, related record updates, and Visualforce pages. Apex code can be initiated by Web service requests and from triggers on objects.

Apex-Managed Sharing
Enables developers to programmatically manipulate sharing to support their application’s behavior. Apex-managed sharing is only available for custom objects.

App
Short for “application.” A collection of components such as tabs, reports, dashboards, and Visualforce pages that address a specific business need. Salesforce provides standard apps such as Sales and Service. You can customize the standard apps to match the way you work. In addition, you can package an app and upload it to the AppExchange along with related components such as custom fields, custom tabs, and custom objects. Then, you can make the app available to other Salesforce users from the AppExchange.

AppExchange
The AppExchange is a sharing interface from Salesforce that allows you to browse and share apps and services for the Lightning Platform.

AppExchange Upgrades
Upgrading an app is the process of installing a newer version.

Application Programming Interface (API)
The interface that a computer system, library, or application provides to allow other computer programs to request services from it and exchange data.

Asynchronous Calls
A call that doesn’t return results immediately because the operation can take a long time. Calls in the Metadata API and Bulk API are asynchronous.

Boolean Operators
You can use Boolean operators in report filters to specify the logical relationship between two values. For example, the AND operator between two values yields search results that include both values. Likewise, the OR operator between two values yields search results that include either value.

Bulk API
The REST-based Bulk API is optimized for processing large sets of data. It allows you to query, insert, update, upsert, or delete a large number of records asynchronously by submitting a number of batches which are processed in the background by Salesforce. See also SOAP API.
C

Class, Apex
A template or blueprint from which Apex objects are created. Classes consist of other classes, user-defined methods, variables, exception types, and static initialization code. In most cases, Apex classes are modeled on their counterparts in Java.

Client App
An app that runs outside the Salesforce user interface and uses only the Lightning Platform API or Bulk API. It typically runs on a desktop or mobile device. These apps treat the platform as a data source, using the development model of whatever tool and platform for which they are designed.

Component, Metadata
A component is an instance of a metadata type in the Metadata API. For example, CustomObject is a metadata type for custom objects, and the MyCustomObject__c component is an instance of a custom object. A component is described in an XML file and can be deployed or retrieved using the Metadata API, or tools built on top of it, such as the Force.com IDE or the Ant Migration Tool.

Component, Visualforce
Something that can be added to a Visualforce page with a set of tags, for example, `<apex:detail>`. Visualforce includes a number of standard components, or you can create your own custom components.

Component Reference, Visualforce
A description of the standard and custom Visualforce components that are available in your organization. You can access the component library from the development footer of any Visualforce page or the Visualforce Developer’s Guide.

Controller, Visualforce
An Apex class that provides a Visualforce page with the data and business logic it needs to run. Visualforce pages can use the standard controllers that come by default with every standard or custom object, or they can use custom controllers.

Controlling Field
Any standard or custom picklist or checkbox field whose values control the available values in one or more corresponding dependent fields.

Custom App
See App.

Custom Field
A field that can be added in addition to the standard fields to customize Salesforce for your organization’s needs.

Custom Help
Custom text administrators create to provide users with on-screen information specific to a standard field, custom field, or custom object.

Custom Links
Custom links are URLs defined by administrators to integrate your Salesforce data with external websites and back-office systems. Formerly known as Web links.

Custom Object
Custom records that allow you to store information unique to your organization.

Custom S-Control
Note: S-controls have been superseded by Visualforce pages. After March 2010 organizations that have never created s-controls, as well as new organizations, won’t be allowed to create them. Existing s-controls will remain unaffected, and can still be edited.

Custom Web content for use in custom links. Custom s-controls can contain any type of content that you can display in a browser, for example a Java applet, an Active-X control, an Excel file, or a custom HTML Web form.
Database
An organized collection of information. The underlying architecture of the Lightning Platform includes a database where your data is stored.

Database Table
A list of information, presented with rows and columns, about the person, thing, or concept you want to track. See also Object.

Data Manipulation Language (DML)
An Apex method or operation that inserts, updates, or deletes records.

Decimal Places
Parameter for number, currency, and percent custom fields that indicates the total number of digits you can enter to the right of a decimal point, for example, 4.98 for an entry of 2. Note that the system rounds the decimal numbers you enter, if necessary. For example, if you enter 4.986 in a field with Decimal Places of 2, the number rounds to 4.99. Salesforce uses the round half-up rounding algorithm. Half-way values are always rounded up. For example, 1.45 is rounded to 1.5. –1.45 is rounded to –1.5.

Dependent Field
Any custom picklist or multi-select picklist field that displays available values based on the value selected in its corresponding controlling field.

Developer Edition
A free, fully-functional Salesforce organization designed for developers to extend, integrate, and develop with the Lightning Platform. Developer Edition accounts are available on developer.salesforce.com.

Salesforce Developers
The Salesforce Developers website at developer.salesforce.com provides a full range of resources for platform developers, including sample code, toolkits, an online developer community, and the ability to obtain limited Lightning Platform environments.

Document Library
A place to store documents without attaching them to accounts, contacts, opportunities, or other records.

Email Alert
Email alerts are actions that send emails, using a specified email template, to specified recipients.

Email Template
A form email that communicates a standard message, such as a welcome letter to new employees or an acknowledgment that a customer service request has been received. Email templates can be personalized with merge fields, and can be written in text, HTML, or custom format.

*Note:* Lightning email templates aren’t packageable.

Enterprise Edition
A Salesforce edition designed for larger, more complex businesses.

Enterprise WSDL
A strongly-typed WSDL for customers who want to build an integration with their Salesforce organization only, or for partners who are using tools like Tibco or webMethods to build integrations that require strong typecasting. The downside of the Enterprise WSDL is that it only works with the schema of a single Salesforce organization because it is bound to all of the unique objects and fields that exist in that organization’s data model.
Entity Relationship Diagram (ERD)
A data modeling tool that helps you organize your data into entities (or objects, as they are called in the Lightning Platform) and define the relationships between them. ERD diagrams for key Salesforce objects are published in the SOAP API Developer’s Guide.

Enumeration Field
An enumeration is the WSDL equivalent of a picklist field. The valid values of the field are restricted to a strict set of possible values, all having the same data type.

Field
A part of an object that holds a specific piece of information, such as a text or currency value.

Field-Level Security
Settings that determine whether fields are hidden, visible, read only, or editable for users. Available in Professional, Enterprise, Unlimited, Performance, and Developer Editions.

Filter Condition/Criteria
Condition on particular fields that qualifies items to be included in a list view or report, such as “State equals California.”

Lightning Platform
The Salesforce platform for building applications in the cloud. Lightning Platform combines a powerful user interface, operating system, and database to allow you to customize and deploy applications in the cloud for your entire enterprise.

Force.com IDE
An Eclipse plug-in that allows developers to manage, author, debug and deploy Lightning Platform applications in the Eclipse development environment.

Ant Migration Tool
A toolkit that allows you to write an Apache Ant build script for migrating Lightning Platform components between a local file system and a Salesforce organization.

Foreign Key
A field whose value is the same as the primary key of another table. You can think of a foreign key as a copy of a primary key from another table. A relationship is made between two tables by matching the values of the foreign key in one table with the values of the primary key in another.

Formula Field
A type of custom field. Formula fields automatically calculate their values based on the values of merge fields, expressions, or other values.

Function
Built-in formulas that you can customize with input parameters. For example, the DATE function creates a date field type from a given year, month, and day.

Gregorian Year
A calendar based on a 12-month structure used throughout much of the world.
HTTP Debugger
An application that can be used to identify and inspect SOAP requests that are sent from the AJAX Toolkit. They behave as proxy servers running on your local machine and allow you to inspect and author individual requests.

ID
See Salesforce Record ID.

Inline S-Control

Note: S-controls have been superseded by Visualforce pages. After March 2010 organizations that have never created s-controls, as well as new organizations, won’t be allowed to create them. Existing s-controls will remain unaffected, and can still be edited.

An s-control that displays within a record detail page or dashboard, rather than on its own page.

Instance
The cluster of software and hardware represented as a single logical server that hosts an organization’s data and runs their applications. The Lightning Platform runs on multiple instances, but data for any single organization is always stored on a single instance.

Integration User
A Salesforce user defined solely for client apps or integrations. Also referred to as the logged-in user in a SOAP API context.

ISO Code
The International Organization for Standardization country code, which represents each country by two letters.

Junction Object
A custom object with two master-detail relationships. Using a custom junction object, you can model a “many-to-many” relationship between two objects. For example, you create a custom object called “Bug” that relates to the standard case object such that a bug could be related to multiple cases and a case could also be related to multiple bugs.

License Management Application (LMA)
A free AppExchange app that allows you to track sales leads and accounts for every user who downloads your managed package (app) from the AppExchange.

License Management Organization (LMO)
The Salesforce organization that you use to track all the Salesforce users who install your package. A license management organization must have the License Management Application (LMA) installed. It automatically receives notification every time your package is
installed or uninstalled so that you can easily notify users of upgrades. You can specify any Enterprise, Unlimited, Performance, or Developer Edition organization as your license management organization. For more information, go to http://www.salesforce.com/docs/en/lma/index.htm.

List View
A list display of items (for example, accounts or contacts) based on specific criteria. Salesforce provides some predefined views.

In the Agent console, the list view is the top frame that displays a list view of records based on specific criteria. The list views you can select to display in the console are the same list views defined on the tabs of other objects. You cannot create a list view within the console.

Local Project
A .zip file containing a project manifest (package.xml file) and one or more metadata components.

Locale
The country or geographic region in which the user is located. The setting affects the format of date and number fields, for example, dates in the English (United States) locale display as 06/30/2000 and as 30/06/2000 in the English (United Kingdom) locale.

In Professional, Enterprise, Unlimited, Performance, and Developer Edition organizations, a user's individual Locale setting overrides the organization's Default Locale setting. In Personal and Group Editions, the organization-level locale field is called Locale, not Default Locale.

Logged-in User
In a SOAP API context, the username used to log into Salesforce. Client applications run with the permissions and sharing of the logged-in user. Also referred to as an integration user.

Lookup Field
A type of field that contains a linkable value to another record. You can display lookup fields on page layouts where the object has a lookup or master-detail relationship with another object. For example, cases have a lookup relationship with assets that allows users to select an asset using a lookup dialog from the case edit page and click the name of the asset from the case detail page.

Managed Package
A collection of application components that is posted as a unit on the AppExchange and associated with a namespace and possibly a License Management Organization. To support upgrades, a package must be managed. An organization can create a single managed package that can be downloaded and installed by many different organizations. Managed packages differ from unmanaged packages by having some locked components, allowing the managed package to be upgraded later. Unmanaged packages do not include locked components and cannot be upgraded. In addition, managed packages obfuscate certain components (like Apex) on subscribing organizations to protect the intellectual property of the developer.

Manifest File
The project manifest file (package.xml) lists the XML components to retrieve or deploy when working with the Metadata API, or clients built on top of the Metadata API, such as the Force.com IDE or the Ant Migration Tool.

Manual Sharing
Record-level access rules that allow record owners to give read and edit permissions to other users who might not have access to the record any other way.

Many-to-Many Relationship
A relationship where each side of the relationship can have many children on the other side. Many-to-many relationships are implemented through the use of junction objects.
Master-Detail Relationship
A relationship between two different types of records that associates the records with each other. For example, accounts have a master-detail relationship with opportunities. This type of relationship affects record deletion, security, and makes the lookup relationship field required on the page layout.

Metadata
Information about the structure, appearance, and functionality of an organization and any of its parts. Lightning Platform uses XML to describe metadata.

Metadata WSDL
A WSDL for users who want to use the Lightning Platform Metadata API calls.

Multitenancy
An application model where all users and apps share a single, common infrastructure and code base.

Namespace
In a packaging context, a one- to 15-character alphanumeric identifier that distinguishes your package and its contents from packages of other developers on AppExchange, similar to a domain name. Salesforce automatically prepends your namespace prefix, followed by two underscores (“__”), to all unique component names in your Salesforce organization.

Native App
An app that is built exclusively with setup (metadata) configuration on Lightning Platform. Native apps do not require any external services or infrastructure.

Object
An object allows you to store information in your Salesforce organization. The object is the overall definition of the type of information you are storing. For example, the case object allows you to store information regarding customer inquiries. For each object, your organization will have multiple records that store the information about specific instances of that type of data. For example, you might have a case record to store the information about Joe Smith’s training inquiry and another case record to store the information about Mary Johnson’s configuration issue.

Object-Level Help
Custom help text that you can provide for any custom object. It displays on custom object record home (overview), detail, and edit pages, as well as list views and related lists.

Object-Level Security
Settings that allow an administrator to hide whole objects from users so that they don’t know that type of data exists. Object-level security is specified with object permissions.

onClick JavaScript
JavaScript code that executes when a button or link is clicked.

One-to-Many Relationship
A relationship in which a single object is related to many other objects. For example, an account may have one or more related contacts.
Organization-Wide Defaults
Settings that allow you to specify the baseline level of data access that a user has in your organization. For example, you can set organization-wide defaults so that any user can see any record of a particular object that is enabled via their object permissions, but they need extra permissions to edit one.

Outbound Message
An outbound message sends information to a designated endpoint, like an external service. Outbound messages are configured from Setup. You must configure the external endpoint and create a listener for the messages using the SOAP API.

Overlay
An overlay displays additional information when you hover your mouse over certain user interface elements. Depending on the overlay, it will close when you move your mouse away, click outside of the overlay, or click a close button.

Owner
Individual user to which a record (for example, a contact or case) is assigned.

Package
A group of Lightning Platform components and applications that are made available to other organizations through the AppExchange. You use packages to bundle an app along with any related components so that you can upload them to AppExchange together.

Partner WSDL
A loosely-typed WSDL for customers, partners, and ISVs who want to build an integration or an AppExchange app that can work across multiple Salesforce organizations. With this WSDL, the developer is responsible for marshaling data in the correct object representation, which typically involves editing the XML. However, the developer is also freed from being dependent on any particular data model or Salesforce organization. Contrast this with the Enterprise WSDL, which is strongly typed.

Picklist
Selection list of options available for specific fields in a Salesforce object, for example, the Industry field for accounts. Users can choose a single value from a list of options rather than make an entry directly in the field. See also Master Picklist.

Picklist (Multi-Select)
Selection list of options available for specific fields in a Salesforce object. Multi-select picklists allow users to choose one or more values. Users can choose a value by double clicking on it, or choose additional values from a scrolling list by holding down the CTRL key while clicking a value and using the arrow icon to move them to the selected box.

Picklist Values
Selections displayed in drop-down lists for particular fields. Some values come predefined, and other values can be changed or defined by an administrator.

Primary Key
A relational database concept. Each table in a relational database has a field in which the data value uniquely identifies the record. This field is called the primary key. The relationship is made between two tables by matching the values of the foreign key in one table with the values of the primary key in another.

Production Organization
A Salesforce organization that has live users accessing data.

Professional Edition
A Salesforce edition designed for businesses who need full-featured CRM functionality.
Glossary

Q

Queue
A holding area for items before they are processed. Salesforce uses queues in a number of different features and technologies.

Query String Parameter
A name-value pair that’s included in a URL, typically after a ‘?’ character. For example:

https://yourInstance.salesforce.com/001/e?name=value

R

Record
A single instance of a Salesforce object. For example, “John Jones” might be the name of a contact record.

Record Name
A standard field on all Salesforce objects. Whenever a record name is displayed in a Lightning Platform application, the value is represented as a link to a detail view of the record. A record name can be either free-form text or an autonumber field. Record Name does not have to be a unique value.

Record Type
A record type is a field available for certain records that can include some or all of the standard and custom picklist values for that record. You can associate record types with profiles to make only the included picklist values available to users with that profile.

Record-Level Security
A method of controlling data in which you can allow a particular user to view and edit an object, but then restrict the records that the user is allowed to see.

Recycle Bin
A page that lets you view and restore deleted information. Access the Recycle Bin by using the link in the sidebar.

Related Object
Objects chosen by an administrator to display in the Agent console’s mini view when records of a particular type are shown in the console’s detail view. For example, when a case is in the detail view, an administrator can choose to display an associated account, contact, or asset in the mini view.

Relationship
A connection between two objects, used to create related lists in page layouts and detail levels in reports. Matching values in a specified field in both objects are used to link related data; for example, if one object stores data about companies and another object stores data about people, a relationship allows you to find out which people work at the company.

Relationship Query
In a SOQL context, a query that traverses the relationships between objects to identify and return results. Parent-to-child and child-to-parent syntax differs in SOQL queries.

Report Type
A report type defines the set of records and fields available to a report based on the relationships between a primary object and its related objects. Reports display only records that meet the criteria defined in the report type. Salesforce provides a set of pre-defined standard report types; administrators can create custom report types as well.

Role Hierarchy
A record-level security setting that defines different levels of users such that users at higher levels can view and edit information owned by or shared with users beneath them in the role hierarchy, regardless of the organization-wide sharing model settings.
Roll-Up Summary Field
A field type that automatically provides aggregate values from child records in a master-detail relationship.

S

SaaS
See Software as a Service (SaaS).

S-Control

Note: S-controls have been superseded by Visualforce pages. After March 2010 organizations that have never created s-controls, as well as new organizations, won’t be allowed to create them. Existing s-controls will remain unaffected, and can still be edited.

Custom Web content for use in custom links. Custom s-controls can contain any type of content that you can display in a browser, for example a Java applet, an Active-X control, an Excel file, or a custom HTML Web form.

Salesforce Record ID
A unique 15- or 18-character alphanumeric string that identifies a single record in Salesforce.

Salesforce SOA (Service-Oriented Architecture)
A powerful capability of Lightning Platform that allows you to make calls to external Web services from within Apex.

Sandbox
A nearly identical copy of a Salesforce production organization for development, testing, and training. The content and size of a sandbox varies depending on the type of sandbox and the edition of the production organization associated with the sandbox.

Search Layout
The organization of fields included in search results, in lookup dialogs, and in the key lists on tab home pages.

Session ID
An authentication token that is returned when a user successfully logs in to Salesforce. The Session ID prevents a user from having to log in again every time they want to perform another action in Salesforce. Different from a record ID or Salesforce ID, which are terms for the unique ID of a Salesforce record.

Session Timeout
The time after login before a user is automatically logged out. Sessions expire automatically after a predetermined length of inactivity, which can be configured in Salesforce from Setup by clicking Security Controls. The default is 120 minutes (two hours). The inactivity timer is reset to zero if a user takes an action in the web interface or makes an API call.

Setup
A menu where administrators can customize and define organization settings and Lightning Platform apps. Depending on your organization’s user interface settings, Setup may be a link in the user interface header or in the dropdown list under your name.

Sharing
Allowing other users to view or edit information you own. There are different ways to share data:

• Sharing Model—defines the default organization-wide access levels that users have to each other’s information and whether to use the hierarchies when determining access to data.

• Role Hierarchy—defines different levels of users such that users at higher levels can view and edit information owned by or shared with users beneath them in the role hierarchy, regardless of the organization-wide sharing model settings.

• Sharing Rules—allow an administrator to specify that all information created by users within a given group or role is automatically shared to the members of another group or role.

• Manual Sharing—allows individual users to share records with other users or groups.
• Apex-Managed Sharing—enables developers to programmatically manipulate sharing to support their application’s behavior. See Apex-Managed Sharing.

Sharing Model
Behavior defined by your administrator that determines default access by users to different types of records.

Sharing Rule
Type of default sharing created by administrators. Allows users in a specified group or role to have access to all information created by users within a given group or role.

Sites
Salesforce Sites enables you to create public websites and applications that are directly integrated with your Salesforce organization—without requiring users to log in with a username and password.

Snippet

Note: S-controls have been superseded by Visualforce pages. After March 2010 organizations that have never created s-controls, as well as new organizations, won’t be allowed to create them. Existing s-controls will remain unaffected, and can still be edited.

A type of s-control that is designed to be included in other s-controls. Similar to a helper method that is used by other methods in a piece of code, a snippet allows you to maintain a single copy of HTML or JavaScript that you can reuse in multiple s-controls.

SOAP (Simple Object Access Protocol)
A protocol that defines a uniform way of passing XML-encoded data.

Software as a Service (SaaS)
A delivery model where a software application is hosted as a service and provided to customers via the Internet. The SaaS vendor takes responsibility for the daily maintenance, operation, and support of the application and each customer’s data. The service alleviates the need for customers to install, configure, and maintain applications with their own hardware, software, and related IT resources. Services can be delivered using the SaaS model to any market segment.

SOQL (Salesforce Object Query Language)
A query language that allows you to construct simple but powerful query strings and to specify the criteria that selects data from the Lightning Platform database.

SOSL (Salesforce Object Search Language)
A query language that allows you to perform text-based searches using the Lightning Platform API.

Standard Object
A built-in object included with the Lightning Platform. You can also build custom objects to store information that is unique to your app.

System Log
Part of the Developer Console, a separate window console that can be used for debugging code snippets. Enter the code you want to test at the bottom of the window and click Execute. The body of the System Log displays system resource information, such as how long a line took to execute or how many database calls were made. If the code did not run to completion, the console also displays debugging information.

Test Method
An Apex class method that verifies whether a particular piece of code is working properly. Test methods take no arguments, commit no data to the database, and can be executed by the \runTests() system method either through the command line or in an Apex IDE, such as the Force.com IDE.
Translation Workbench
The Translation Workbench lets you specify languages you want to translate, assign translators to languages, create translations for customizations you’ve made to your Salesforce organization, and override labels and translations from managed packages. Everything from custom picklist values to custom fields can be translated so your global users can use Salesforce in their language.

Trigger
A piece of Apex that executes before or after records of a particular type are inserted, updated, or deleted from the database. Every trigger runs with a set of context variables that provide access to the records that caused the trigger to fire, and all triggers run in bulk mode—that is, they process several records at once, rather than just one record at a time.

Trigger Context Variable
Default variables that provide access to information about the trigger and the records that caused it to fire.

Unit Test
A unit is the smallest testable part of an application, usually a method. A unit test operates on that piece of code to make sure it works correctly. See also Test Method.

Unlimited Edition
Unlimited Edition is Salesforce’s solution for maximizing your success and extending that success across the entire enterprise through the Lightning Platform.

Unmanaged Package
A package that cannot be upgraded or controlled by its developer.

URL (Uniform Resource Locator)
The global address of a website, document, or other resource on the Internet. For example, http://www.salesforce.com.

URL S-Control
Note: S-controls have been superseded by Visualforce pages. After March 2010 organizations that have never created s-controls, as well as new organizations, won’t be allowed to create them. Existing s-controls will remain unaffected, and can still be edited.

An s-control that contains an external URL that hosts the HTML that should be rendered on a page. When saved this way, the HTML is hosted and run by an external website. URL s-controls are also called web controls.

Validation Rule
A rule that prevents a record from being saved if it does not meet the standards that are specified.

Visualforce
A simple, tag-based markup language that allows developers to easily define custom pages and components for apps built on the platform. Each tag corresponds to a coarse or fine-grained component, such as a section of a page, a related list, or a field. The components can either be controlled by the same logic that is used in standard Salesforce pages, or developers can associate their own logic with a controller written in Apex.
Web Control
See URL S-Control.

Web Links
See Custom Links.

Web Service
A mechanism by which two applications can easily exchange data over the Internet, even if they run on different platforms, are written in different languages, or are geographically remote from each other.

Web Service Method
An Apex class method or variable that external systems can use, like a mash-up with a third-party application. Web service methods must be defined in a global class.

Web Services API
A Web services application programming interface that provides access to your Salesforce organization’s information. See also SOAP PI and Bulk API.

Web Tab
A custom tab that allows your users to use external websites from within the application.

Workflow Action
A workflow action, such as an email alert, field update, outbound message, or task, fires when the conditions of a workflow rule are met.

Workflow Email Alert
A workflow action that sends an email when a workflow rule is triggered. Unlike workflow tasks, which can only be assigned to application users, workflow alerts can be sent to any user or contact, as long as they have a valid email address.

Workflow Field Update
A workflow action that changes the value of a particular field on a record when a workflow rule is triggered.

Workflow Outbound Message
A workflow action that sends data to an external Web service, such as another cloud computing application. Outbound messages are used primarily with composite apps.

Workflow Queue
A list of workflow actions that are scheduled to fire based on workflow rules that have one or more time-dependent workflow actions.

Workflow Rule
A workflow rule sets workflow actions into motion when its designated conditions are met. You can configure workflow actions to execute immediately when a record meets the conditions in your workflow rule, or set time triggers that execute the workflow actions on a specific day.

Workflow Task
A workflow action that assigns a task to an application user when a workflow rule is triggered.

WSDL (Web Services Description Language) File
An XML file that describes the format of messages you send and receive from a Web service. Your development environment’s SOAP client uses the Salesforce Enterprise WSDL or Partner WSDL to communicate with Salesforce using the SOAP API.
XML (Extensible Markup Language)
A markup language that enables the sharing and transportation of structured data. All Lightning Platform components that are retrieved or deployed through the Metadata API are represented by XML definitions.

Zip File
A data compression and archive format.
A collection of files retrieved or deployed by the Metadata API. See also Local Project.