
Custom Metadata Types Implementation Guide

Salesforce, Summer '16



CONTENTS

- CUSTOM METADATA TYPES 1**
- CUSTOM METADATA TYPES LIMITATIONS 3**
- CUSTOM METADATA LIMITS 5**
- CREATE, EDIT, AND DELETE CUSTOM METADATA TYPES AND RECORDS 6**
 - Define a Custom Metadata Type Declaratively 6
 - Add or Edit Custom Metadata Records Declaratively 8
 - Load Records with the Custom Metadata Loader 8
- ACCESS CUSTOM METADATA TYPES, RECORDS, AND FIELDS 9**
 - Access Custom Metadata Types and Records 9
 - Access Custom Metadata Fields 10
- PACKAGE CUSTOM METADATA TYPES AND RECORDS 11**
 - Considerations for Custom Metadata Type Packages 12

CUSTOM METADATA TYPES

You can create your own declarative developer frameworks for internal teams, partners, and customers. Rather than building apps from data, you can build apps that are defined and driven by their own types of metadata. Metadata is the information that describes the configuration of each customer's organization.

Custom metadata is customizable, deployable, packageable, and upgradeable application metadata. First, you create a *custom metadata type*, which defines the form of the application metadata. Then you build reusable functionality that determines the behavior based on metadata of that type. Similar to a custom object or custom setting, a custom metadata type has a list of custom fields that represent aspects of the metadata. After you create a public custom metadata type, you or others can declaratively create *custom metadata records* that are defined by that type. When you package a public custom metadata type, customers who install the package can add their own records to the metadata type. Your reusable functionality reads your custom metadata and uses it to produce customized application behavior.

Custom metadata rows resemble custom object rows in structure. You create, edit, and delete custom metadata rows in the Metadata API or in Setup. Because the records are metadata, you can migrate them using packages or Metadata API tools. Custom metadata records are read-only in Apex and in the Enterprise and Partner APIs.

With custom metadata types, you can issue unlimited Salesforce Object Query Language (SOQL) queries for each Apex transaction.

Custom metadata types support the following custom field types.

- Checkbox
- Date
- Date and Time
- Email
- Number
- Percent
- Phone
- Picklist
- Text
- Text Area
- URL



Note: This release contains a beta version of picklists on custom metadata types that is production quality but has known limitations.

A subscriber to a managed package containing a custom metadata type can't add their own fields to that type. Only the org that develops the type can add custom fields to it.

Custom metadata fields are *manageable*, which means that the developer of a type can decide who can change field values after they are deployed to a subscriber organization.

- Locked after release—For any record of the type, the value of the field is immutable after deployment, even on the developer organization where the record was created.
- Subscriber editable—Anyone with the correct permissions can change the value of the field at will. Any changes the developer deploys do not overwrite values in the subscriber's organization.

EDITIONS

Available in: Salesforce Classic

Available in: **Enterprise, Performance, Unlimited, Developer,** and **Database.com** Editions

Custom Metadata Types

- Upgradable—The developer of a record can change the value of the field by releasing a new version of the custom metadata package. The subscriber can't change the value of the field.

Custom metadata types and records have names and labels. Type names must be unique within their namespace. Record names must be unique within their custom metadata type and namespace.

Custom metadata records can be protected. If a developer releases protected records in a managed package, access to them 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 only with a package upgrade. The subscriber can't read or modify protected records. The developer name of a protected record can't be changed after release.

If you create a protected custom metadata record in your organization, then it's accessible only by your code, code from unmanaged packages, and code from the managed package that defines its type.

Custom metadata types can also be protected, providing the same access protection as protected records. If you change a type from protected to public, its protected records remain protected and all other records become public. If you use Setup to create a new record on a protected type, the Protected Component checkbox is checked by default. Once a type is public, you can't convert it to protected. The subscriber can't create records of a protected type.

CUSTOM METADATA TYPES LIMITATIONS

When using custom metadata types, be aware of these special behaviors and limitations.

Updating Types and Records

You can't update protected types and records in an installed managed package programmatically. You can modify protected types and records only by performing a package upgrade.

You can't update public types and records by using Apex directly. To modify records from Apex, you must make calls to the Metadata API.

Application lifecycle management tools

Custom metadata types don't support these application lifecycle management tools:

- Version control
- Tooling API
- Developer Console

Licenses

Licenses that are defined for an extension package aren't enforced on custom metadata records in that package unless the types are also in the package.

SOQL

Custom metadata types support the following SOQL query syntax.

```
SELECT fieldList [...]
FROM objectType
    [USING SCOPE filterScope]
[WHERE conditionExpression]
[ORDER BY field {ASC|DESC} [NULLS {FIRST|LAST}} ]
```

- The *fieldList* can include only non-relationship fields.
- FROM can include only one object.
- You can use the following operators.
 - IN and NOT IN
 - =, >, >=, <, <=, and !=
 - LIKE, including wild cards
 - AND
- You can use ORDER BY, ASC, and DESC with multiple fields.
- You can only use ORDER BY when the ordered field is a selected field.

Protected custom metadata types

Subscribers can't add custom metadata records to installed custom metadata types that are protected. To allow subscribers to create custom metadata records that are defined by a custom metadata type, the type must be public.

Metadata API returns protected custom entity definitions (but not custom metadata records) in subscriber organizations.

EDITIONS

Available in: Salesforce Classic

Available in: **Enterprise, Performance, Unlimited, Developer,** and **Database.com** Editions

Custom Metadata Types Limitations

Caching

Custom metadata records are cached at the type level after the first read request. This enhances performance on subsequent requests. Requests that are in flight when metadata is updated won't get the most recent metadata.

Global Picklists


Global picklists aren't supported on custom metadata types. You can only use sObject picklists.

Picklists and Released Managed Packages

Subscribers to a released managed package that contains a custom metadata type with a picklist can't add or remove values from that picklist. Developers who release a managed packaged that contains a custom metadata type with a picklist can add values to the picklist, but can't delete them.

CUSTOM METADATA LIMITS

Be aware of these requirements for custom metadata types and records.

Description	Maximum amount
SOQL queries per Apex transaction	Unlimited
Custom metadata per organization *	10 MB
Custom metadata per certified managed package *	10 MB
	 Note: Custom metadata records in certified managed packages that you've installed don't count toward your organization's limit. However, custom metadata records that you create do count toward the limit. This rule applies regardless of whether you create records in your own custom metadata type or in a type from a certified managed package.
Fields per custom metadata type or record	100
Custom metadata types per organization	100. This number includes all types developed in the organization and installed from managed and unmanaged packages.
Characters per description field	1,000
Records returned per transaction	50,000
Custom metadata records in one call	200

* Record size is based on the maximum field size of each field type, not the actual storage that's used in each field. When adding fields to a custom metadata record, use the appropriate type and specify a length that doesn't exceed what's needed for your data. This action helps you avoid reaching the cached data limit. For example, if you create a US social security number (SSN) field, select the `Text` data type and specify a length of 9. If instead you selected `Text Area`, the field would add 255 characters to the usage count for each record, regardless of the number of characters entered.

EDITIONS

Available in: Salesforce Classic

Available in: **Enterprise, Performance, Unlimited, Developer,** and **Database.com** Editions

CREATE, EDIT, AND DELETE CUSTOM METADATA TYPES AND RECORDS

To create, update, and delete custom metadata types and records, use the Metadata API.

For more information, see “Custom Metadata Types (CustomObject)” in the [Metadata API Developer Guide](#)

[Define a Custom Metadata Type Declaratively](#)

Use Salesforce UI to create and update custom metadata types declaratively.

[Add or Edit Custom Metadata Records Declaratively](#)

You can add, delete, or modify custom metadata declaratively from Setup.

[Load Records with the Custom Metadata Loader](#)

Use the custom metadata loader to bulk load records to your custom metadata types from a .csv file.

EDITIONS

Available in: Salesforce Classic

Available in: **Professional, Enterprise, Performance, Unlimited, Developer,** and **Database.com** Editions

Professional Edition organizations can create, edit, and delete only custom metadata records from types in installed packages.

Define a Custom Metadata Type Declaratively

Use Salesforce UI to create and update custom metadata types declaratively.

1. Search Setup for **Custom Metadata Types**.
2. On the All Custom Metadata Types page, click **New Custom Metadata Type**, or click **Edit** to modify an existing custom metadata type.
3. Complete these fields.

Field	Description
Label	This name is used to refer to the type in a user interface page.
Plural Label	The plural name of the type. If you create a tab for this type, this name is used for the tab.
Starts with a vowel sound	If it is appropriate for your organization’s default language, indicate whether the label is preceded by “an” instead of “a.”
Object Name	A unique name used to refer to the object when using the API. In managed packages, this name prevents naming conflicts with package installations. Use only alphanumeric characters and underscores. The name must begin with a letter and have no spaces. It cannot end with an underscore nor have two consecutive underscores.
Description	An optional description of the object. A meaningful description helps you remember the differences between your custom objects when you are viewing them in a list.


EDITIONS

Available in:

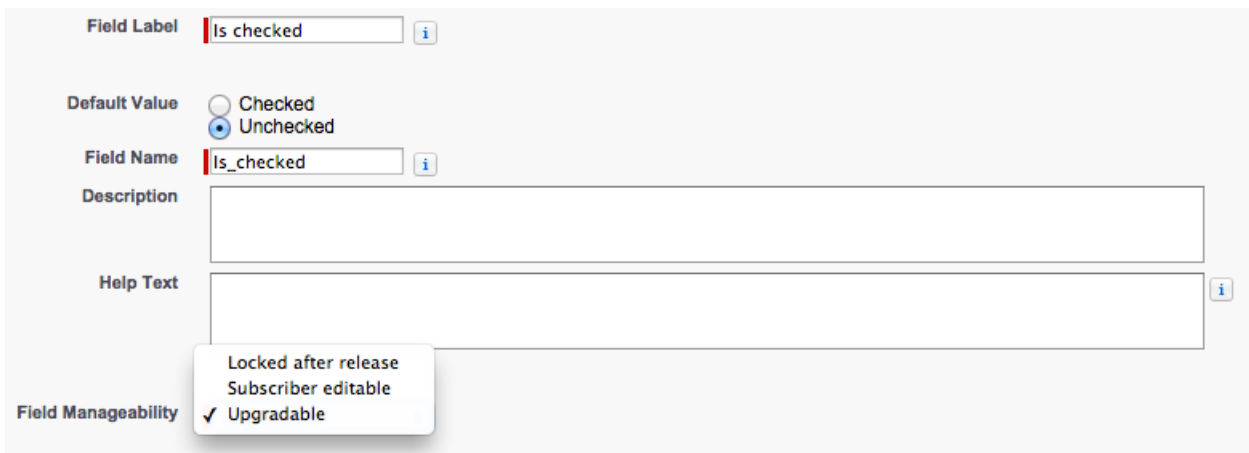
USER PERMISSIONS


To create or edit custom metadata types:

- “Author Apex”

Field	Description
Context-Sensitive Help Setting	<p>Defines what displays when a user clicks Help for this Page from the custom object record home (overview), edit, and detail pages, list views, and related lists.</p> <p>To display the standard Salesforce help available for any custom object record, select Open the standard Salesforce Help & Training window.</p> <p>To display custom object-level help for your custom object, select Open a window using a Visualforce page and then select the Visualforce page to use as the target of the context-sensitive help link from that custom object's pages.</p> <p> Note: This setting doesn't affect the Help & Training link at the top of a page. That link always opens the Salesforce Help & Training window.</p>
Content Name	The name used in page layouts, list views, related lists, and search results.
Visibility	<p>Who should see the type:</p> <ul style="list-style-type: none"> Public—anyone can see it. Protected—if the type is installed as part of a managed package, only Apex code in that managed package can use it.

- Click **Save**.
- Under **Custom Fields**, click **New** to start adding fields to the custom metadata type. For each field, remember to choose a **Field Manageability** value to determine who can change the field later.



-  **Note:** Custom metadata types that were created before the Winter '15 release don't automatically get layouts. Before adding records to this kind of custom metadata type using the UI, you must add a layout that contains all the fields that you want to make editable for the custom metadata type. In the All Custom Metadata Types page, click the custom metadata type. Then click **New** under Page Layouts. If you plan to release a custom metadata type as a managed package, make sure you add all the fields you want to add first. After a customer downloads the managed package, any changes to the layout must be done manually because you can't add fields to a layout via an upgrade.

Add or Edit Custom Metadata Records Declaratively

You can add, delete, or modify custom metadata declaratively from Setup.

1. Search Setup for `Custom Metadata Types`.
2. Click **Manage Records** next to the type of custom metadata that you want to add or modify.
3. On the list of custom metadata records, click **New**, or click **Edit** to modify an existing custom metadata record.
4. Fill out the fields. The **Protected Component** checkbox determines whether the record is *protected*. A protected record is only accessible to code in the same namespace as either the record or its associated custom metadata type: code you create, code in an unmanaged package, and code in the same managed package as either the protected record or its custom metadata type.
5. Click **Save**.

EDITIONS

Available in: **Professional, Enterprise, Performance, Unlimited, Developer**, and **Database.com** Editions

USER PERMISSIONS

To create or modify custom metadata records:

- “CustomizeApplication”

Load Records with the Custom Metadata Loader

Use the custom metadata loader to bulk load records to your custom metadata types from a `.csv` file.

The custom metadata loader lets you load up to 200 records with a single call.

1. [Download](#) the tool from GitHub. Deploy the package to your org via Workbench. Note that you have to create a `.zip` file of the contents of the `custom_md_loader` directory instead of zipping up the directory itself.
2. Create a `.csv` file with a header that contains the custom metadata type's field API names. Either the `Label` field or the `Developer Name` field is required. See `sample.csv` in your download for an example. If your org is namespaced, be sure to include the namespace prefix in your header.
3. From Setup, assign the Custom Metadata Loader permission set to the appropriate users, including yourself.
4. Select **Custom Metadata Loader** from the App Picker.
5. Go to the Custom Metadata Loader tab. The app prompts you to configure your Remote Site Settings if you haven't already done so.
6. Select your `.csv` file and the corresponding custom metadata type.
7. Click **Create custom metadata** to bulk load the records from the `.csv` file.

EDITIONS

Available in: **Enterprise, Performance, Unlimited, Developer**, and **Database.com**

USER PERMISSIONS

To create custom metadata records:

- “Customize Application”

To use the custom metadata loader:

- “Custom Metadata Loader”

ACCESS CUSTOM METADATA TYPES, RECORDS, AND FIELDS

Access Custom Metadata Types and Records

Use SOQL to access your custom metadata types and to retrieve the API names of the records on those types. DML operations aren't allowed on custom metadata in Apex, the Partner APIs, and Enterprise APIs.

For information about the *Custom Metadata Type__mdt* sObject, see [Custom Metadata Type__mdt](#) in the *Object Reference for Salesforce and Force.com*.

For example, declare an Apex variable *custMeta* of the custom metadata type *MyCustomMetadataType__mdt*, which is in your namespace, as follows.

```
MyCustomMetadataType__mdt custMeta;
```

Declare the *custMeta* variable of the custom metadata type *TheirCustomMetadataType__mdt*, which isn't in your namespace but is in the *their_ns* namespace, as follows.

```
their_ns__TheirCustomMetadataType__mdt custMeta;
```

To get the names of all objects of the *MyMdt__mdt* custom metadata type:

```
MyMdt__mdt[] allEntityNames = [select QualifiedApiName from MyMdt__mdt]
```

You can't use `queryMore()` with custom metadata, but you can use the SOQL keywords `LIMIT` and `OFFSET` to page through large numbers of records. For more information, see [Paginating Data for Force.com Applications](#).

Alternatively, to provide an entity that looks more like a `Schema.SObjectDescribeResult` than SOQL, make the Apex class `Acme.MyMdtDescribeResult` encapsulate the information queried from `Acme__MyMdt`. Then create the class `Acme.Acme` with methods such as:

```
Acme.MyMdtDescribeResult describeMyMdt(String qualifiedApiName) {  
    ///perform queries and create object  
}
```

EDITIONS

Available in: Salesforce Classic

Available in: **Professional, Enterprise, Performance, Unlimited, Developer, and Database.com** Editions

Professional Edition organizations can access only custom metadata records from installed custom metadata types.

Access Custom Metadata Fields

Read-only access to the fields on your custom metadata types and records is available through SOQL.

Custom fields on custom metadata types in SOQL are referred to in the same way as they are in the Metadata API. For example, the following SOQL statement retrieves all `Field__c` and `Picklist__c` values of any `PicklistUsage__mdt` related to any custom object named `InterplanetaryGreeting__c`.

```
SELECT Field__c, Picklist__c
FROM PicklistUsage__mdt
WHERE SObjectType__c = 'InterplanetaryGreeting__c'
```

The information that's common to all custom metadata is represented as standard fields. For more information, see "*Custom Metadata Type__mdt*" in the [Object Reference for Salesforce and Force.com](#).

The following Apex statement in the `picklist1234` namespace retrieves the label and namespace for the custom metadata that's represented in the file-based Metadata API as `picklist1234__ReusablePicklistOption.travelApp1234__Motel6`. This statement assigns the object to the variable `motelEx`.

```
ReusablePicklistOption__mdt motelEx = [SELECT MasterLabel, NamespacePrefix
FROM ReusablePicklistOption__mdt
WHERE NamespacePrefix = 'travelApp1234'
AND DeveloperName='Motel6'];
```



Note: Subscribers can run packaged Apex code that queries protected custom metadata types in the same package. However, subscribers can't query protected types in an installed package by using Apex code that they have written.

EDITIONS

Available in: Salesforce Classic

Available in: **Professional, Enterprise, Performance, Unlimited, Developer,** and **Database.com** Editions

Professional Edition organizations can access only custom metadata fields from installed custom metadata types.

PACKAGE CUSTOM METADATA TYPES AND RECORDS

You can package custom metadata types and records in unmanaged packages, managed packages, or managed package extensions. Your packages can then be installed in Professional, Developer, Enterprise, Performance, Unlimited, and Database.com Edition organizations. Use change sets to deploy custom metadata types and records from a sandbox.

You can add custom metadata types and records to packages using the Force.com user interface. From Setup, enter *Packages* in the **Quick Find** box, then select **Packages**, click your package name, and then click **Add**.

Then, to add custom metadata types:

1. Select the **Custom Metadata Type** component type.
2. Select the custom metadata type to add to your package.
3. Click **Add to Package**.

To add custom metadata records:

1. Select the custom metadata type's label from the available component types—for example, `ReusablePicklist__mdt`, or if the type is from a package that you're extending, `ReusablePicklist__mdt [picklist1234]`.
2. Select the records to add.
3. Click **Add to Package**.

If you add a record to your package, its corresponding type is added. If you add a record to a change set, its corresponding type is included in the list of dependent components.

For information on change sets and deploying your package, see the [Development Lifecycle Guide](#).



Note: You can't uninstall a package with a custom metadata type if you've created your own records of that custom metadata type.

As with all packageable metadata components, you can also add custom metadata types and records to a package by specifying the package's full name in your `package.xml` file. For example, we specify the package in this fragment from Picklists R Us's `package.xml` file.

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

EDITIONS

Available in: Salesforce Classic

Available in: **Enterprise, Performance, Unlimited, Developer,** and **Database.com** Editions

Considerations for Custom Metadata Type Packages

Be aware of the following behaviors for packages that contain custom metadata types.

Considerations for Custom Metadata Type Packages

Be aware of the following behaviors for packages that contain custom metadata types.

Once you upload a Managed - Released package that contains a custom metadata type, you can't:

- Add required fields to the custom metadata type
- Set any non-required fields to required
- Delete custom fields

EDITIONS

Available in: Salesforce Classic

Available in: **Enterprise, Performance, Unlimited, Developer,** and **Database.com** Editions