

ISVforce Guide

Version 34.0, Summer '15



CONTENTS

Chapter 1: Introduction	1
Resources for Partners	2
Roles in the Application Lifecycle	2
How to Sign Up for Test Environments	3
Chapter 2: ISVforce Quick Start	4
Tutorial #1: Sign Up for AppExchange	5
Step 1: Sign Up for the Partner Program	5
Step 2: Create a Development and Test Environment	5
Step 3: Get a Business Org	6
Step 4: Edit Your Publisher Profile	6
Sign-up Summary	6
Tutorial #2: Developing Your App	7
Step 1: Create an App	7
Step 2: Package Your App	8
Step 3: Assign a Namespace	8
Step 4: Upload a Beta	9
Step 5: Install and Test the Beta	9
Development Summary	10
Tutorial #3: Publishing and Licensing	11
Step 1: Uploading to the AppExchange	11
Step 2: Create an AppExchange Listing	11
Step 3: Complete the AppExchange Listing	12
Step 4: Manage Licenses for Your App	12
Publishing and Licensing Summary	13
Tutorial #4: Updating Your App	13
Step 1: Creating a Patch Organization	14
Step 2: Developing a Patch	14
Step 3: Uploading the Patch	15
Step 4: Installing or Pushing a Patch	15
Updating Your App Summary	16
Chapter 3: Designing and Building Your App	17
Overview of Packages	18
Planning the Release of Managed Packages	19
Creating and Editing a Package	20
Developing and Distributing Unmanaged Packages	20
Components Available in Managed Packages	21
Editing Components and Attributes after Installation	27

Contents

Components Automatically Added to Packages	34
Special Behavior of Components in Packages	36
Protected Components	45
Understanding Dependencies	46
About Permission Sets and Profile Settings	47
Creating Custom Profile Settings	48
Protecting Your Intellectual Property	49
Creating Packaged Applications with Chatter	49
Matching the Salesforce Look and Feel	50
Developing App Documentation	51
About API and Dynamic Apex Access in Packages	51
Managing API and Dynamic Apex Access in Packages	54
Configuring Default Package Versions for API Calls	55
About the Partner WSDL	56
Generating an Enterprise WSDL with Managed Packages	57
Working with External Services	57
Architectural Considerations for Group and Professional Editions	58
Features Available in Group and Professional Editions	59
Limits for Group and Professional Editions	60
Access Control in Group and Professional Editions	60
Using Apex in Group and Professional Editions	60
API Access in Group and Professional Editions	61
Designing Your App to Support Multiple Editions	62
Sample Design Scenarios for Group and Professional Editions	64
Connected Apps Overview	64
Creating a Connected App	66
Edit, Package, or Delete a Connected App	72
Installing a Connected App	74
View Connected App Details	75
Managing a Connected App	76
Edit a Connected App	78
Monitoring Usage for a Connected App	81
Uninstalling a Connected App	82
Environment Hub	83
Set Up the Environment Hub	84
Setting up My Domain for the Environment Hub	85
Environment Hub Best Practices	86
Connecting an Organization to the Environment Hub	87
Viewing an Environment Hub Member's Details	87
Editing an Environment Hub Member's Details	89
Creating a New Organization from the Environment Hub	90
Enabling Single Sign-On in the Environment Hub	91
Disabling Single Sign-On in the Environment Hub	92
Mapping Users for Single Sign-On in the Environment Hub	92

Chapter 4: Packaging and Testing Your App	93
About Managed Packages	94
Configuring Your Developer Settings	94
Registering a Namespace Prefix	95
Specifying a License Management Organization	96
Creating and Uploading a Managed Package	98
Viewing Package Details	100
Installing a Package	103
Component Availability After Deployment	105
Uninstalling a Package	105
Installing Managed Packages using the API	106
Resolving Apex Test Failures	107
Running Apex on Package Install/Upgrade	108
Running Apex on Package Uninstall	111
Publishing Extensions to Managed Packages	113
Chapter 5: Passing the Security Review	114
Security Review	115
Security Review Steps	115
Security Review Wizard	117
Submit a Client or Mobile App for Security Review	118
Submit an Extension Package for Security Review	118
Security Review Resources	118
Security Review FAQ	119
Chapter 6: Publish Your Offering on the AppExchange	123
What Is the AppExchange?	124
Publish on the AppExchange	124
Connect a Packaging Organization to the AppExchange	124
Create or Edit Your Provider Profile	125
Create or Edit an AppExchange Listing	125
Make Your AppExchange Listing Effective	126
Choose an Installation Option	126
Register Your Package and Choose License Settings	127
Submit Your Listing for Security Review	127
Email Notifications	128
Manage Billing and Subscriptions with AppExchange Checkout	129
AppExchange Checkout FAQ	129
Work with AppExchange Leads	132
AppExchange Leads FAQ	133
Analytics Reports for Publishers	135
Update the Package in an AppExchange Listing	136
AppExchange FAQ	137

Chapter 7: Managing Orders	141
Key Objects in the Channel Order App	142
Installing and Configuring the Channel Order App	142
Install the Channel Order Application	142
Configure the Channel Order App: Define a New Email Service	145
Configure the Channel Order App: Provide Service Order Credentials	147
Add Custom Fields to the Order Detail Page	148
Enable the Channel Order App for Non-Admin Users	149
Managing Contract Terms and Product Catalogs	151
Add New Contract Terms	151
Update Contract Terms and Product Catalogs	152
Submitting and Managing Orders	154
View and Manage Orders	154
Submit an Order	155
Using the Partner Order Submit API	158
Partner Order Submit API	158
Chapter 8: Managing Licenses	163
License Management App (LMA) Overview	164
Understanding License Management	164
Entity Relationship Diagram	164
LMA Terminology	165
Installing the LMA	166
Configuring the LMA	167
Associating a Package with Your LMO	168
License Defaults	168
Using the LMA	169
About Leads	169
About Packages	169
About Package Versions	171
About Licenses	172
Integrating with Sales and Marketing	175
Best Practices	176
License Management App FAQ	177
Troubleshooting	177
Chapter 9: Provide a Free Trial	179
Why Use Trialforce?	180
Trialforce	180
Set Up Trialforce	182
Link a Package with Your License Management Organization	182
Request a Trialforce Management Organization	183
Setting up Custom Branding for Trialforce	183
Create a Trialforce Source Organization	185

Contents

Create a Trialforce Template	186
Link a Trialforce Template to the AppExchange	186
Submit a Trialforce Template for Security Review	186
Provide a Free Trial on the AppExchange	187
Provide a Free Trial on the AppExchange Using Trialforce	187
Provide a Test Drive on the AppExchange	188
Provide a Free Trial on the AppExchange When Your Offering Is Installed	188
Provide a Free Trial on Your Website	188
Requesting a Signup Form for Trialforce	189
Link a Trialforce Template to the Sign-Up Form	189
Customizing the HTML Registration Form	190
Provisioning New Trial Organizations	190
Modify the Trial for an Upgrade	191
Trialforce Best Practices	192
Creating Signups using the API	192
Trialforce FAQ	203
Chapter 10: Supporting Your Customers	205
Subscriber Support Console	206
Viewing Subscriber Details	206
Requesting Login Access	206
Logging in to Subscriber Organizations	207
Troubleshooting in Subscriber Organizations	207
Usage Metrics	208
Setting up Usage Metrics	208
Accessing Usage Metrics Data	209
MetricsDataFile	210
Usage Metrics Visualization	212
Chapter 11: Upgrading Your App	215
About Package Versions	217
Creating and Uploading Patches	217
Working with Patch Versions	218
Versioning Apex Code	220
Apex Deprecation Effects for Subscribers	221
Publishing Upgrades to Managed Packages	222
Deleting Components in Managed Packages	223
Viewing Deleted Components	224
Modifying Custom Fields after a Package is Released	225
Managing Versions	226
Pushing an Upgrade	226
About Push Upgrades	226
Push Upgrade Best Practices	227
Assigning Access to New Components and Fields	228

Contents

Sample Post Install Script for a Push Upgrade 229

Known Limitations for Push Upgrade 230

Scheduling Push Upgrades 231

Viewing Push Upgrade Details 232

Viewing an Organization's Upgrade History 233

APPENDICES 235

Appendix A: ISVforce User License Comparison 235

Appendix B: OEM User License Comparison 239

GLOSSARY 243

INDEX 246

CHAPTER 1 Introduction

In this chapter ...

- [Resources for Partners](#)
- [Roles in the Application Lifecycle](#)
- [How to Sign Up for Test Environments](#)

The *ISVforce Guide* is written for independent software vendors (ISVs) who want to build and sell applications using the Force.com platform. This guide is organized by the following chapters:

- [Quick Start](#)—Start here to acquire and configure all of the environments you need in order to build and sell apps.
- [Designing and Building Your App](#)—Before you start development, it's important to know how all the pieces fit together. This chapter covers architectural decisions to consider before development.
- [Packaging and Testing Your App](#)—This chapter also covers specifics about developing and testing packaged apps.
- [Passing the Security Review](#)—Learn about security best practices and plan for security review.
- [Publish Your Offering on the AppExchange](#)—List your app on the AppExchange marketplace.
- [Managing Orders](#)—Use the Channel Order App to create, manage, and submit orders to the Partner Operations team.
- [Managing Licenses](#)—Use the License Management App to manage your customer and app licenses.
- [Providing a Free Trial](#)—Create a free trial to help you sell your app to non-Salesforce customers.
- [Supporting Your Customers](#)—Give your customers technical support for the installation and use of your app.
- [Upgrading Your App](#)—When it's time to upgrade your packaged app, you can push minor patches or create major releases.

Resources for Partners

The Partner Community, at <https://partners.salesforce.com>, is the primary resource for all ISVs. To get started, we recommend visiting the [Education](#) page, your one-stop shop for all ISV content. In addition, you can use the Partner Community to:

- Collaborate with other partners and salesforce.com using our Chatter community.
- Stay up-to-date on news and events related to the Salesforce Partner Program.
- Log cases for access to partner-specific features and customer support.
- Use enhanced search, integrated with the Success Community, to quickly find relevant resources.
- Browse the Salesforce Partner Online Training catalog and sign up for courses.

The Partner Community is self-service—the first person to register your partnership becomes your designated administrator and manages the creation of additional users for your company. You can change or add administrators, as required.

Roles in the Application Lifecycle

This guide covers the entire lifecycle of a package application, so some of the topics might not be relevant to you. The following list has topic suggestions by role.

An application *architect*

The application architect determines the scope of the application and the internal structures that support it. Architects need to know details about the underlying Force.com platform that will determine not only the application's use, but which editions it supports, how it's installed, configured, and upgraded. Architects need to be familiar with this entire guide, but especially the following chapters:

- [Designing and Building Your App](#) on page 17
- [Passing the Security Review](#) on page 114

A *developer* creates, packages, and uploads an app

A developer, or often a team of developers, create an app, package it, and upload it to the AppExchange. Developers also update the app with bug fixes and new features. As a developer, you'll want to see the following chapters:

- [Designing and Building Your App](#) on page 17
- [Packaging and Testing Your App](#) on page 93
- [Developing App Documentation](#) on page 51
- [Upgrading Your App](#) on page 215

A *publisher* distributes, sells, and supports the app

The publisher of an app is the person or company who has a profile and one or more listings for the app on AppExchange. Publisher listings contain a link to an app they have uploaded to AppExchange, or to a third-party website. Publishers also set default license settings. As a publisher, you'll want to see the following chapters:

- [Publish Your Offering on the AppExchange](#) on page 123
- [Provide a Free Trial](#) on page 179
- [Supporting Your Customers](#) on page 205

An *administrator* installs the app

An administrator, or *admin*, downloads your app from AppExchange and installs it into their organization. Admins might also customize the app to further suit their business needs. See the following topic to learn how admins will interact with your app.

- [Installing a Package](#) on page 103

How to Sign Up for Test Environments

To sign up for test environments (organizations), use the Environment Hub.



Note: If you're a new Salesforce user, log in to the organization that you received when you signed up for the Partner Program. The Environment Hub is enabled in this organization by default. If you're an existing Salesforce user and are using a different organization to manage development, log a case in the [Partner Community](#) to enable the Environment Hub.

1. Log in to the organization where Environment Hub is enabled.
2. Select the Environment Hub tab, then click **Create Organization**.
3. In the Purpose drop-down list, select **Test/Demo**.
4. In the Edition drop-down list, choose the edition you want to test against.
5. Fill in the remaining required fields. Optionally, set up My Domain.
6. Agree to the terms and then click **Create**.
7. You'll receive an email that will prompt you to log in and change your password. Click the link, change your password, and create a password question and answer.

CHAPTER 2 ISVforce Quick Start

In this chapter ...

- Tutorial #1: Sign Up for AppExchange
- Tutorial #2: Developing Your App
- Tutorial #3: Publishing and Licensing
- Tutorial #4: Updating Your App

This quick start is meant to take you through the application lifecycle as quickly as possible. Upon completion, you'll have everything you need to develop and publish a packaged application.



Note: You must be an ISVforce/OEM partner to complete all steps in this quick start, as it covers some features only available to eligible partners.

How is the Quick Start Organized?

The quick start is broken up into four tutorials, which are meant to be completed in order. Because some of the steps require an automated email response, the time to complete the steps can vary. However, you can stop at any step and pick up where you left off.

- Tutorial #1 takes you through the process of signing up for the Salesforce ISV Partner Program and getting all of the organizations (environments) you'll need.
- Tutorial #2 walks you through creating a simple Hello World application.
- Tutorial #3 helps you publish and manage your app.
- Tutorial #4 tells you how to update your app for major and minor releases.

Tell Me More....

At the end of each step, there is an optional Tell Me More section. If you like to do things quickly, move on to the next step. However, if you're a smell-the-roses type, there's a lot of useful information here.

- For a list of useful terms, see the [Glossary](#) on page 243.
- To learn more about Force.com and to access a rich set of resources, visit Salesforce Developers at <https://developer.salesforce.com>.
- For a gentle introduction to developing on Force.com, see the Force.com Workbook at https://developer.salesforce.com/page/Force.com_workbook.

Tutorial #1: Sign Up for AppExchange

In this tutorial, you set up the tools you need to develop, sell, and support apps and components built on the Force.com platform. You start by signing up for the Partner Program. You then have access to the Partner Community, which allows you to view helpful resources, create support cases, and collaborate with other partners and Salesforce. The Partner Community is also the best source for news and events about the Partner Program. In addition, you can access the Environment Hub, where you can create development and test organizations.

If you're familiar with Salesforce, you know that an organization is a cloud unto itself. If you're new to Salesforce, think of your organization as a separate environment for developing, testing, and publishing your offering.

Step 1: Sign Up for the Partner Program

The first step is to sign up for the Partner Program.

1. In your browser, go to <https://partners.salesforce.com> and click **Join Now**.



Note: The signup process varies according to the region or country. Follow the instructions presented.

2. Fill in the fields about you and your company.
3. Select the first option: **Independent Software Vendor (ISV)**.
4. Click **Submit Registration**.

In a moment, you'll receive a confirmation, followed by an email welcoming you to the Partner Program and including login credentials.

Congratulations, you're now part of the Salesforce ISV Partner Program! Click the link to the Partner Community (<https://partners.salesforce.com>) and log in. Bookmark this page. You'll be using it a lot.

Step 2: Create a Development and Test Environment

To build and sell on the Force.com platform, you need different environments for different tasks. We call these environments *organizations*, or *orgs* for short. You use the Environment Hub to create these orgs. The first org you need is the Partner Developer Edition, which is where you develop and package your offering. If you already have a Developer Edition org, we recommend signing up for the Partner Developer Edition org because you can have more data storage, licenses, and users.



Note: If you're a new Salesforce user, log in to the organization that you received when you signed up for the Partner Program. The Environment Hub is enabled in this organization by default. If you're an existing Salesforce user and are using a different organization to manage development, log a case in the [Partner Community](#) to enable the Environment Hub.

1. Log in to the organization where the Environment Hub is enabled, usually your partner business org.
2. Click the **Environment Hub** tab, and then click **Create Organization**.
3. In the Purpose drop-down list, select **Development**. For simplicity, we refer to this as your *dev org*.
4. Fill in the required fields. Optionally, set up My Domain.
5. Agree to the terms and then click **Create**.
6. In the Purpose drop-down list, select **Test/Demo** and **Partner Enterprise** for the org edition. This process creates a test org, where you test the app or component that you are developing.
7. Shortly, you'll receive emails that prompt you to log in and change your password for your dev and test orgs.

Tell Me More...

The Environment Hub has several types of test orgs available, because different editions of Salesforce have different features. If you plan to distribute your app or component to a particular edition, you want to test your offering and make sure that it works there. Although that's beyond the scope of this quick start. For more information, see [Architectural Considerations for Group and Professional Editions](#) on page 58.

Step 3: Get a Business Org

In the previous step, you created orgs for developing and testing your offering. To manage sales and distribution, you need another organization. In this step, you log a case in the Partner Community to have a partner business org provisioned for you. Your business org contains the apps that you use to manage sales and distribution, including the License Management App (LMA) and Channel Order App (COA).

1. In the Partner Community, under the Support tab, click **New Case**.
2. For the first category, choose **Orders and Contracts**.
3. For the second category, choose **Request ISV Business Org**.
4. In the Subject field, enter *Need Partner Business Org*.
5. In the Description field, tell us if you have an existing org or if you need a new one. If you have an existing Salesforce org, enter the Org ID in the Description field to add two more CRM licenses to your org. If you don't have an existing org, we provide a new one for you. In either case, make sure to enter your business address and then click **Submit Case**.



Note: It can take 24–48 hours for your case to be closed. You can check the status of your case at any time under the Support tab of the Partner Community.

6. You'll receive an email prompting you to log in and change your password. Do that, and then bookmark the page.

Step 4: Edit Your Publisher Profile

In this step, you log in to the Partner Community and provide information about your company. We display some of this information on AppExchange listings to help customers get to your business.

1. Log in to the Partner Community using the username and password of your business org.
2. On the Publishing page, click **Company Info**.
3. Fill out the information in the Provider Profile, and then click **Save**.

Sign-up Summary

In this first tutorial, you signed up for the Partner Program and all the organizations you need to develop, test, and sell your offering. Let's review what you signed up for and the purpose of each.

Partner Program

The Partner Program gives you access to the Partner Community, where you can get help and training information, log cases for support issues, and collaborate with other partners. You also get access to the Environment Hub, which lets you create and manage new test and development orgs.

Partner Developer Edition

Also known as your *dev org*, this is where you develop your offering and eventually package it for distribution.

Test Organization

Also known as your *test org*, this is where you install and test your offering.

Partner Business Organization

This is where you license and manage your offering.

Tutorial #2: Developing Your App

In this tutorial you'll create a very simple “Hello World” application. It won't do much, but it's enough to understand where development takes place in the lifecycle of a packaged application.

Step 1: Create an App

In this step you're going to create an app that contains a page, and a tab to display the page.

1. In your browser, log in to your Partner Developer Edition organization. Hereafter we'll call this your “dev org”.
2. From Setup, click **Develop > Pages**.
3. In the Visualforce list, click **New**.
4. In the `Label` field enter *Greeting*.
5. In the Visualforce Markup area, replace the contents of the `<h1>` tag with *Hello World*.

Visualforce Page Editor

The screenshot shows the Visualforce Page Editor interface. At the top, the page is titled "Greeting". Below the title, there are "Page Edit" buttons: "Save", "Quick Save", and "Cancel". The "Page Information" section contains fields for "Label" (Greeting), "Name" (Greeting), and "Description". Below this, there are tabs for "Visualforce Markup" and "Version Settings". The "Visualforce Markup" tab is active, showing the following code:

```
1 <apex:page >
2 <!-- Begin Default Content REMOVE THIS -->
3 <h1>Hello World!</h1>
4 This is your new Page
5 <!-- End Default Content REMOVE THIS -->
6 </apex:page>
```

6. Click **Save**.

Now you'll associate the page with a tab.

1. In the sidebar menu, click **Create > Tabs**.
2. In the Visualforce Tabs list, click **New**.
3. In the New Visualforce Tab wizard, click the drop-down box and select the Hello World page you just created.
4. For the Tab Label, enter *Hello*.
5. Click the Tab Style field and choose any icon to represent your tab.
6. Click **Next**, then **Next** again, and **Save** on the final page.

Now you'll create a new app that contains your tab and page.

1. In the sidebar menu, click **Create > Apps**.

2. Click **New**.
3. In the App Label field enter Hello World and then click **Next** and **Next** again on the following page.
4. On the Choose the Tabs page, scroll to the bottom of the Available Tabs list, find your Hello tab, and add it to the Selected Tabs list. Click **Next**.
5. Select the **Visible** checkbox to make this app visible to all profiles and then click **Save**.

Tell Me More....

If it seems like you just created a page within a container, within another container, you did. And you're about to put all of that in another container! What's with all these containers and what do they do?

- A *tab* is a container for things you want to display on the same page, such as a chart, a table, or the Visualforce page you created.
- An *app* is a container for tabs that appear next to each other. When you create an app, it's available in the app picker in the upper right hand corner of the screen.
- A *package* is a container for things you upload to the AppExchange. Usually a package contains an app your customers can install in their org, but you can also upload packages that extend existing apps. You haven't created a package yet, you'll do that in the next step.

Step 2: Package Your App

In this step you'll package the app so you can distribute it on the AppExchange. A package is simply a container for components. In this case it's your app, tab, and page.

1. In the sidebar menu, click **Create > Packages**, and then click **New**.
2. In the Package Name field enter *Hello World* and then click **Save**.
3. On the Package Detail page click **Add Components**.
4. Select your Hello World app and then click **Add to Package**.

Tell Me More....

When you clicked **Add to Package**, did you notice that your Hello tab and Greeting page were automatically added to the package? When you create a package, the framework automatically detects dependent components and adds them to the package.

Step 3: Assign a Namespace

In this step you'll choose a unique identifier called a namespace. A namespace differentiates your components from other components and allows you to do things such as upgrade the app after it's been installed. Choose your namespace carefully as it can't be changed later.

1. In the sidebar menu, click **Create > Packages**.
2. In the Developer Settings list, click **Edit** and on the following page click **Continue**.
3. In the Namespace Prefix field, enter a 1-15 character alphanumeric ID and then click **Check Availability**. Repeat this step until you have a unique namespace.
4. In the Package to be managed field choose your Hello World package and then click **Review Your Selections**.
5. Review the information on the page and then click **Save**.

Tell Me More....

Within the underlying code, your namespace is prepended to all components that are packaged from your dev org. This allows your package and its contents to be distinguished from those of other developers, and ensures your exclusive control of all packaged components.

Step 4: Upload a Beta

Before you upload a production version of your app, it's a common practice to upload a beta version for testing.

1. In the sidebar menu, click **Create > Packages**.
2. On the Packages page, click your **Hello World** package and then click **Upload**.
3. On the Upload Package page, enter a version name and number.
4. For the Release Type, make sure to choose **Managed — Beta**.
5. Scroll to the bottom and click **Upload**. It may take a moment for the upload to complete.

Congratulations, you've uploaded an app to the AppExchange! Your app isn't available to the public, but you can access it through an install link. You'll do that in the next step.

Tell Me More....

The purpose of a beta is for testing only. Therefore, a beta can only be installed in a test org, Developer Edition, or sandbox (more on that later). Next you'll install the beta in the test org you created in Step 2: Create a Development and Test Environment.

Step 5: Install and Test the Beta

Installing the beta is easy, just click the link and provide the username and password you use for your test org.

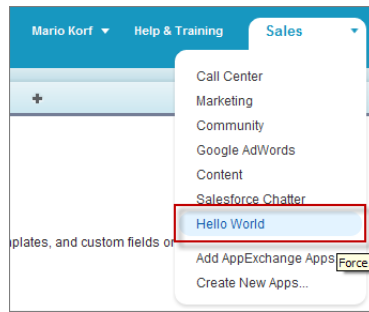
1. Click the Installation URL now.

Installation URL Link

Package Version Detail		Change Password	Deprecate
Package Name	Hello World		
Version Name	Spring 2011		
Version Number	1.0 (Beta 1)		
Language	English		
Installation URL	https://login.salesforce.com/?startURL=%2Fpackage%2FinstallPackage.apexp%3Fp0%3D04A00000000lxSk		
Description		Change Password	Deprecate

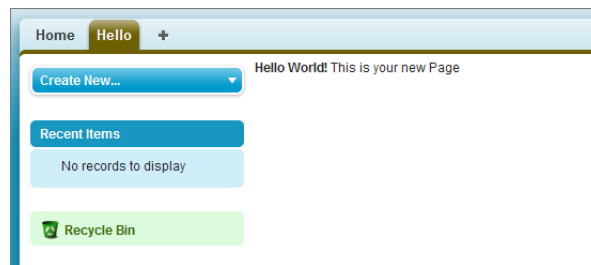
2. On the login page, enter the Username and Password of your test org.
3. On the Package Installation Details page, click **Continue**.
4. Click **Next**.
5. On the Security Level page, **Grant access to all users** and click **Next**.
6. Click **Install**.
7. Once the installation completes, you can select your app from the app picker in the upper right corner.

Hello World App



8. You should see your Hello tab, and the greeting text on your page.

Hello World Tab and Page



At this point you would normally test the application and make sure it works as designed. Your app installs easily and displays what you want, so let's move on.

Tell Me More....

Beta packages can also be installed in sandboxes. A sandbox is a replica of your customer's org that allows them to develop, test, or install apps, and verify the changes they want to commit. None of the orgs you've signed up for in this workbook have a sandbox, but if you have a sandbox in another org and want to install your app in it, you must replace the initial portion of the **Installation URL** with `http://test.salesforce.com`.

Development Summary

Congratulations, you just completed an essential part of the software development lifecycle! Further changes to your app will follow the same procedure:

1. Modify the existing app in your dev org.
2. Package the app.
3. Upload as a beta package.
4. Install the beta in a test org.
5. Test the installed app.

Tutorial #3: Publishing and Licensing

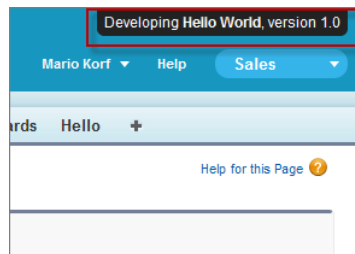
Imagine you've been through a few development cycles with your beta and you're ready to publish a public app. The next step is to upload a production app, or what we call a *managed released* version of your app. Then you can create a listing so that other people can find your app and know what it does. Finally, you want to connect your app to your business org so you manage the licenses for people that install your app.

Step 1: Uploading to the AppExchange

This step will seem familiar, it's similar to uploading a beta.

1. If you've been following along non-stop, you're probably still logged in to your test org. Go ahead and log in to your dev org now.
2. Notice in the upper right corner there's a link that says **Developing Hello World, version 1.0**. Click that link to go directly to the Package Detail page.

Developing Hello World, version 1.0

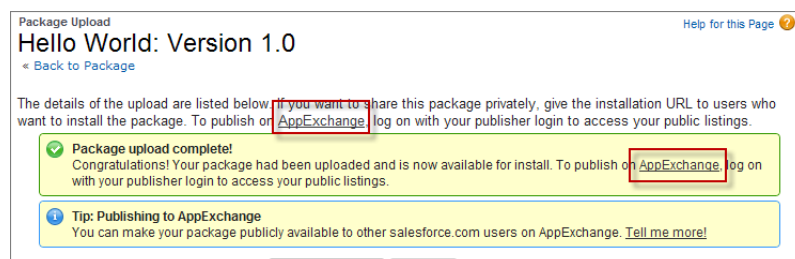


3. On the Package Detail page, click **Upload**.
4. For the Release Type, choose **Managed — Released**.
5. Scroll to the bottom and click **Upload**.
6. Click **OK** on the popup.

Step 2: Create an AppExchange Listing

In this step, you create an AppExchange listing, which is the primary way customers discover apps, components, and services to enhance their Salesforce experience.

1. After your package uploads, click the link to publish on the AppExchange. You are directed to the Listings tab on the Publishing page.



2. If prompted, enter your login credentials for the Partner Community.

3. Read and agree to the terms and conditions, and then click **I Agree**.
4. The first question asks if you've already listed on the AppExchange. You did that in Tutorial 1, [Step 4: Edit Your Publisher Profile](#) on page 6, so select **Yes** and click **Continue**.
5. Click **Link New Organization**.
6. You're prompted for your username and password. Enter the values for your development org.
7. Click the **Publishing** tab.
8. Click **New Listing**.
9. Enter a listing title, such as *Hello World App by <your name>*. Adding your name helps ensure that your listing title is unique.
10. Choose **App**, and then click **Save & Next** to open the AppExchange publishing console.
11. On the Text tab, fill in the required fields, and then click **Save & Next** again.

Tell Me More...

Don't be concerned with making your listing perfect, because it's not public yet, and you can change the listing at any time.

Step 3: Complete the AppExchange Listing


Many customers like to see and experience a product before they decide to purchase. We give you several ways to show off your app or component in an AppExchange listing. For example, you can add screenshots and videos to draw attention to key features, or add white papers to help demonstrate business value. You can also let customers try your offering in their own organizations or set up a test environment that you've customized.

1. If you're not already there, click the Media tab in the AppExchange publishing console.
2. Add an app logo, tile image, and screenshot. Because your listing isn't used outside of this tutorial, use any image file that you have available.
3. Click the **App** tab, and then select **An app that includes a package (entirely or in part)**.
4. Click **Select Package** and choose the package that you uploaded in the previous step.
5. For the installation method, select **Directly from the AppExchange**.
6. Choose whether you want the app to be installed for every user in the customer's organization or just system administrators. For this tutorial, either option is fine.
7. For app specifications, select editions and languages. For this tutorial, you can select any available edition and language.
8. Click **Save & Next**.
9. Click **Save & Next** twice, because you don't want to configure a free trial or set up lead collection for this app.
10. For pricing, select **Free**. Use the default values for all other fields.
11. Agree to the terms and conditions, and then click **Save**.

Congratulations—you've completed your first listing! Like everything else you've done so far, you can go back and change it later if you want.

Step 4: Manage Licenses for Your App

The License Management App (LMA) helps you manage sales, licensing, and support for your offering. The LMA comes preinstalled in your business organization. In this step, you connect your app to the LMA.

 **Note:** This feature is available to eligible partners. For more information on the Partner Program, including eligibility requirements, visit www.salesforce.com/partners.

1. If you haven't done so already, log in to the Partner Community.
2. On the Publishing page, click the **Packages** tab.
3. Find the package that you want to link, and then click **Manage Licenses**.
4. Click **Register**.
5. Enter the login credentials of your partner business org, and then click **Submit**.
6. For the default license type, choose free trial.
7. Enter a trial length in days.
8. For the number of seats, choose the site-wide license.
9. Click **Save**.

It can take up to 30 minutes for your app to be connected to the LMA. Take a break; you've earned it!

Publishing and Licensing Summary

In this tutorial, you uploaded your managed-released app to the AppExchange and created a listing for your app. You also linked your app to the License Management App, available in your business organization. You can use the LMA to manage and renew licenses and to set default license settings. For example, you can license your app as a free trial that expires after a specified number of days. For more information, see [Managing Licenses](#) on page 163.

Right now your app has a private listing on the AppExchange that you can share with potential customers, but the public doesn't see it unless they have the link. Before you can list the app publicly, you'll need to pass a security review, which is beyond the scope of this quick start. For more information, see [Security Review Steps](#) on page 115.

Tutorial #4: Updating Your App

If you're familiar with Salesforce, you know we do weekly patch releases to fix bugs, and a few times a year we have a major release to introduce new features. As an ISV, you can do the same thing by delivering a patch release to fix bugs and a major release for new features.

- For new features, the process is the same as you've experienced. You start by modifying your app, package it, upload a beta, test the beta, and then upload a managed-released version. Major releases increment the version to the next whole number, from 1.0 to 2.0, for example, and minor releases to the first dot from 1.0 to 1.1. There are no hard rules for what constitutes a major or minor release. That's up to you.
- For bug fixes, the process is slightly different. You start by creating a patch org, a special environment which has limited functionality and can only be used to develop a patch for a specific package. After you upload the patch, you have the option of pushing the patch to your customers, so they get your bug fixes the next time they log in. Minor releases increment the version number to the second decimal, from 1.0 to 1.0.1, for example.
- Major or minor releases must be installed by customers (pulled). However, you can push patch releases directly to customer orgs. This feature is only available to registered ISVforce/OEM partners. For more information on the Partner Program, including eligibility requirements, please visit us at www.salesforce.com/partners.

Since the process for developing a major release is already familiar, let's do a patch release and then deliver it by pushing the patch to our customers.

Step 1: Creating a Patch Organization

In order to create a patch, you need to generate a new patch development organization.

To create a patch version:

1. From Setup, click **Create > Packages**.
2. Click the name of your managed package.
3. Click the Patch Organization tab and then click **New**.
4. Select the package version that you want to create a patch for in the Patching Major Release drop-down list. The release type must be Managed - Released.
5. Enter a `Username` for a login to your patch organization.
6. Enter an `Email Address` associated with your login.
7. Click **Save**.



Note: If you ever lose your login information, click **Reset** on the package detail page under Patch Development Organizations to reset the login to your patch development organization.

In a moment you'll receive an email with your login credentials. After you've logged in and changed your password, proceed to the next step.

Tell Me More....

Development in a patch development organization is restricted. The following is a list of caveats:

- New package components can't be added.
- Existing package components can't be deleted.
- API and dynamic Apex access controls can't change for the package.
- No deprecation of any Apex code.
- No new Apex class relationships, such as `extends`, can be added.
- No new Apex access modifiers, such as `virtual` or `global`, can be added.
- No new Web services can be added.
- No new feature dependencies can be added.

Step 2: Developing a Patch

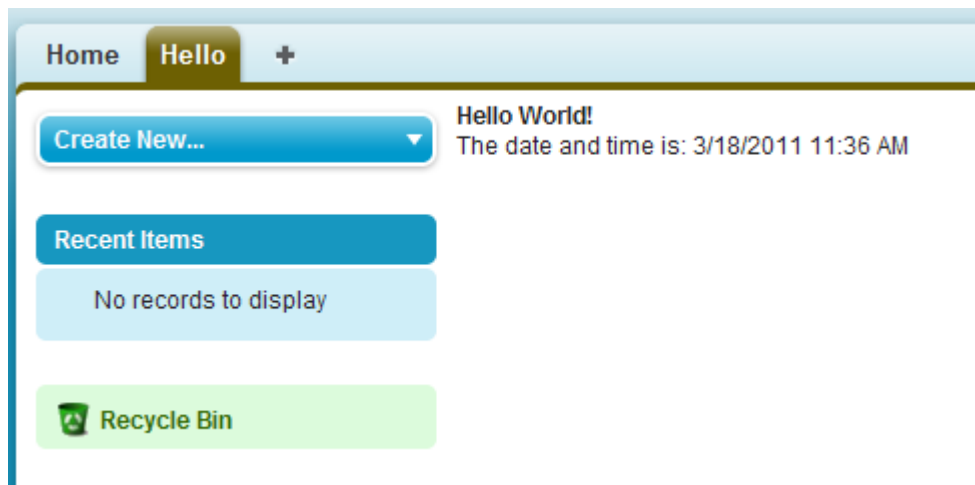
We're going to make a simple change to your app. Instead of displaying just Hello World, you'll add today's date.

1. In your patch org, from Setup, click **Create > Packages** and click your **Hello World** package.
2. In the list of Package Components, click your **Greeting** page.
3. Click **Edit**.
4. Right after the closing `</h1>` tag, enter the following:

```
<br/>
<apex:outputText value="The date and time is: {!NOW()}" />
```

5. Click **Save**.
6. To see the output, click the **Hello** tab and you'll notice that today's time and date are displayed.

Display the date and time



That's as much as we need to do in this patch. Let's move on.

Tell Me More....

The `!NOW` function returns the date in a standard format. There are many more built-in functions and ways to format the output. For more information, see the [Visualforce Developer's Guide](#).

Step 3: Uploading the Patch

Typically the next step is to upload a beta patch and install that in a test organization. Since this is very similar to Step 4: Upload a Beta and Step 5: Install and Test the Beta, that you completed in Tutorial #2: Developing Your App, we won't make you do that again.

1. In your patch org, from Setup, click **Create** > **Packages** and click your **Hello World** package.
2. On the Upload Package page, click **Upload**.
3. Enter a version name, such as today's date.
4. Notice that the `Version Number` has had its `patchNumber` incremented.
5. Select **Managed — Released**.
6. Optionally, enter and confirm a password to share the package privately with anyone who has the password. Don't enter a password if you want to make the package available to anyone on AppExchange and share your package publicly.
7. Salesforce automatically selects the requirements it finds. In addition, select any other required components from the `Package Requirements` and `Object Requirements` sections to notify installers of any requirements for this package.
8. Click **Upload**.

Congratulations, you've uploaded a patch release. You'll want to share that patch with others, and you'll do that next.

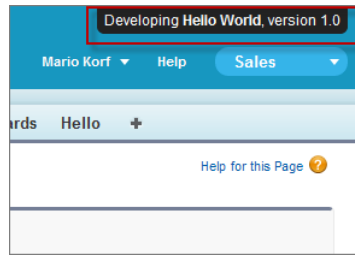
Step 4: Installing or Pushing a Patch

There are two ways to deliver a patch, you can have your customers install it, or you can *push* it to them. Push upgrades happen automatically, that is, the next time your customer logs in, they have the updates. Let's try that.

1. Log in to your dev org.

2. In the upper right corner, click **Developing Hello World, version 1.0**.

Developing Hello World, version 1.0



3. On the Package Detail page, click **Push Upgrades**.
4. Click **Schedule Push Upgrades**.
5. From the Patch Version drop-down list, select the patch version to push.
6. In the **Scheduled Start Date** field, enter today's date.
7. In the Select Target Organizations section, select your test org.
8. Click **Schedule**.

And you've done it! You pushed a patch release to your subscriber so that they automatically get your updates. You should verify that your customers received the patch to ensure it was installed successfully.

Tell Me More....

Beta versions aren't eligible for push upgrades. You must uninstall a beta and then install a new one.

Updating Your App Summary

In this tutorial you learned how to update your app in a patch org and push that update to your customers. You started by creating a patch organization that was specific to a released package version. Then you modified your app, uploaded it, and scheduled the push upgrade to your customers.

Congratulations, you're done! Or have you really just begun? You can modify your existing app to be anything you want it to be, or create a new dev org in the Environment Hub and build another app. You can use the same sales and test orgs and everything else you've configured to publish and manage many more apps. You're on your way to ISVforce success!

CHAPTER 3 Designing and Building Your App

In this chapter ...

- [Overview of Packages](#)
- [Components Available in Managed Packages](#)
- [About API and Dynamic Apex Access in Packages](#)
- [Architectural Considerations for Group and Professional Editions](#)
- [Connected Apps Overview](#)
- [Environment Hub](#)

This section contains important concepts and architectural decisions to consider before you start development, such as:

- [Understanding Managed and Unmanaged Packages](#)
- [Components Available for Packaging](#)
- [Special Behavior of Components in Packages](#)
- [Limits for Group and Professional Editions](#)
- [Understanding Dependencies](#)
- [Working With External Services](#)
- [Protecting Your Intellectual Property](#)
- [Working with Connected Apps](#)

Overview of Packages

A *package* is a container for something as small as an individual component or as large as a set of related apps. After creating a package, you can distribute it to other Salesforce users and organizations, including those outside your company.

Packages come in two forms—unmanaged and managed:

Unmanaged packages

Unmanaged packages are typically used to distribute open-source projects or application templates to provide developers with the basic building blocks for an application. Once the components are installed from an unmanaged package, the components can be edited in the organization they are installed in. The developer who created and uploaded the unmanaged package has no control over the installed components, and can't change or upgrade them. Unmanaged packages should not be used to migrate components from a sandbox to production organization. Instead, use Change Sets.

Managed packages

Managed packages are typically used by Salesforce partners to distribute and sell applications to customers. These packages must be created from a Developer Edition organization. Using the AppExchange and the License Management Application (LMA), developers can sell and manage user-based licenses to the app. Managed packages are also fully upgradeable. To ensure seamless upgrades, certain destructive changes, like removing objects or fields, can not be performed.

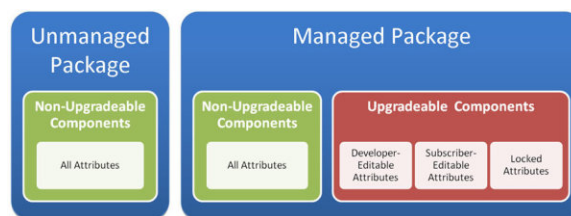
Managed packages also offer the following benefits:

- Intellectual property protection for Apex
- Built-in versioning support for API accessible components
- The ability to branch and patch a previous version
- The ability to seamlessly push patch updates to subscribers
- Unique naming of all components to ensure conflict-free installs

Packages consist of one or more Salesforce components, which, in turn, consist of one or more attributes. Components and their attributes behave differently in managed and unmanaged packages.

The following definitions illustrate these concepts:

Unmanaged and Managed Packages



Components

A *component* is one constituent part of a package. It defines an item, such as a custom object or a custom field. You can combine components in a package to produce powerful features or applications. In an unmanaged package, components are not upgradeable. In a managed package, some components can be upgraded while others can't.





Attributes

An *attribute* is a field on a component, such as the name of an email template or the `Allow Reports` checkbox on a custom object. On a non-upgradeable component in either an unmanaged or managed package, attributes are editable by both the developer (the one who created the package) and the subscriber (the one who installed the package). On an upgradeable component in a managed package, some attributes can be edited by the developer, some can be edited by the subscriber, and some are locked, meaning they can't be edited by either the developer or subscriber.

Planning the Release of Managed Packages

Releasing an AppExchange package is similar to releasing any other program in software development. You may want to roll it out in iterations to ensure each component functions as planned. You may even have beta testers who have offered to install an early version of your package and provide feedback.

Once you release a package by publishing it on AppExchange, anyone can install it. So, plan your release carefully. Review the states defined below to familiarize yourself with the release process. Salesforce automatically applies the appropriate state to your package and components depending on the upload settings you choose and where it is in the release process.

State	Description
Unmanaged	The package has not been converted into a managed package or the component has not been added to a managed package. Note that a component that is “Managed - Beta” can become “Unmanaged” if it is removed from a managed package. All packages are unmanaged unless otherwise indicated by one of the managed icons below.
 Managed - Beta	<p>The package or component was created in the current Salesforce organization and is managed, but it is not released because of one of these reasons:</p> <ul style="list-style-type: none"> • It has not been uploaded. • It has been uploaded with Managed - Beta option selected. This option prevents it from being published, publicly available on AppExchange. The developer can still edit any component but the installer may not be able to depending on which components were packaged. <p> Note: Don't install a Managed - Beta package over a Managed - Released package. If you do, the package is no longer upgradeable and your only option is to uninstall and reinstall it.</p>
 Managed - Released	<p>The package or component was created in the current Salesforce organization and is managed. It is also uploaded with the Managed - Released option selected, indicating that it can be published on AppExchange and is publicly available. Note that once you have moved a package to this state, some properties of the components are no longer editable for both the developer and installer.</p> <p>This type of release is considered a major release on page 217.</p>
Patch	<p>If you need to provide a minor upgrade to a managed package, consider creating a patch instead of a new major release. A patch enables a developer to change the functionality of existing components in a managed package, while ensuring that subscribers experience no visible changes to the package.</p> <p>This type of release is considered a patch release on page 217.</p>
 Managed - Installed	The package or component was installed from another Salesforce organization but is managed.


A developer can refine the functionality in a managed package over time, uploading and releasing new versions as the requirements evolve. This might involve redesigning some of the components in the managed package. Developers can delete some, but not all, types of components in a Managed - Released package when upgrading it. For details, see [Deleting Components in Managed Packages](#) on page 223.

Creating and Editing a Package

An app can contain many different components and you can create, upload, and register your apps on your own timeline. To group components in a container to upload to Force.com AppExchange, create a package and add components to the package. A package is the container for the app that you must use to upload all its components together.

 **Tip:** Before you begin, determine if you want to create and upload a [managed or unmanaged package](#).

To create a new package:

1. From Setup, click **Create > Packages**.
 2. Click **New**.
 3. Enter a name for your package. This does not have to be the same name that appears on AppExchange.
 4. From the drop-down menu, select the default language of all component labels in the package.
 5. Optionally, choose a custom link from the `Configure Custom Link` field to display configuration information to installers of your app. You can select a predefined custom link to a URL or s-control that you have created for your home page layouts; see the [Configure Option](#) on page 51. The custom link displays as a **Configure** link within Salesforce on the Force.com AppExchange Downloads page and app detail page of the installer's organization.
 6. Optionally, in the `Notify on Apex Error` field, enter the username of the person who should receive an email notification if an exception occurs in Apex that is not caught by the Apex code. If you do not specify a username, all uncaught exceptions generate an email notification that is sent to Salesforce. This is only available for managed packages. For more information, see *"Handling Apex Exceptions in Managed Packages" in the Salesforce Help*.
-  **Note:** Apex can only be packaged from Developer, Enterprise, Unlimited, and Performance Edition organizations.
7. Optionally, enter a description that describes the package. You will have a chance to change this description before you upload it to AppExchange.
 8. Optionally, specify a post install script. This is an Apex script that runs in the subscriber organization after the package is installed or upgraded. For more information, see [Running Apex on Package Install/Upgrade](#).
 9. Optionally, specify an uninstall script. This is an Apex script that runs in the subscriber organization after the package is uninstalled. For more information, see [Running Apex on Package Uninstall](#).
 10. Click **Save**.

EDITIONS

Available in:

- Group
- Professional
- Enterprise
- Performance
- Unlimited
- Developer

USER PERMISSIONS

To create packages:

- "Create AppExchange Packages"

Developing and Distributing Unmanaged Packages

Unmanaged packages are traditionally used for distributing open-source projects to developers, or as a one time drop of applications that require customization after installation. You should never use unmanaged packages for sandbox to production migration. Instead, use the Force.com IDE or the Force.com Migration Tool. If you're using Enterprise, Unlimited, or Performance Edition, see "Change Sets" in the Salesforce Help.

SEE ALSO:

[Components Available in Unmanaged Packages](#)

Creating and Uploading an Unmanaged Package


Use the following procedure to upload an unmanaged package.

1. Create the package:

- a. From Setup, click **Create > Packages**.
- b. Click **New**.
- c. Fill in the details of the package.
- d. Click **Save**.


2. Add the necessary components for your app.

- a. Click **Add Components**.
- b. From the drop-down list, choose the type of component.
- c. Select the components you want to add.

 **Note:** Some components cannot be added to Managed - Released packages. For a list of packageable components, see [Components Available in Managed Packages](#) on page 21. Also, S-controls cannot be added to packages with restricted API access.

d. Click Add To Package.

- e. Repeat these steps until you have added all the components you want in your package.

 **Note:** Some related components are automatically included in the package even though they might not display in the Package Components list. For example, when you add a custom object to a package, its custom fields, page layouts, and relationships with standard objects are automatically included. For a complete list of components, see [Components Automatically Added to Packages](#) on page 34.

3. Click Upload.

You will receive an email that includes an installation link when your package has been uploaded successfully. Wait a few moments before clicking the installation link or distributing it to others, as it might take a few minutes for it to become active.

Components Available in Managed Packages

Not all components can be packaged for distribution. If you create an app that uses components that aren't packageable, your subscribers will have to create and configure those components after they install your app. If ease of installation is an important concern for your subscribers, keep the packageable components in mind as you develop.

The following table shows the components that are available in a managed package, and whether or not it is updateable or deletable. The following sections describe the table columns and their values.

Upgradeable

Some components are updated to a newer version when a package is upgraded.

- **No:** The component is not upgraded.
- **Yes:** The component is upgraded.

Subscriber Deletable

A subscriber or installer of a package can delete the component.

- **No:** The subscriber cannot delete the component.
- **Yes:** The subscriber can delete the component.

Developer Deletable

A developer can delete some components after the package is uploaded as Managed - Released. Deleted components are not deleted in the subscriber's organization during a package upgrade. The Protectable attribute contains more details on deleting components.

- **No:** The developer cannot delete a Managed - Released component.
- **Yes:** The developer can delete a Managed - Released component.

Protectable

Developers can mark certain components as protected. Protected components can't be linked to or referenced by components created in a subscriber organization. A developer can delete a protected component in a future release without worrying about failing installations. However, once a component is marked as unprotected and is released globally, the developer can't delete it. When the subscriber upgrades to a version of the package where the component is deleted, the component is removed from the subscriber's organization.

- **No:** The component cannot be marked protected.
- **Yes:** The component can be marked protected.

IP Protection

Certain components automatically include intellectual property protection, such as obfuscating Apex code. The only exceptions are Apex methods declared as global, meaning that the method signatures can be viewed by the subscriber. The information in the components you package and publish might be visible to users on AppExchange. Use caution when adding your code to a custom s-control, formula, Visualforce page, or any other component that you cannot hide in your app.

- **No:** The component does not support intellectual property protection.
- **Yes:** The component supports intellectual property protection.

Component	Upgradeable	Subscriber Deletable	Developer Deletable	Protectable	IP Protection
Action	Yes	No	No	No	No
Reporting Snapshot	No	Yes	Yes	No	No
Apex Class	Yes	No	Yes (if not set to global access)	No	Yes
Apex Sharing Reason	Yes	No	No	No	No
Apex Sharing Recalculation	No	Yes	Yes	No	No
Apex Trigger	Yes	No	Yes	No	Yes
Application	No	Yes	Yes	No	No
Article Type	Yes	No	No	No	No
Call Center	No	Yes	No	No	No
Compact Layout	Yes	No	No	No	No
Connected App	Yes	Yes	Yes	No	No

Component	Upgradeable	Subscriber Deletable	Developer Deletable	Protectable	IP Protection
Custom Button or Link	Yes	Yes*	Yes**	No, except custom links (for Home page only)	No
Custom Field	Yes	Yes*	Yes**	No	No
Custom Label	Yes	No	Yes, if protected	Yes	No
Custom Object	Yes	Yes*	Yes**	No	No
Custom Permission	Yes	No	No	No	No
Custom Report Type	Yes	No	No	No	No
Custom Setting	Yes	Yes*	Yes**	No	Yes
Dashboard	No	Yes	Yes	No	No
Document	No	Yes	Yes	No	No
Email Template	No	Yes	Yes	No	No
External Data Source	Yes	No	No	No	No
Field Set	Yes	Yes*	Yes**	No	No
Lightning Page	Yes	No	No	No	No
Flow	Yes	Yes	No	No	No
Folder	No	Yes	Yes	No	No
Home Page Component	Yes	No	No	No	No
Home Page Layout	No	Yes	Yes	No	No
Letterhead	No	Yes	Yes	No	No
Lightning Application	Yes	No	No	No	No
Lightning Component	Yes	No	No	No	No
Lightning Event	Yes	No	No	No	No
Lightning Interface	Yes	No	No	No	No
List View	No	Yes	Yes	No	No
Named Credential	Yes	No	No	No	No
Page Layout	No	Yes	Yes	No	No

Component	Upgradeable	Subscriber Deletable	Developer Deletable	Protectable	IP Protection
Permission Set	Yes	Yes*	Yes**	No	No
Record Type	Yes	Yes*	Yes**	No	No
Remote Site Setting	No	Yes	Yes	No	No
Report	No	Yes	Yes	No	No
S-Control	Yes	No	No	No	No
Static Resource	Yes	Yes*	Yes**	No	No
Tab	Yes	Yes*	Yes**	No	No
Translation	Yes	No	No	No	No
Validation Rule	Yes	Yes*	Yes**	No	No
Visualforce Component	Yes	Yes***	Yes**	No	Yes
Visualforce Page	Yes	Yes*	Yes**	No	No
Workflow Email Alert	Yes	No	Yes, if protected	Yes	No
Workflow Field Update	Yes	No	Yes, if protected	Yes	No
Workflow Outbound Message	Yes	No	Yes, if protected	Yes	No
Workflow Rule	Yes	No	No	No	No
Workflow Task	Yes	No	Yes, if protected	Yes	No

* If you remove this component type from a new version of your package and a subscriber upgrades, the Administrator (System Administrator) of the subscriber organization can delete the component.

** If the ability to remove components has been enabled for your packaging organization, you can delete these component types even if they are part of a Managed - Released package.

*** If you remove a public Visualforce component from a new version of your package and a subscriber upgrades, the component is removed from the subscriber's organization upon upgrade. If the Visualforce component is global, it remains in the subscriber organization until the Administrator (System Administrator) deletes it.

Component Attributes and Behaviors

Only some attributes of a component are upgradeable. Many components also behave differently or include additional restrictions in a managed package. It's important to consider these behaviors when designing your package.

Deleting Visualforce Pages and Global Visualforce Components

Before you delete Visualforce pages or global Visualforce components from your package, remove all references to public Apex classes and public Visualforce components from the pages or components that you're deleting. After removing the references, upgrade your subscribers to an interim package version before you delete the page or global component.

SEE ALSO:

[Editing Components and Attributes after Installation](#)

[Components Automatically Added to Packages](#)

[Deleting Components in Managed Packages](#)

Components Available in Unmanaged Packages

Not all components can be packaged for distribution. The following table lists the components that are available in an unmanaged package, how the component is included in the package, and whether the component supports automatic renaming.

Packaged Explicitly or Implicitly

Components can be added either explicitly or implicitly. Explicit components must be included directly in the package, while implicit components are automatically added. For example, if you create a custom field on a standard object, you must explicitly add the custom field to your package. However, if you create a custom object and add a custom field to it, the field is implicitly added to the package when you add the custom object.

- **Explicitly:** The component must be manually added to the package.
- **Implicitly:** The component is automatically added to the package when another dependent component, usually a custom object, is added.

Automatic Renaming

Salesforce can resolve naming conflicts automatically on install.

- **No:** If a naming conflict occurs the install is blocked.
- **Yes:** If a naming conflict occurs Salesforce can optionally change the name of the component being installed.

Component	Packaged Explicitly or Implicitly	Automatic Renaming
Reporting Snapshot	Explicitly	Yes
Apex Class	Explicitly	No
Apex Sharing Reason	Implicitly On an extension: Explicitly	No
Apex Sharing Recalculation	Implicitly	No
Apex Trigger	On a standard or extension object: Explicitly On an object in the package: Implicitly	No
Application	Explicitly	No
Custom Button or Link	On a standard object: Explicitly On a custom object: Implicitly	No

Component	Packaged Explicitly or Implicitly	Automatic Renaming
Custom Field	On a standard object: Explicitly On a custom object: Implicitly	No
Custom Label	Implicitly	No
Custom Object	Explicitly	No
Custom Permission	Implicitly With required custom permissions: Explicitly	No
Custom Report Type	Explicitly	No
Custom Setting	Explicitly	No
Dashboard	Explicitly In a folder: Implicitly	Yes
Document	Explicitly In a folder: Implicitly	Yes
Email Template	Explicitly In a folder: Implicitly	Yes
External Data Source	Explicitly Referenced by an external object: Implicitly Assigned by a permission set: Implicitly	No
Folder	Explicitly	Yes
Home Page Component	Explicitly	No
Home Page Layout	Explicitly	No
Letterhead	Explicitly	Yes
Lightning Application	Explicitly	No
Lightning Component	Explicitly	No
Lightning Event	Explicitly	No
Lightning Interface	Explicitly	No
List View	On a standard object: Explicitly On a custom object: Implicitly	Yes
Named Credential	Explicitly	No
Page Layout	On a standard object: Explicitly On a custom object: Implicitly	No

Component	Packaged Explicitly or Implicitly	Automatic Renaming
Record Type	On a standard object: Explicitly On a custom object: Implicitly	No
Report	Explicitly In a folder: Implicitly	Yes
S-Control	Explicitly	No
Static Resource	Explicitly	No
Tab	Explicitly	No
Translation	Explicitly	No
Validation Rule	On a standard object: Explicitly On a custom object: Implicitly	No
Visualforce Component	Explicitly	No
Visualforce Page	Explicitly	No
Workflow Email Alert	Explicitly	No
Workflow Field Update	Explicitly	No
Workflow Outbound Message	Explicitly	No
Workflow Rule	Explicitly	No
Workflow Task	Explicitly	No

SEE ALSO:

[Components Automatically Added to Packages](#)

Editing Components and Attributes after Installation

The following table shows which components and attributes are editable after installation from a managed package. The following sections describe the table columns and their values.

Developer Editable

The developer can edit the component attributes in this column. These attributes are locked in the subscriber's organization.

Subscriber and Developer Editable

The subscriber and developer can edit the component attributes in this column. However, these attributes aren't upgradeable. Only new subscribers receive the latest changes.

Locked

After a package is Managed - Released, the developer and subscriber can't edit the component attributes in this column.

Component	Developer Editable	Subscriber and Developer Editable	Locked
Action		<ul style="list-style-type: none"> Target Record Type Action layout Predefined values for action fields 	<ul style="list-style-type: none"> All fields except Target Record Type
Reporting Snapshot		<ul style="list-style-type: none"> All attributes except Reporting Snapshot Unique Name 	<ul style="list-style-type: none"> Reporting Snapshot Unique Name
Apex Class	<ul style="list-style-type: none"> API Version Code 		<ul style="list-style-type: none"> Name
Apex Sharing Reason	<ul style="list-style-type: none"> Reason Label 		<ul style="list-style-type: none"> Reason Name
Apex Sharing Recalculation		<ul style="list-style-type: none"> Apex Class 	
Apex Trigger	<ul style="list-style-type: none"> API Version Code 		<ul style="list-style-type: none"> Name
Application		<ul style="list-style-type: none"> All attributes except App Name 	<ul style="list-style-type: none"> App Name
Article Types	<ul style="list-style-type: none"> Description Label Plural Label Starts with a Vowel Sound 	<ul style="list-style-type: none"> Available for Customer Portal Channel Displays Default Sharing Model Development Status Enable Divisions Grant Access Using Hierarchy Search Layouts 	<ul style="list-style-type: none"> Name
Compact Layout	<ul style="list-style-type: none"> All attributes 		
Connected App	<ul style="list-style-type: none"> Access Method Canvas App URL Callback URL Connected App Name Contact Email Contact Phone 	<ul style="list-style-type: none"> ACS URL Entity ID IP Relaxation Manage Permission Sets Manage Profiles Mobile Start URL 	<ul style="list-style-type: none"> API Name Created Date/By Consumer Key Consumer Secret Installed By Installed Date

Component	Developer Editable	Subscriber and Developer Editable	Locked
	<ul style="list-style-type: none"> Description Icon URL Info URL Trusted IP Range Locations Logo Image URL OAuth Scopes 	<ul style="list-style-type: none"> Permitted Users Refresh Token Policy SAML Attributes Service Provider Certificate Start URL Subject Type 	<ul style="list-style-type: none"> Last Modified Date/By Version
Custom Button or Link	<ul style="list-style-type: none"> Behavior Button or Link URL Content Source Description Display Checkboxes Label Link Encoding 	<ul style="list-style-type: none"> Height Resizable Show Address Bar Show Menu Bar Show Scrollbars Show Status Bar Show Toolbars Width Window Position 	<ul style="list-style-type: none"> Display Type Name
Custom Field	<ul style="list-style-type: none"> Auto-Number Display Format Decimal Places Description Default Value Field Label Formula Length Lookup Filter Related List Label Roll-Up Summary Filter Criteria 	<ul style="list-style-type: none"> Chatter Feed Tracking Help Text Mask Type Mask Character Sharing Setting Sort Picklist Values Track Field History 	<ul style="list-style-type: none"> Child Relationship Name Data Type External ID Field Name Required Roll-Up Summary Field Roll-Up Summary Object Roll-Up Summary Type Unique
Custom Label	<ul style="list-style-type: none"> Category Short Description Value 		<ul style="list-style-type: none"> Name
Custom Object	<ul style="list-style-type: none"> Description Label Plural Label Record Name 	<ul style="list-style-type: none"> Allow Activities Allow Reports Available for Customer Portal 	<ul style="list-style-type: none"> Object Name Record Name Data Type Record Name Display Format

Component	Developer Editable	Subscriber and Developer Editable	Locked
	<ul style="list-style-type: none"> Starts with a Vowel Sound 	<ul style="list-style-type: none"> Context-Sensitive Help Setting Default Sharing Model Development Status Enable Divisions Enhanced Lookup Grant Access Using Hierarchy Search Layouts Track Field History 	
Custom Permission	<ul style="list-style-type: none"> Connected App Description Label Name 		
Custom Report Type	<ul style="list-style-type: none"> All attributes except Development Status and Report Type Name 	<ul style="list-style-type: none"> Development Status 	<ul style="list-style-type: none"> Report Type Name
Custom Setting	<ul style="list-style-type: none"> Description Label 		<ul style="list-style-type: none"> Object Name Setting Type Visibility
Dashboard		<ul style="list-style-type: none"> All attributes except Dashboard Unique Name 	<ul style="list-style-type: none"> Dashboard Unique Name
Document		<ul style="list-style-type: none"> All attributes except Document Unique Name 	<ul style="list-style-type: none"> Document Unique Name
Email Template		<ul style="list-style-type: none"> All attributes except Email Template Name 	<ul style="list-style-type: none"> Email Template Name
External Data Source	<ul style="list-style-type: none"> Type 	<ul style="list-style-type: none"> Auth Provider Certificate Custom Configuration Endpoint Identity Type OAuth Scope Password 	<ul style="list-style-type: none"> Name

Component	Developer Editable	Subscriber and Developer Editable	Locked
		<ul style="list-style-type: none"> Protocol Username 	
Field Set	<ul style="list-style-type: none"> Description Label Available fields 	<ul style="list-style-type: none"> Selected fields (only subscriber controlled) 	<ul style="list-style-type: none"> Name
Lightning Page	<ul style="list-style-type: none"> Lightning Page 		
Flow	<ul style="list-style-type: none"> Entire flow 	<ul style="list-style-type: none"> Name Description Status 	<ul style="list-style-type: none"> Flow Unique Name URL
Folder		<ul style="list-style-type: none"> All attributes except Folder Unique Name 	<ul style="list-style-type: none"> Folder Unique Name
Home Page Component	<ul style="list-style-type: none"> Body Component Position 		<ul style="list-style-type: none"> Name Type
Home Page Layout		<ul style="list-style-type: none"> All attributes except Layout Name 	<ul style="list-style-type: none"> Layout Name
Letterhead		<ul style="list-style-type: none"> All attributes except Letterhead Name 	<ul style="list-style-type: none"> Letterhead Name
Lightning Application	<ul style="list-style-type: none"> API Version Description Label Markup 		Name
Lightning Component	<ul style="list-style-type: none"> API Version Description Label Markup 		Name
Lightning Event	<ul style="list-style-type: none"> API Version Description Label Markup 		Name

Component	Developer Editable	Subscriber and Developer Editable	Locked
Lightning Interface	<ul style="list-style-type: none"> • API Version • Description • Label • Markup 		Name
List View		<ul style="list-style-type: none"> • All attributes except View Unique Name 	<ul style="list-style-type: none"> • View Unique Name
Named Credential	<ul style="list-style-type: none"> • Endpoint • Label 	<ul style="list-style-type: none"> • Auth Provider • Certificate • Identity Type • OAuth Scope • Password • Protocol • Username 	<ul style="list-style-type: none"> • Name
Page Layout		<ul style="list-style-type: none"> • All attributes except Page Layout Name 	<ul style="list-style-type: none"> • Page Layout Name
Permission Set	<ul style="list-style-type: none"> • Description • Label • Custom object permissions • Custom field permissions • Apex class access settings • Visualforce page access settings 		<ul style="list-style-type: none"> • Name
Record Type	<ul style="list-style-type: none"> • Description • Record Type Label 	<ul style="list-style-type: none"> • Active • Business Process 	<ul style="list-style-type: none"> • Name
Remote Site Setting		All attributes except Remote Site Name	<ul style="list-style-type: none"> • Remote Site Name
Report		<ul style="list-style-type: none"> • All attributes except Report Unique Name 	<ul style="list-style-type: none"> • Report Unique Name
S-Control	<ul style="list-style-type: none"> • Content • Description • Encoding • Filename 	<ul style="list-style-type: none"> • Prebuild in Page 	<ul style="list-style-type: none"> • S-Control Name • Type


Component	Developer Editable	Subscriber and Developer Editable	Locked
	<ul style="list-style-type: none"> Label 		
Static Resource	<ul style="list-style-type: none"> Description File 		<ul style="list-style-type: none"> Name
Tab	<ul style="list-style-type: none"> Description Encoding Has Sidebar Height Label S-control Splash Page Custom Link Type URL Width 	<ul style="list-style-type: none"> Salesforce Classic Ready Tab Style 	<ul style="list-style-type: none"> Tab Name
Translation	<ul style="list-style-type: none"> All attributes 		
Validation Rule	<ul style="list-style-type: none"> Description Error Condition Formula Error Location Error Message 	<ul style="list-style-type: none"> Active 	<ul style="list-style-type: none"> Rule Name
Visualforce Component	<ul style="list-style-type: none"> API Version Description Label Markup 		<ul style="list-style-type: none"> Name
Visualforce Page	<ul style="list-style-type: none"> API Version Description Label Markup 		<ul style="list-style-type: none"> Name
Workflow Email Alert		<ul style="list-style-type: none"> Additional Emails Email Template From Email Address Recipients 	<ul style="list-style-type: none"> Description
Workflow Field Update	<ul style="list-style-type: none"> Description 	<ul style="list-style-type: none"> Lookup 	<ul style="list-style-type: none"> Name



Component	Developer Editable	Subscriber and Developer Editable	Locked
	<ul style="list-style-type: none"> Field Value Formula Value 		
Workflow Outbound Message	<ul style="list-style-type: none"> Description Endpoint URL Fields to Send Send Session ID 	<ul style="list-style-type: none"> User to Send As 	<ul style="list-style-type: none"> Name
Workflow Rule	<ul style="list-style-type: none"> Description Evaluation Criteria Rule Criteria 	<ul style="list-style-type: none"> Active 	<ul style="list-style-type: none"> Rule Name
Workflow Task		<ul style="list-style-type: none"> Assign To Comments Due Date Priority Record Type Status 	<ul style="list-style-type: none"> Subject

Components Automatically Added to Packages


When adding components to your package, some related components are automatically added, if required. For example, if you add a Visualforce page to a package that references a custom controller, that Apex class is also added.

To understand what components might be automatically included, review the following list:

When you add this component:	These types of components might be automatically included:
Action	Action target object (if it's a custom object), action target field, action record type, predefined field values, action layout; and any custom fields that the action layout or predefined values refer to on the target object
Reporting Snapshot	Reports
Apex class	<p>Custom fields, custom objects, and other explicitly referenced Apex classes, as well as anything else that is directly referenced by the Apex class</p> <p> Note: If an Apex class references a custom label, and that label has translations, you must explicitly package the individual languages desired in order for those translations to be included.</p>
Apex trigger	Custom fields, custom objects, and any explicitly referenced Apex classes, as well as anything else that is directly referenced by the Apex trigger

When you add this component:	These types of components might be automatically included:
Article type	Custom fields, the default page layout
Compact layout	Custom fields
Custom app	Custom tabs (including web tabs), documents (stored as images on the tab), documents folder
Custom button or link	Custom fields and custom objects
Custom field	Custom objects
Custom home page layouts	Custom home page components on the layout
Custom settings	Apex sharing reasons, Apex sharing recalculations, Apex triggers, custom buttons or links, custom fields, list views, page layouts, record types, validation rules
Custom object	<p>Custom fields, validation rules, page layouts, list views, custom buttons, custom links, record types, Apex sharing reasons, Apex sharing recalculations, and Apex triggers</p> <p> Note:</p> <ul style="list-style-type: none"> • Apex sharing reasons are unavailable in extensions. • When packaged and installed, only public list views from an app are installed. If a custom object has any custom list views that you want to include in your package, ensure that the list view is accessible by all users.
Custom object (as an external object)	<p>External data source, custom fields, page layouts, list views, custom buttons, and custom links</p> <p> Note:</p> <ul style="list-style-type: none"> • When packaged and installed, only public list views from an app are installed. If an external object has any custom list views that you want to include in your package, ensure that the list view is accessible by all users. • In managed and unmanaged packages, external objects are included in the custom object component.
Custom tab	Custom objects (including all of its components), s-controls, and Visualforce pages
Dashboard	Folders, reports (including all of its components), s-controls, and Visualforce pages
Document	Folder
Email template	Folder, letterhead, custom fields, and documents (stored as images on the letterhead or template)
Field set	Any referenced fields
Lightning Page	Any associated actions
Lightning Page tab	Lightning Page
Flow	Custom objects, custom fields, Apex classes, and Visualforce pages
Folder	Everything in the folder

When you add this component:	These types of components might be automatically included:
Lightning application	All Lightning resources referenced by the application, such as components, events, and interfaces. Custom fields, custom objects, list views, page layouts, and Apex classes referenced by the application.
Lightning component	All Lightning resources referenced by the component, such as nested components, events, and interfaces. Custom fields, custom objects, list views, page layouts, and Apex classes referenced by the component.
Lightning event	Custom fields, custom objects, list views, and page layouts
Lightning interface	Custom fields, custom objects, list views, and page layouts
Page layout	Actions, custom buttons, custom links, s-controls, and Visualforce pages
Permission set	Any custom permissions, external data sources, Visualforce pages, and Apex classes that are assigned in the permission set
Record type	Record type mappings, compact layout
Report	Folder, custom fields, custom objects, custom report types, and custom s-controls
S-control	Custom fields and custom objects
Translation	Translated terms for the selected language on any component in the package
Validation rule	Custom fields (referenced in the formula)
Visualforce home page component	Associated Visualforce page
Visualforce pages	Apex classes that are used as custom controllers, Visualforce custom components, and referenced field sets
Workflow rule	All associated workflow alerts, field updates, outbound messages, and tasks; also, if the workflow rule is designed for a custom object, the custom object is automatically included

 **Note:** Some package components, such as validation rules or record types, might not display in the list of package components, but are included and install with the other components.

Special Behavior of Components in Packages

When you're building an app for distribution, it's important to consider how packaging affects your app and its components. Use the following information to help you determine what to include in your packages, how to design your app, and how to distribute your packages (managed or unmanaged).

 **Note:**

- For more information on the properties of each component in packages, see the [packaged components properties table](#).
- For more information on the attributes of each component in packages, see the [component attributes table](#).
- Component names must be unique within an organization. To ensure that your component names don't conflict with those in an installer's organization, use a managed package so that all your component names contain your namespace prefix.

Reporting Snapshot

Developers of managed packages must consider the implications of introducing reporting snapshots that reference reports released in a previous version of the package. If the subscriber deleted the report or moved the report to a personal folder, the reporting snapshot referencing the report will not be installed, even though the Package Installation page may indicate that it will be. Also, if the subscriber has modified the report, that report may return results impacting the information displayed by the reporting snapshot. As a best practice, the developer should release the reporting snapshot and the related reports in the same version.

Since the running user is selected by the subscriber, some reporting snapshot field mappings could become invalid if the running user does not have access to source or target fields.

Apex Classes or Triggers

Any Apex that is included as part of a package must have at least 75% cumulative test coverage. Each trigger must also have some test coverage. When you upload your package to AppExchange, all tests are run to ensure that they run without errors. In addition, all tests are run when the package is installed in the installer's organization. The installer can decide whether or not to install the package if any tests fail.



Tip: To prevent naming conflicts, Salesforce recommends using managed packages for all packages that contain Apex. This way, all of the Apex objects contain your [namespace prefix](#). For example, if there is an Apex class called `MyHelloWorld` and the namespace for your organization is `OneTruCode`, the class is referenced as `OneTruCode.MyHelloWorld`.

Keep the following considerations in mind when including Apex in your package:

- Managed packages receive a unique namespace. This namespace is automatically prepended to your class names, methods, variables, and so on, which helps prevent duplicate names in the installer's organization.
- In a single transaction, you can only reference 10 unique namespaces. For example, suppose you have an object that executes a class in a managed package when the object is updated. Then that class updates a second object, which in turn executes a different class in a different package. Even though the second package wasn't accessed directly by the first, because it occurs in the same transaction, it's included in the number of namespaces being accessed in a single transaction.
- If you are exposing any methods as Web services, include detailed documentation so that subscribers can write external code that calls your Web service.
- If an Apex class references a custom label, and that label has translations, you must explicitly package the individual languages desired in order for those translations to be included in the package.
- If you reference a custom object's sharing object (such as `MyCustomObject__share`) in Apex, this adds a sharing model dependency to your package. You must set the organization-wide sharing default access level for the custom object to Private in order for other organizations to install your package successfully.
- The code contained in an Apex class or trigger that is part of a managed package is automatically obfuscated and cannot be viewed in an installing organization. The only exceptions are methods declared as global, meaning that the method signatures can be viewed in an installing organization.
- You can use the `deprecated` annotation in Apex to identify `global` methods, classes, exceptions, enums, interfaces, and variables that can no longer be referenced in subsequent releases of the managed package in which they reside. This is useful when you are refactoring code in managed packages as the requirements evolve. After you upload another package version as Managed - Released, new subscribers that install the latest package version cannot see the deprecated elements, while the elements continue to function for existing subscribers and API integrations.
- Any Apex contained in an unmanaged package that explicitly references a namespace cannot be uploaded.
- Apex code that refers to Data Categories can't be uploaded.
- Before you delete Visualforce pages or global Visualforce components from your package, remove all references to public Apex classes and public Visualforce components from the pages or components that you're deleting. After removing the references, upgrade your subscribers to an interim package version before you delete the page or global component.

Apex Sharing Reasons

Apex sharing reasons can be added directly to a package, but are only available for custom objects.

Compact Layouts

When you package a compact layout, its record type mappings aren't included. Subscribers or installers of a package containing a compact layout must recreate its record type mappings in their organization.

Connected Apps

- Connected apps can be added to managed packages, only. Connected apps are not supported for unmanaged packages.
- Subscribers or installers of a package can't delete a connected app by itself; they can only uninstall its package. A developer can delete a connected app after a package is uploaded as Managed - Released. The connected app will be deleted in the subscriber's organization during a package upgrade.
- If you update a connected app and include it in a new package version, upgrading that package in a customer organization updates the existing connected app.
- If you push upgrade a package containing a connected app whose OAuth scope or IP ranges have changed from the previous version, the upgrade will fail. This is a security feature, to block unauthorized users from gaining broad access to a customer organization by upgrading an installed package. A customer can still perform a pull upgrade of the same package; this is allowed because it's with the customer's knowledge and consent.
- You can add an existing connected app (that is, one created prior to Summer '13) to a managed package. You can also combine new and existing connected apps in the same managed package.
- For connected apps created prior to Summer '13, the existing install URL continues to be valid until you package and upload a new version. Once you upload a new version of the package with an updated connected app, the install URL will no longer work.

Custom Console

A package that has a custom console component can only be installed in an organization with the Service Cloud license or Sales Console permission enabled.

Custom Fields

- Picklist field values for custom fields can be added, edited, or deleted by subscribers. A developer should carefully consider this when explicitly referencing a picklist value in code. Picklist values can be added or deleted by the developer. During a package upgrade, no new picklist values are installed into the subscriber's organization for existing fields. Any picklist values deleted by the developer are still available in the subscriber's organization.
- Developers can add required and universally required custom fields to managed packages as long as they have default values.
- Auto-number type fields and required fields cannot be added after the object is uploaded in a Managed - Released package.

Custom Labels

If a label is translated, the language must be explicitly included in the package in order for the translations to be included in the package. Subscribers can override the default translation for a custom label.

Custom Objects

- If a developer enables the `Allow Reports` or `Allow Activities` attributes on a packaged custom object, the subscriber's organization also has these features enabled during an upgrade. Once enabled in a Managed - Released package, the developer and the subscriber cannot disable these attributes.
- Standard button and link overrides are also packageable.

Custom Permissions

If you deploy a change set with a custom permission that includes a connected app, the connected app must already be installed in the destination organization.

Custom Report Types

A developer can edit a custom report type in a managed package after it is released, and add new fields. Subscribers automatically receive these changes when they install a new version of the managed package. However, developers can't remove objects or fields from the report type once the package is released.

Custom Settings

- If a custom setting is contained in a managed package, and the `Visibility` is specified as Protected, the custom setting is not contained in the list of components for the package on the subscriber's organization. All data for the custom setting is hidden from the subscriber.

Custom Tabs

- The `Tab Style` for a custom tab must be unique within your app. However, it does not need to be unique within the organization where it is installed. A custom tab's style will not conflict with an existing custom tab in the installer's environment.
- To provide custom tab names in different languages, from Setup, click **Customize > Tab Names and Labels > Rename Tabs and Labels**.
- Subscribers cannot edit custom tabs in a managed package.

Customer Portal and Partner Portal

Packages referring to Customer Portal or partner portal fields are supported. The subscriber installing the package must have the respective portal enabled to install the package.

Dashboard Components

Developers of managed packages must consider the implications of introducing dashboard components that reference reports released in a previous version of the package. If the subscriber deleted the report or moved the report to a personal folder, the dashboard component referencing the report is dropped during install. Also, if the subscriber has modified the report, that report may return results impacting what information is displayed by the dashboard component. As a best practice, the developer should release a dashboard and the related reports in the same version.

Divisions

- When divisions are enabled on a custom object in a package, the subscribing organization must have the divisions feature enabled to install the package.
- Setting the division filter on a report does not cause a dependency. The setting is dropped when installed into the subscriber's organization.
- Summarizing by the object's division field—for example, Account Division—in a report causes a dependency.
- If the object's division field in a report is included as a column, and the subscriber's organization does not support divisions on the object, then the column is dropped during install.
- If you install a custom report type that includes an object's division field as a column, that column is dropped if the organization does not support divisions.

External Data Sources

- After installing an external data source from a managed or unmanaged package, the subscriber must re-authenticate to the external system.
 - For password authentication, the subscriber must re-enter the password in the external data source definition.
 - For OAuth, the subscriber must update the callback URL in the client configuration for the authentication provider and then re-authenticate by selecting `Start Authentication Flow on Save` on the external data source.
- Certificates aren't packageable. If you package an external data source that specifies a certificate, make sure that the subscriber organization has a valid certificate with the same name.

External Objects

In managed and unmanaged packages, external objects are included in the custom object component.


Field Dependencies

- Developers and subscribers can add, change, or remove field dependencies.
- If the developer adds a field dependency, it is added during installation unless the subscriber has already specified a dependency for the same field.
- If a developer removes a dependency, this change is not reflected in the subscriber's organization during an upgrade.
- If the developer introduces a new picklist value mapping between the dependent and controlling fields, the mapping is added during an upgrade.
- If a developer removes a picklist value mapping, the change is not reflected in the subscriber's organization during an upgrade.

Field Sets

Field sets in installed packages perform different merge behaviors during a package upgrade:

If a package developer:	Then in the package upgrade:
Changes a field from <code>Unavailable</code> to Available for the Field Set or In the Field Set	The modified field is placed at the end of the upgraded field set in whichever column it was added to.
Adds a new field	The new field is placed at the end of the upgraded field set in whichever column it was added to.
Changes a field from Available for the Field Set or In the Field Set to <code>Unavailable</code>	The field is removed from the upgraded field set.
Changes a field from In the Field Set to Available for the Field Set (or vice versa)	The change is not reflected in the upgraded field set.

 **Note:** Subscribers aren't notified of changes to their installed field sets. The developer must notify users—through the package release notes or other documentation—of any changes to released field sets. Merging has the potential to remove fields in your field set.

Once a field set is installed, a subscriber can add or remove any field.

Flows

- You can package only active flows. The active version of the flow is determined when you upload a package version. If none of the flow's versions are active, the upload fails.
- To update a managed package with a different flow version, activate that version and upload the package again. You don't need to add the newly activated version to the package. However, if you activate a flow version by mistake and upload the package, you'll distribute that flow version to everyone. Be sure to verify which version you really want to upload.
- In a development organization, you can't delete a flow or flow version after you upload it to a released or beta managed package.
- You can't delete flow components from Managed - Beta package installations in development organizations.
- You can't delete a flow from an installed package. To remove a packaged flow from your organization, deactivate it and then uninstall the package.
- If you have multiple versions of a flow installed from multiple unmanaged packages, you can't remove only one version by uninstalling its package. Uninstalling a package—managed or unmanaged—that contains a single version of the flow removes the entire flow, including all versions.
- You can't include flows in package patches.

- An active flow in a package is active after it's installed. The previous active version of the flow in the destination organization is deactivated in favor of the newly installed version. Any in-progress flows based on the now-deactivated version continue to run without interruption but reflect the previous version of the flow.
- Upgrading a managed package in your organization installs a new flow version only if there's a newer flow version from the developer. After several upgrades, you can end up with multiple flow versions.
- If you install a package that contains multiple flow versions in a fresh destination organization, only the latest flow version is deployed.
- If you install a flow from an unmanaged package that has the same name but a different version number as a flow in your organization, the newly installed flow becomes the latest version of the existing flow. However, if the packaged flow has the same name and version number as a flow already in your organization, the package install fails. You can't overwrite a flow.
- The Cloud Flow Designer can't open flows that are installed from managed packages.

Folders

- Components that Salesforce stores in folders, such as documents, cannot be added to packages when stored in personal and unfiled folders. Put documents, reports, and other components that Salesforce stores in folders in one of your publicly accessible folders.
- Components such as documents, email templates, reports, or dashboards are stored in new folders in the installer's organization using the publisher's folder names. Give these folders names that indicate they are part of the package.
- If a new report, dashboard, document, or email template is installed during an upgrade, and the folder containing the component was deleted by the subscriber, the folder is re-created. Any components in the folder that were previously deleted are not restored.
- The name of a component contained in a folder must be unique across all folders of the same component type, excluding personal folders. Components contained in a personal folder must be unique within the personal folder only.

Home Page Components

When you package a custom home page layout, all the custom home page components included on the page layout are automatically added. Standard components such as Messages & Alerts are not included in the package and do not overwrite the installer's Messages & Alerts. To include a message in your custom home page layout, create an HTML Area type custom Home tab component containing your message, under Setup, in **Customize > Home > Home Page Components** and add it to your custom home page layout.

Home Page Layouts

Once installed, your custom home page layouts are listed with all the subscriber's home page layouts. Distinguish them by including the name of your app in the page layout name.

List Views

List views associated with queues cannot be included in a package.

Multi-Currency

- If a subscriber installs a report or custom report type that includes an object's currency field as a column, that column is dropped if the subscriber's organization is not enabled for multiple currencies.
- Referencing an object's currency field in a report's criteria—for example, `Account Currency`—causes a dependency.
- Summarizing by an object's currency field in a report causes a dependency.
- Using a currency designation in a report criteria value—for example, "Annual Revenue equals GBP 100"—does not cause a dependency. The report generates an error when run in the installers organization if it does not support the currency.
- If an object's currency field in a report is included as a column and the subscriber's organization is not enabled for multiple currencies, that column is dropped during install.
- If a subscriber installs a custom report type that includes an object's currency field as a column, that column is dropped if the organization is not enabled for multiple currencies.

Named Credentials

- After installing an external data source from a managed or unmanaged package, the subscriber must re-authenticate to the external system.
 - For password authentication, the subscriber must re-enter the password in the named credential definition.
 - For OAuth, the subscriber must update the callback URL in the client configuration for the authentication provider and then re-authenticate by selecting **Start Authentication Flow on Save** on the named credential.
- Named credentials aren't automatically added to packages. If you package Apex code that specifies a named credential as a callout endpoint, also add the named credential to that package or otherwise make sure that the subscriber organization has a valid named credential with the same name.

If you have multiple organizations, you can create a named credential with the same name in each organization. Each of these named credentials can have a different endpoint URL, for example, to accommodate differences in development and production environments. Because the code references only the named credential's name, you can package and deploy the same Apex class in all your organizations without programmatically checking the environment.

If you add the Apex code to a managed package that doesn't contain the referenced named credential, include the namespace prefix when specifying the endpoint. For a subscriber organization that has no namespace set, use the `.` namespace prefix to reference the named credential. For example:

```
req.setEndpoint('callout:.__My_Named_Credential/some_path');
```

- Certificates aren't packageable. If you package a named credential that specifies a certificate, make sure that the subscriber organization has a valid certificate with the same name.

Page Layouts

The page layout of the person uploading a package is the layout used for Group and Professional Edition organizations and becomes the default page layout for Enterprise, Unlimited, Performance, and Developer Edition organizations.

Page layouts should be packaged alongside complimentary record types if the layout is being installed on an existing object. Otherwise, the installed page layouts must be manually applied to profiles.

If a page layout and a record type are created as a result of installing a package, then the uploading user's page layout assignment for that record type is assigned to that record type for all profiles in the subscriber organization unless a profile is mapped during an install or upgrade.

Permission Sets

You can include permission sets as components in a package, with the following permissions and access settings:

- Custom object permissions
- External object permissions
- Custom field permissions
- Custom permissions
- Apex class access
- Visualforce page access
- External data source access



Note: Assigned apps and tab settings are *not* included in permission set components.

Use permission sets to install or upgrade a collection of permissions. In contrast to profile settings, permission sets don't overwrite profiles.

Picklist Values

- Subscribers can rename or delete picklist field values. A developer should carefully consider this when explicitly referencing a picklist field value in Apex.
- Picklist field values can be added or deleted in the developer's organization. Upon upgrade, no new values are installed. Any values deleted by the developer are still available in the subscriber's organization until the subscriber deletes them.

Profile Settings

Profile settings include the following for components in the package:

- Assigned apps
- Assigned connected apps
- Tab settings
- Page layout assignments
- Record type assignments
- Custom object permissions
- External object permissions
- Custom field permissions
- Custom permissions
- Apex class access
- Visualforce page access
- External data source access

Profile settings overwrite existing profiles in the installer's organization with specific permission and setting changes.

Record Types

- If record types are included in the package, the subscriber's organization must support record types to install the package.
- When a new picklist value is installed, it is associated with all installed record types according to the mappings specified by the developer. A subscriber can change this association.
- Referencing an object's record type field in a report's criteria—for example, `Account Record Type`—causes a dependency.
- Summarizing by an object's record type field in a report's criteria—for example, `Account Record Type`—causes a dependency.
- If an object's record type field is included as a column in a report, and the subscriber's organization is not using record types on the object or does not support record types, then the column is dropped during install.
- If you install a custom report type that includes an object's record type field as a column, that column is dropped if the organization does not support record types or the object does not have any record types defined.

Reports

If a report includes elements that cannot be packaged, those elements will be dropped or downgraded, or the package upload will fail. For example:

- Hierarchy drill-downs are dropped from activity and opportunities reports.
- Filters on unpackageable fields are automatically dropped (for example, in filters on standard object record types).
- Package upload fails if a report includes filter logic on an unpackageable field (for example, in filters on standard object record types).
- Lookup values on the `Select Campaign` field of standard campaign reports are dropped.
- Reports are dropped from packages if they have been moved to a private folder or to the Unfiled Public Reports folder.
- When a package is installed into an organization that does not have Chart Analytics 2.0:

- Combination charts are downgraded instead of dropped. For example, a combination vertical column chart with a line added is downgraded to a simple vertical column chart; a combination bar chart with additional bars is downgraded to a simple bar chart.
- Unsupported chart types, such as donut and funnel, are dropped.

S-Controls

Only s-controls in unmanaged packages created before January 1st, 2010 can be installed by subscribers.

S-controls have been deprecated, and are superseded by “Visualforce” in the Salesforce Help pages.

Translation Workbench

- If you have enabled the translation workbench and added a language to your package, any associated translated values are automatically packaged for the appropriate components in your package. Make sure that you have provided translations for all possible components.
- An installer of your package can see which languages are supported on the package detail page. The installer does not need to enable anything for the packaged language translations to appear. The only reason installers may want to enable the translation workbench is to change translations for unmanaged components after installation, override custom label translations in a managed package, or to translate into additional languages.
- If you are designing a package extension, you can include translations for the extension components but not additional translations for components in the base package.

Validation Rules

For custom objects that are packaged, any associated validation rules are implicitly packaged as well.

Workflow

- Salesforce prevents you from uploading workflow alerts that have a public group, partner user, or role recipient. Change the recipient to a user before uploading your app. During installation, Salesforce replaces that user with the user installing the app, and the installer can customize it as necessary.
- Salesforce prevents you from uploading workflow field updates that change an `Owner` field to a queue. Change the updated field value to a user before uploading your app. During installation, Salesforce replaces that user with the user installing the app, and the installer can customize it as necessary.
- Salesforce prevents you from uploading workflow rules, field updates, and outbound messages that reference a record type on a standard or managed-installed object.
- Salesforce prevents you from uploading workflow tasks that are assigned to a role. Change the `Assigned To` field to a user before uploading your app. During installation, Salesforce replaces that user with the user installing the app, and the installer can customize it as necessary.
- You can package workflow rules and associated workflow actions, such as email alerts and field updates. However, any time-based triggers aren’t included in the package. Notify your installers to set up any time-based triggers that are essential to your app.
Flow triggers aren’t packageable. The Process Builder has superseded flow trigger workflow actions, formerly available in a pilot program. Organizations that are using flow trigger workflow actions can continue to create and edit them, but flow trigger workflow actions aren’t available for new organizations. For information on enabling the Process Builder in your organization, contact Salesforce.
- Some workflow actions can be protected by the developer. For more information on protected components, see “Protected Components” in the Salesforce Help.
- Developers can associate or disassociate workflow actions with a workflow rule at any time. These changes, including disassociation, are reflected in the subscriber’s organization upon install. In managed packages, a subscriber cannot disassociate workflow actions from a workflow rule if it was associated by the developer.

- References to a specific user in workflow actions, such as the email recipient of a workflow email alert, are replaced by the user installing the package. Workflow actions referencing roles, public groups, account team, opportunity team, or case team roles may not be uploaded.
- References to an organization-wide address, such as the `From email address` of a workflow email alert, are reset to Current User during installation.
- On install, all workflow rules newly created in the installed or upgraded package, have the same activation status as in the uploaded package.

Protected Components

Developers can mark certain components as *protected*. Protected components can't be linked to or referenced by components created in a subscriber organization. A developer can delete a protected component in a future release without worrying about failing installations. However, once a component is marked as unprotected and is released globally, the developer can't delete it.

The developer can mark the following components as protected in managed packages.

- Custom labels
- Custom links (for Home page only)
- Workflow alerts
- Workflow field updates
- Workflow outbound messages
- Workflow tasks
- Workflow flow triggers

The Process Builder has superseded flow trigger workflow actions, formerly available in a pilot program. Organizations that are using flow trigger workflow actions can continue to create and edit them, but flow trigger workflow actions aren't available for new organizations. For information on enabling the Process Builder in your organization, contact Salesforce.


Understanding Dependencies

Package dependencies are created when one component references another component, permission, or preference that is required for the component to be valid. Force.com tracks certain dependencies, including:

- Organizational dependencies, such as whether multicurrency or campaigns are enabled
- Component-specific dependencies, such as whether particular record types or divisions exist
- References to both standard and custom objects or fields

Packages, Apex classes, Apex triggers, Visualforce components, and Visualforce pages can have dependencies on components within an organization. These dependencies are recorded on the Show Dependencies page.

Dependencies are important for packaging because any dependency in a component of a package is considered a dependency of the package as a whole.

 **Note:** An installer's organization must meet all dependency requirements listed on the Show Dependencies page or else the installation will fail. For example, the installer's organization must have divisions enabled to install a package that references divisions.

Dependencies are important for Apex classes or triggers because any component on which a class or trigger depends must be included with the class or trigger when the code is deployed or packaged.

In addition to dependencies, the *operational scope* is also displayed on the Show Dependencies page. The operational scope is a table that lists any data manipulation language (DML) operations (such as `insert` or `merge`) that Apex executes on a specified object. The operational scope can be used when installing an application to determine the full extent of the application's database operations.

To view the dependencies and operational scope for a package, Apex class, Apex trigger, or Visualforce page:

1. Navigate to the appropriate component from Setup:
 - For packages, click **Create** > **Packages**.
 - For Apex classes, click **Develop** > **Apex Classes**.
 - For Apex triggers on standard objects, click **Customize**, click the name of the object, and then click **Triggers**.
 - For Apex triggers on custom objects, click **Create**, click the name of the object, and then click **Triggers**.
 - For Visualforce pages, click **Develop** > **Pages**.
2. Select the name of the component.
3. Click **View Dependencies** for a package, or **Show Dependencies** for all other components, to see a list of objects that depend upon the selected component.

If a list of dependent objects displays, click **Fields** to access the field-level detail of the operational scope. The field-level detail includes information, such as whether a field is updated by Apex. For more information, see "Field Operational Scope" in the Salesforce Help.

Packages, Apex code, and Visualforce pages can be dependent on many components, including but not limited to:

- Custom field definitions
- Validation formulas
- Reports
- Record types
- Apex

EDITIONS

AppExchange packages and Visualforce are available in: **Group, Professional, Enterprise, Performance, Unlimited, and Developer** Editions

Apex available in: **Enterprise, Performance, Unlimited, and Developer** Editions

USER PERMISSIONS

To upload packages:

- "Upload AppExchange Packages"

To view Apex dependencies:

- "Author Apex"

To view Visualforce dependencies:


- "Developer Mode"

- Visualforce pages and components

For example, if a Visualforce page includes a reference to a multicurrency field, such as `{ !contract.ISO_code }`, that Visualforce page has a dependency on multicurrency. If a package contains this Visualforce page, it also has a dependency on multicurrency. Any organization that wants to install this package must have multicurrency enabled.

About Permission Sets and Profile Settings

Developers can use permission sets or profile settings to grant permissions and other access settings to a package. When deciding whether to use permission sets, profile settings, or a combination of both, consider the similarities and differences.

Behavior	Permission Sets	Profile Settings
What permissions and settings are included?	<ul style="list-style-type: none"> • Custom object permissions • External object permissions • Custom field permissions • Custom permissions • Apex class access • Visualforce page access • External data source access <p> Note: Permission sets include assigned apps and tab settings, but these settings can't be packaged as permission set components.</p>	<ul style="list-style-type: none"> • Assigned apps • Assigned connected apps • Tab settings • Page layout assignments • Record type assignments • Custom object permissions • External object permissions • Custom field permissions • Custom permissions • Apex class access • Visualforce page access • External data source access
Can they be upgraded in managed packages?	Yes.	Profile settings are applied to existing profiles in the subscriber's organization on install or upgrade. Only permissions related to new components created as part of the install or upgrade are applied.
Can subscribers edit them?	Subscribers can edit permission sets in unmanaged packages, but not in managed packages.	Yes.
Can you clone or create them?	Yes. However, if a subscriber clones a permission set or creates one that's based on a packaged permission set, it won't be updated in subsequent upgrades. Only the permission sets included in a package are upgraded.	Yes. Subscribers can clone any profile that includes permissions and settings related to packaged components.
Do they include standard object permissions?	No. Also, you can't include object permissions for a custom object in a master-detail relationship where the master is a standard object.	No.
Do they include user permissions?	No.	No.

Behavior	Permission Sets	Profile Settings
Are they included in the installation wizard?	No. Subscribers must assign permission sets after installation.	Yes. Profile settings are applied to existing profiles in the subscriber's organization on install or upgrade. Only permissions related to new components created as part of the install or upgrade are applied.
What are the user license requirements?	<p>A permission set is only installed if the subscriber organization has at least one user license that matches the permission set. For example, permission sets with the Salesforce Platform user license aren't installed in an organization that has no Salesforce Platform user licenses. If a subscriber subsequently acquires a license, they must reinstall the package to get the permission sets associated with the newly acquired license.</p> <p>Permission sets with no user license are always installed. If you assign a permission set with no user license, all of its enabled settings and permissions must be allowed by the user's license, or the assignment will fail.</p>	None. In a subscriber organization, the installation overrides the profile settings, not their user licenses.
How are they assigned to users?	Subscribers must assign packaged permission sets after installing the package.	Profile settings are applied to existing profiles.

Best Practices

- Use permission sets in addition to packaged profiles so your subscribers can easily add new permissions for existing app users.
- If users need access to apps, tabs, page layouts, and record types, don't use permission sets as the sole permission-granting model for your app. Assigned apps and tab settings are available in the permission set user interface, but they aren't included in permission set package components.
- Create packaged permission sets that grant access to the custom components in a package, but not standard Salesforce components.

Creating Custom Profile Settings

When designing your app for AppExchange, create custom profiles for the various users of your app. Give each custom profile the appropriate level of access to your app, such as tab visibility or field-level security settings for approval fields on custom objects. Use the following tips when creating custom profiles for apps you want to publish:

- Give each custom profile a meaningful name that makes it obvious that the profile applies to the app. For example, if you are creating a Human Resources app with the name "HR2GO," you might want to name a custom profile "HR2GO Approving Manager" for managers that approve requisitions.
- If your custom profiles have a hierarchy, make sure the name indicates where in the hierarchy that profile is. For example, a senior-level manager's profile can have the name "HR2GO Level 2 Approving Manager."

- Avoid giving your custom profiles names that can be interpreted differently in other organizations. For example, “HR2GO Level 2 Approving Manager” is a more appropriate profile name than “Sr. Manager.”
- The profile `Description` displays to the user installing your app. Provide a meaningful description for each custom profile for those users.

When you package an app, you can include your custom profiles. Profile settings are applied to existing profiles in the subscriber's organization on install or upgrade. Only permissions related to new components created as part of the install or upgrade are applied. The security settings associated with standard objects and existing custom objects in an installer's organization are unaffected.

Alternatively, you can use permission sets to avoid overwriting permissions in subscriber profiles and to maintain control of permission settings through the upgrade process. Permission sets contain a subset of profile access settings, including object permissions, field permissions, Apex class access, and Visualforce page access. These are the same permissions that are available on profiles. You can add a permission set as a component in a package.



Note: In packages, assigned apps and tab settings aren't included in permission set components.

Protecting Your Intellectual Property

The details of your custom objects, custom links, reports, and all other installed items are completely revealed to installers. While this allows an installer to review an app and all its components for any malicious content, it prevents developers from protecting some intellectual property.

The following information is important when considering your intellectual property and its protection.

- Only publish package components that are your intellectual property and that you have the rights to share.
- Once components are available on AppExchange, you cannot recall them from anyone who has installed them.
- The information in the components you package and publish might be visible to customers. Use caution when adding your code to a formula, Visualforce page, or any other component that you cannot hide in your app.
- The code contained in an Apex class or trigger that is part of a managed package is automatically obfuscated and cannot be viewed in an installing organization. The only exceptions are methods declared as global, meaning that the method signatures can be viewed in an installing organization.
- If a custom setting is contained in a managed package, and the `Visibility` is specified as Protected, the custom setting is not contained in the list of components for the package on the subscriber's organization. All data for the custom setting is hidden from the subscriber.

Creating Packaged Applications with Chatter

The objects, field settings, and field settings history of Chatter are packageable. However, an object's field is only tracked if the object itself is tracked. For example, you can create a new custom field on the Account standard object, but the field will only be tracked if you have enabled feed tracking on Accounts.

When developing applications that use Chatter, it's important to be aware that some organizations might not have Chatter enabled. By default, when you upload Chatter applications, the package is only available to organizations that have Chatter enabled. You can change this behavior and allow organizations to install the package even if they don't have Chatter. Note the following:

- You must use a managed package. Unmanaged packages that include Chatter functionality can only be installed in organizations that have Chatter enabled.
- DML operations and SOSL, and SOQL calls will throw a runtime exception if the subscriber organization does not have Chatter enabled. You must catch and handle any Apex exceptions that are thrown as a result of the missing Chatter feature. These exceptions are of the type `REQUIRED_FEATURE_MISSING_EXCEPTION` for SOSL and SOQL calls. For DML calls, you must check for the specific `REQUIRED_FEATURE_MISSING` status code on a DML Exception.

- When you upload the package, deselect the Chatter required checkbox (this is automatically selected if you have an Apex reference to Chatter).



Note: If the Chatter required checkbox can't be deselected, then some component in the package has a special requirement for Chatter. This can happen, for example, if you package a custom report type that relies on Chatter. If the Chatter-required checkbox can't be disabled, then the package can only be installed in organizations that have Chatter enabled.

The following example tries to post to feeds and get a user's feed. If Chatter is not enabled in the organization, the code catches the `REQUIRED_FEATURE_MISSING` exception. Note that this is an incomplete code example and does not run.

```
public void addFeedItem(String post, Id objId) {
    FeedItem fpost = new FeedItem();
    // Get the parent ID of the feed
    fpost.ParentId = objId;
    fpost.Body = post;
    try{
        insert fpost;
    } catch (System.DmlException e) {
        for (Integer i = 0; i < e.getNumDml(); i++) {
            // Chatter not enabled, do not insert record
            System.assertEquals(StatusCode.REQUIRED_FEATURE_MISSING, e.getDmlType(i));
            System.Debug('Chatter not enabled in this organization:' + e.getDMLMessage());
        }
    }
}

public List<NewsFeed> getMyFeed() {
    List<NewsFeed> myfeed;
    try{
        myfeed = [SELECT Id, Type, CreatedById, CreatedBy.FirstName, CreatedBy.LastName,
            CreatedDate, ParentId, Parent.Name, FeedItemId, Body,
            Title, CreatedById, LinkUrl,
            (SELECT Id, FieldName, OldValue, NewValue
             FROM FeedTrackedChanges ORDER BY Id DESC),
            (SELECT Id, CommentBody, CreatedDate, CreatedById,
             CreatedBy.FirstName, CreatedBy.LastName
             FROM FeedComments ORDER BY CreatedDate DESC, ID DESC LIMIT 10)
            FROM NewsFeed
            ORDER BY CreatedDate DESC, ID DESC LIMIT 20];
    } catch (System.RequiredFeatureMissingException e){
        // The above has returned an empty NewsFeed
        // Chatter is not enabled in this organization
        myfeed = new List<NewsFeed>{};
        System.Debug('Chatter not enabled in organization:' + e.getMessage());
    }
    return myfeed;
}
```

Matching the Salesforce Look and Feel

Apps that resemble the Salesforce user interface look and feel are instantly more familiar to users and easy to use. The easiest way to model the design of your app after the Salesforce user interface look and feel is to use Visualforce. When you use a standard controller

with a Visualforce page, your new page takes on the style of the associated object's standard tab in Salesforce. For more information, see [Using Salesforce Styles](#) in the *Visualforce Developer's Guide*.

Developing App Documentation

Salesforce recommends publishing your app on AppExchange with the following types of documentation:

Configure Option

You can include a **Configure** option for installers. This option can link to installation and configuration details, such as:

- Provisioning the external service of a composite app
- Custom app settings

The **Configure** option is included in your package as a custom link. You can create a custom link for your home page layouts and add it to your package.

1. Create a custom link to a URL that contains configuration information or a Visualforce page that implements configuration. When you create your custom link, set the display properties to `Open in separate popup window` so that the user returns to the same Salesforce page when done.
2. When you create the package, choose this custom link in the `Configure Custom Link` field of your package detail.

Data Sheet


Give installers the fundamental information they need to know about your app before they install.

Customization and Enhancement Guide

Let installers know what they must customize after installation as part of their implementation.

Custom Help

You can provide custom help for your custom object records and custom fields.

 **Tip:** To give your custom help a professional tone using Salesforce terminology, follow the [Salesforce Style Guide for Documentation and User Interface Text](#).

EDITIONS

Available in:

- Group
- Professional
- Enterprise
- Performance
- Unlimited
- Developer

About API and Dynamic Apex Access in Packages

ApexPackage components have access via dynamic Apex and the API to standard and custom objects in the organization where they are installed. Developers of Force.com AppExchange packages that are intended for external customers (also called third-party developers or partners) may wish to restrict this access. Restricting access makes packages safer for administrators to install. Also, administrators who install such packages may wish to restrict this access after installation, even if the package developers have not, for improved security.

`API Access` is a package setting that controls the dynamic Apex and API access that s-controls and other package components have to standard and custom objects. The setting displays for both the developer and installer on the package detail page. With this setting:

- The developer of an AppExchange package can restrict API access for a package before uploading it to Force.com AppExchange. Once restricted, the package components receive Apex and API sessions that are restricted to the custom objects in the package. The developer can also enable access to specific standard objects, and any custom objects in other packages that this package depends on.
- The installer of a package can accept or reject package access privileges when installing the package to his or her organization.

EDITIONS

Available in:

- Contact Manager
- Group
- Professional
- Enterprise
- Performance
- Unlimited
- Developer

- After installation, an administrator can change Apex and API access for a package at any time. The installer can also enable access on additional objects such as custom objects created in the installer's organization or objects installed by unrelated packages.

There are two possible options for the `API Access` setting:

- The default `Unrestricted`, which gives the package components the same API access to standard objects as the user who is logged in when the component sends a request to the API. Apex runs in system mode. Unrestricted access gives Apex read access to all standard and custom objects.
- `Restricted`, which allows the administrator to select which standard objects the components in the package can access. Further, the components in restricted packages can only access custom objects in the current package if the user has the object permissions that provide access to them.

Considerations for API and Dynamic Apex Access in Packages

By default, dynamic Apex can only access the components with which the code is packaged. To provide access to standard objects not included in the package, the developer must set the `API Access`.

1. Go to **Create > Packages**.
2. Select the package that contains a dynamic Apex that needs access to standard objects in the installing organization.
3. In the Package Detail related list, click **Enable Restrictions** or `Restricted`, whichever is available.
4. Set the access level (Read, Create, Edit, Delete) for the standard objects that the dynamic Apex can access.
5. Click **Save**.

Choosing `Restricted` for the `API Access` setting in a package affects the following:

- API access in a package overrides the following user permissions:
 - Author Apex
 - Customize Application
 - Edit HTML Templates
 - Edit Read Only Fields
 - Manage Billing
 - Manage Call Centers
 - Manage Categories
 - Manage Custom Report Types
 - Manage Dashboards
 - Manage Letterheads
 - Manage Package Licenses
 - Manage Public Documents
 - Manage Public List Views
 - Manage Public Reports
 - Manage Public Templates
 - Manage Users
 - Transfer Record
 - Use Team Reassignment Wizards
 - View Setup and Configuration
 - Weekly Export Data

- If Read, Create, Edit, and Delete access are not selected in the API access setting for objects, users do not have access to those objects from the package components, even if the user has the “Modify All Data” and “View All Data” permissions.
- A package with Restricted API access can’t create new users.
- Salesforce denies access to Web service and executeanonymous requests from an AppExchange package that has Restricted access.

The following considerations also apply to API access in packages:

- Workflow rules and Apex triggers fire regardless of API access in a package.
- If a component is in more than one package in an organization, API access is unrestricted for that component in all packages in the organization regardless of the access setting.
- If Salesforce introduces a new standard object after you select restricted access for a package, access to the new standard object is not granted by default. You must modify the restricted access setting to include the new standard object.
- When you upgrade a package, changes to the API access are ignored even if the developer specified them. This ensures that the administrator installing the upgrade has full control. Installers should carefully examine the changes in package access in each upgrade during installation and note all acceptable changes. Then, because those changes are ignored, the administrator should manually apply any acceptable changes after installing an upgrade.
- S-controls are served by Salesforce and rendered inline in Salesforce. Because of this tight integration, there are several means by which an s-control in an installed package could escalate its privileges to the user’s full privileges. In order to protect the security of organizations that install packages, s-controls have the following limitations:
 - For packages you are developing (that is, not installed from AppExchange), you can only add s-controls to packages with the default Unrestricted API access. Once a package has an s-control, you cannot enable Restricted API access.
 - For packages you have installed, you can enable access restrictions even if the package contains s-controls. However, access restrictions provide only limited protection for s-controls. Salesforce recommends that you understand the JavaScript in an s-control before relying on access restriction for s-control security.
 - If an installed package has Restricted API access, upgrades will be successful only if the upgraded version does not contain any s-controls. If s-controls are present in the upgraded version, you must change the currently installed package to Unrestricted API access.

Managing API and Dynamic Apex Access in Packages

API Access is a package setting that controls the dynamic Apex and API access that s-controls and other package components have to standard and custom objects. The setting displays for both the developer and installer on the package detail page. With this setting:

- The developer of an AppExchange package can restrict API access for a package before uploading it to Force.com AppExchange. Once restricted, the package components receive Apex and API sessions that are restricted to the custom objects in the package. The developer can also enable access to specific standard objects, and any custom objects in other packages that this package depends on.
- The installer of a package can accept or reject package access privileges when installing the package to his or her organization.
- After installation, an administrator can change Apex and API access for a package at any time. The installer can also enable access on additional objects such as custom objects created in the installer's organization or objects installed by unrelated packages.

Setting API and Dynamic Apex Access in Packages

To change package access privileges in a package you or someone in your organization has created:

1. From Setup, click **Create > Packages**.
2. Select a package.
3. The **API Access** field displays the current setting, **Restricted** or **Unrestricted**, and a link to either **Enable Restrictions** or **Disable Restrictions**. If **Read**, **Create**, **Edit**, and **Delete** access are not selected in the API access setting for objects, users do not have access to those objects from the package components, even if the user has the "Modify All Data" and "View All Data" permissions.

Use the **API Access** field to:

Enable Restrictions

This option is available only if the current setting is **Unrestricted**. Select this option if you want to specify the dynamic Apex and API access that package components have to standard objects in the installer's organization. When you select this option, the Extended Object Permissions list is displayed. Select the **Read**, **Create**, **Edit**, or **Delete** checkboxes to enable access for each object in the list. This selection is disabled in some situations. Click **Save** when finished. For more information about choosing the **Restricted** option, including information about when it is disabled, see [Considerations for API and Dynamic Apex Access in Packages](#) on page 52.

Disable Restrictions

This option is available only if the current setting is **Restricted**. Select this option if you do not want to restrict the Apex and API access privileges that the components in the package have to standard and custom objects. This option gives all the components in the package the same API access as the user who is logged in. For example, if a user can access accounts, an Apex class in the package that accesses accounts would succeed when triggered by that user.

Restricted

Click this link if you have already restricted API access and wish to edit the restrictions.

EDITIONS

Available in:

- Group
- Professional
- Enterprise
- Performance
- Unlimited
- Developer

USER PERMISSIONS

To edit API and dynamic Apex access for a package you have created or installed:

- "Create AppExchange packages"

To accept or reject package API and dynamic Apex access for a package as part of installation:

- "Download AppExchange packages"

Accepting or Rejecting API and Dynamic Apex Access Privileges During Installation

To accept or reject the API and dynamic Apex access privileges for a package you are installing:

- Start the installation process on Force.com AppExchange.
- In **Approve API Access**, either accept by clicking **Next**, or reject by clicking **Cancel**. Complete the installation steps if you have not canceled.

Changing API and Dynamic Apex Access Privileges After Installation

To edit the package API and dynamic Apex access privileges after you have installed a package:

1. From Setup, click **Installed Packages**.
2. Click the name of the package you wish to edit.
3. The **API Access** field displays the current setting, **Restricted** or **Unrestricted**, and a link to either **Enable Restrictions** or **Disable Restrictions**. If **Read**, **Create**, **Edit**, and **Delete** access are not selected in the API access setting for objects, users do not have access to those objects from the package components, even if the user has the “Modify All Data” and “View All Data” permissions.

Use the **API Access** field to:

Enable Restrictions

This option is available only if the current setting is **Unrestricted**. Select this option if you want to specify the dynamic Apex and API access that package components have to standard objects in the installer's organization. When you select this option, the Extended Object Permissions list is displayed. Select the **Read**, **Create**, **Edit**, or **Delete** checkboxes to enable access for each object in the list. This selection is disabled in some situations. Click **Save** when finished. For more information about choosing the **Restricted** option, including information about when it is disabled, see [Considerations for API and Dynamic Apex Access in Packages](#) on page 52.

Disable Restrictions

This option is available only if the current setting is **Restricted**. Select this option if you do not want to restrict the Apex and API access privileges that the components in the package have to standard and custom objects. This option gives all the components in the package the same API access as the user who is logged in. For example, if a user can access accounts, an Apex class in the package that accesses accounts would succeed when triggered by that user.

Restricted

Click this link if you have already restricted API access and wish to edit the restrictions.

Configuring Default Package Versions for API Calls

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. Publishers can use package versions to evolve the components in their managed packages gracefully by releasing subsequent package versions without breaking existing customer integrations using the package.

Default package versions for API calls provide fallback settings if package versions are not provided by an API call. Many API clients do not include package version information, so the default settings maintain existing behavior for these clients.

You can specify the default package versions for enterprise API and partner API calls. The enterprise WSDL is for customers who want to build an integration with their Salesforce organization only. It is strongly typed, which means that calls operate on objects and fields with specific data types, such as `int` and `string`. The partner WSDL is for customers, partners, and ISVs who want to

EDITIONS

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer**, Editions

USER PERMISSIONS

To configure default package versions for API calls:

- “Customize Application”

build an integration that can work across multiple Salesforce organizations, regardless of their custom objects or fields. It is loosely typed, which means that calls operate on name-value pairs of field names and values instead of specific data types.

You must associate the enterprise WSDL with specific package versions to maintain existing behavior for clients. There are options for setting the package version bindings for an API call from client applications using either the enterprise or partner WSDL. The package version information for API calls issued from a client application based on the enterprise WSDL is determined by the first match in the following settings.

1. The PackageVersionHeader SOAP header.
2. The SOAP endpoint contains a URL with a format of `serverName/services/Soap/c/api_version/ID` where `api_version` is the version of the API, such as 34.0, and `ID` encodes your package version selections when the enterprise WSDL was generated.
3. The default enterprise package version settings.

The partner WSDL is more flexible as it is used for integration with multiple organizations. If you choose the Not Specified option for a package version when configuring the default partner package versions, the behavior is defined by the latest installed package version. This means that behavior of package components, such as an Apex trigger, could change when a package is upgraded and that change would immediately impact the integration. Subscribers may want to select a specific version for an installed package for all partner API calls from client applications to ensure that subsequent installations of package versions do not affect their existing integrations.

The package version information for partner API calls is determined by the first match in the following settings.

1. The PackageVersionHeader SOAP header.
2. An API call from a Visualforce page uses the package versions set for the Visualforce page.
3. The default partner package version settings.

To configure default package versions for API calls:

1. From Setup, click **Develop > API**.
2. Click **Configure Enterprise Package Version Settings** or **Configure Partner Package Version Settings**. These links are only available if you have at least one managed package installed in your organization.
3. Select a **Package Version** for each of your installed managed packages. If you are unsure which package version to select, you should leave the default selection.
4. Click **Save**.



Note: Installing a new version of a package in your organization does not affect the current default settings.

About the Partner WSDL

The Partner Web Services WSDL is used for client applications that are metadata-driven and dynamic in nature. It is particularly—but not exclusively—useful to Salesforce partners who are building client applications for multiple organizations. As a loosely typed representation of the Salesforce data model that works with name-value pairs of field names and values instead of specific data types, it can be used to access data within any organization. This WSDL is most appropriate for developers of clients that can issue a query call to get information about an object before the client acts on the object. The partner WSDL document only needs to be downloaded and consumed once per version of the API.

For more information about the Partner WSDL, see [Using the Partner WSDL](#) in the *SOAP API Developer's Guide*.

Generating an Enterprise WSDL with Managed Packages

If you are downloading an enterprise WSDL and you have managed packages installed in your organization, you need to take an extra step to select the version of each installed package to include in the generated WSDL. The enterprise WSDL is strongly typed, which means that it contains objects and fields with specific data types, such as `int` and `string`.

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. Publishers can use package versions to evolve the components in their managed packages gracefully by releasing subsequent package versions without breaking existing customer integrations using the package. A subscriber can select a package version for each installed managed package to allow their API client to continue to function with specific, known behavior even when they install subsequent versions of a package. Each package version can have variations in the composition of its objects and fields, so you must select a specific version when you generate the strongly typed WSDL.

To download an enterprise WSDL when you have managed packages installed:

1. From Setup, click **Develop > API**.
2. Click **Generate Enterprise WSDL**.
3. Select the `Package Version` for each of your installed managed packages. If you are unsure which package version to select, you should leave the default, which is the latest package version.
4. Click **Generate**.
5. Use the **File** menu in your browser to save the WSDL to your computer.
6. On your computer, import the local copy of the WSDL document into your development environment.

Note the following in your generated enterprise WSDL:

- Each of your managed package version selections is included in a comment at the top of the WSDL.
- The generated WSDL contains the objects and fields in your organization, including those available in the selected versions of each installed package. If a field or object is added in a later package version, you must generate the enterprise WSDL with that package version to work with the object or field in your API integration.
- The SOAP endpoint at the end of the WSDL contains a URL with a format of `serverName/services/Soap/c/api_version/ID` where *api_version* is the version of the API, such as 34.0, and *ID* encodes your package version selections when you communicate with Salesforce.

You can also select the default package versions for the enterprise WSDL without downloading a WSDL at **Develop > API > Generate Enterprise WSDL**. Default package versions for API calls provide fallback settings if package versions are not provided by an API call. Many API clients do not include package version information, so the default settings maintain existing behavior for these clients.

EDITIONS

Available in: **Enterprise, Performance, Unlimited, and Developer**, Editions

USER PERMISSIONS

To download a WSDL:

- “Customize Application”

Working with External Services

You might want to update your Salesforce data when changes occur in another service. Likewise, you might also want to update the data in another service (outside of Salesforce) based on changes to your Salesforce data. Salesforce provides ways of doing both of these transactions. For example, you might want to send a mass email to more contacts and leads than Salesforce allows. To do so, you can use an external mail service that allows users to build a recipient list of names and email addresses using the contact and lead information in your Salesforce organization.

An app built on the Force.com platform can connect with an external service in many different ways. For example:

- You can create a custom link or custom formula field that passes information to an external service.
- You can use the Force.com API to transfer data in and out of Salesforce.
- You can use an Apex class that contains a Web service method.

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 will fail. For information on registering components, see “Configuring Remote Settings” in the Salesforce Help.

 **Warning:** Do not store usernames and passwords within any external service.

Provisioning External Services

If your app links to an external service, users who install the app must be signed up to use the service. Provide access in one of two ways:

- Access by all active users in an organization with no real need to identify an individual
- Access on a per user basis where identification of the individual is important

The Salesforce service provides two globally unique IDs to support these options. The user ID identifies an individual and is unique across all organizations. User IDs are never reused. Likewise, the organization ID uniquely identifies the organization.

Avoid using email addresses, company names, and Salesforce usernames when providing access to an external service. Usernames can change over time and email addresses and company names can be duplicated.

If you are providing access to an external service, we recommend the following:

- Use Single Sign-On (SSO) techniques to identify new users when they use your service.
- For each point of entry to your app, such as a custom link or web tab, include the user ID in the parameter string. Have your service examine the user ID to verify that the user ID belongs to a known user. Include a session ID in the parameter string so that your service can read back through the Force.com API and validate that this user has an active session and is authenticated.
- Offer the external service for any known users. For new users, display an alternative page to collect the required information.
- Do not store passwords for individual users. Besides the obvious security risks, many organizations reset passwords on a regular basis, which requires the user to update the password on your system as well. We recommend designing your external service to use the user ID and session ID to authenticate and identify users.
- If your application requires asynchronous updates after a user session has expired, dedicate a distinct administrator user license for this.

Architectural Considerations for Group and Professional Editions

Salesforce CRM is offered in five tiers, or editions:

- Group Edition (GE)
- Professional Edition (PE)
- Enterprise Edition (EE)
- Unlimited Edition (UE)
- Performance Edition (PXE)

 **Note:** Contact Manager Edition (CME) does not allow AppExchange package installations, so it won't be discussed here.

If you plan to sell your app to existing Salesforce customers, it's important to understand the differences between these editions because they will affect the design of your app. It's convenient to think about them in clusters, GE/PE and EE/UE/PXE, as the editions in each cluster have similar functionality. For example, you might only want to support EE/UE/PXE if your app requires certain objects and features

that aren't available in GE/PE. Also, instead of a single solution that supports all editions, you can have a tiered offering. This would consist of a basic solution for GE/PE and an advanced one for EE/UE/PXE customers that takes advantage of the additional features.

EE/UE/PXE have the most robust functionality. They support Force.com platform licenses in addition to Salesforce CRM licenses. If your app doesn't require Salesforce CRM features (such as Leads, Opportunities, Cases, etc.), Force.com platform licenses provide you with the most flexibility in deploying your app to users who might not normally be Salesforce users. Your app is still subject to the edition limits and packaging rules.

GE/PE don't contain all of the functionality that you can build in a Developer Edition (DE). Therefore, an application developed in your DE organization might not install in a GE/PE organization. If you're designing an application to work specifically in GE/PE, you must be aware of how these editions differ.

There are a number of other considerations to keep in mind when deciding whether to support these editions. Force.com platform licenses cannot be provisioned in GE/PE organizations. This means that only existing Salesforce CRM users can use your app. There are some features that aren't available in GE/PE. There are several special permissions available to eligible partner apps that overcome these limitations.

See the following sections for available features, limits, and other design considerations.

- [Features Available in Group and Professional Editions](#)
- [Limits for Group and Professional Editions](#)
- [Access Control in Group and Professional Editions](#)
- [Using Apex in Group and Professional Editions](#)
- [API Access in Group and Professional Editions](#)
- [Designing Your App to Support Multiple Editions](#)
- [Sample Design Scenarios](#)

Features Available in Group and Professional Editions

The easiest way to determine which features and objects are available in a particular edition is by reviewing the [Edition Comparison Table](#). You can also look up which editions support a specific feature or object by searching the online help. It's important that you check these resources before you start designing your app, so you can make an informed decision on which editions to target. Once you've finished building your app, we recommend you test it by installing your package in GE and PE test organizations to ensure everything functions properly.

The following table shows the key features that are different between GE, PE, and DE.

Feature	Group Edition	Professional Edition	Developer Edition
Assets	No	For additional cost	Yes
Campaigns	No	Yes	Yes
Contracts	No	Yes	Yes
Forecasts	No	Yes	Yes
Ideas	No	Yes	Yes
Products	No	Yes	Yes
Solutions	No	Yes	Yes
Record Types	No	No	Yes

Feature	Group Edition	Professional Edition	Developer Edition
Permission Sets	No	No	Yes
Custom Profiles	No	No	Yes
Custom Report Types	No	Yes	Yes
Workflow and Approvals	No	For additional cost	Yes
Apex Code	See following note	See following note	Yes
Sharing Rules	No	Yes	Yes
API	See following note	See following note	Yes
Sites	No	No	Yes



Note: A client ID allows your app to use the API for integration to composite apps. For more information, see [Using Apex in Group and Professional Editions](#) and [API Access in Group and Professional Editions](#).

Limits for Group and Professional Editions

All Salesforce editions have limits that restrict the number of apps, objects, and tabs that can be used. For details on the limits for various editions, see the [Edition Limits Table](#).

For partners who are enrolled in the ISV Program, any managed package publicly posted on the AppExchange no longer counts against the apps/objects/tabs limits for your Salesforce Edition. This effectively means that ISV partners no longer have to worry about package installation failures because of apps/objects/tabs limits being exceeded. This feature is automatically enabled after your app passes the security review.

Access Control in Group and Professional Editions

Both GE and PE do not support custom profiles or field level security. As a result, field level security is also handled by the page layout for each object. Therefore, when a customer installs your app they will not be able to define which profiles have access to what and you will need to ensure that your design works for the Standard User Profile. Permission sets can be installed in GE/PE orgs but not updated.

Because field level security is handled by the page layout, any fields you want to be visible must be added to the page layout. This means that for fields to be accessible via the API or Visualforce, they must be added to the page layout.

Using Apex in Group and Professional Editions

Your app can contain business logic such as classes, triggers, email services, etc. written in Apex. As a general rule, Apex is not supported in GE/PE, so it will not run in these editions. However, Apex developed as part of an ISV app and included in a managed package can run in GE/PE, even though those editions do not support Apex by default.

You must be an eligible partner with salesforce.com and your app has to pass the security review. The appropriate permissions will automatically be enabled after you pass the security review.

Here are some important considerations for using Apex in GE/PE.

- GE/PE customers can't create or modify Apex in your app; they can only run the existing Apex.
- Your Apex code should not depend on features and functionality that exist only in DE, EE, UE, or PXE, or your app will fail to install.

- Make sure to use REST if you plan to expose an Apex method as a Web service. Apex classes that have been exposed as a SOAP Web service can't be invoked from an external web app in GE/PE.
- Using Apex to make Web service callouts is allowed in GE/PE. For instance, if you're planning to make a Web service callout to an external Web service, as long as the managed package is authorized, these classes will function in GE/PE.

API Access in Group and Professional Editions

API access is not normally supported in GE/PE organizations. However, once your app passes the security review, you're eligible to use some APIs for building composite applications.

- Currently, the standard Data SOAP and REST APIs are supported for GE and PE apps, and Metadata API is supported in PE apps. To request API access, see [How do I get an API token for my app?](#) Also, you can contact Salesforce to request that a connected app be whitelisted to use the REST API in GE or PE organizations.
- Other APIs, such as the Bulk API and Apex Methods exposed using the SOAP Web service, remain unavailable.
- REST based Web services can be enabled by connected app consumer whitelisting
- SOAP-based Web services, including Metadata API, can be enabled using an API token called a Client ID, which needs to be appended to your SOAP headers in integration calls. This special key enables your app to successfully make calls to GE/PE organizations for Data API and PE organizations for Metadata API, even if the customer does not have API access.

The Client ID has these properties.

1. You can't use the Client ID with the AJAX Toolkit in custom JavaScript, S-controls, or anywhere in your app where its value would be exposed to the end customer.
2. For development purposes, GE/PE organizations created via the Environment Hub already have the Metadata API and SOAP API (Data API) enabled. This allows you to develop and test your app before the security review. After your app passes the security review and you obtain an API token, test your app again to ensure it is working correctly.
3. The Client ID grants GE/PE access to the SOAP API, and PE access to the Metadata API. The Metadata API allows you to dynamically create various components you typically create in Setup. For instance, you can create a custom field dynamically in a PE organization with the API token.

This table shows which APIs are accessible when using GE/PE and the method of access.

API	Access to GE/PE
Web Services (SOAP)	Yes, with token
Apex Methods exposed as Web Services (SOAP)	No
Web Services (REST)	Yes, with connected app consumer whitelisting
Apex Methods exposed as Web Services (REST)	Yes, with connected app consumer whitelisting
Chatter REST API	Yes
Metadata API	Yes, with token
Bulk API	No
Data Loader Tool (uses SOAP Web Services)	No, cannot set the token

Accessing the REST API in Group and Professional Editions

The Force.com REST API provides you a powerful, convenient, and simple Web services API for interacting with Force.com. Qualified partners can request salesforce.com to enable your application for REST API calls to GE/PE organizations. To get access to the REST API, you must meet these conditions.

- Access to the Partner Community – If you're new, please learn about and join one of the ISV Partner Programs.
- Pass the security review – All applications enrolled in the AppExchange and/or OEM Program must go through a periodic security review.
- Access to Salesforce Developer Edition – If you don't already have access to a DE organization, you can get the Partner Developer Edition from the Environment Hub.

To request the REST API token:

1. Create a new connected app from your DE organization. Log in to salesforce.com with your developer account. From Setup, click **Create > Apps**, and click **New** in the Connected Apps section.



Note: We strongly recommend that you do this in an organization you will continue using for a long time, such as the one where you build your managed package or your Trialforce management organization (TMO).

2. Enter the information requested and click **Save**. Saving your app gives you the Consumer Key and Consumer Secret the app uses to communicate with Salesforce.
3. Submit a case from the Partner Community and provide your DE Org ID and the credentials for your connected app.

We'll evaluate your request and enable the appropriate permission. Once this is done, you'll receive a case notification from us. Please wait 24 hours to make sure the permission is completely activated. Your `client_id` (or Consumer Key) and `client_secret` (or Consumer Secret) will be checked against the information you submit via the case during the OAuth authentication. If it matches, the system will allow you to communicate with GE/PE.



Note:

- This permission is intended solely for REST API. It does not enable your application to use SOAP Web Services API, Bulk API, Metadata API, etc. for GE/PE.
- This permission is applied only to your application. We do not turn on the API in the GE/PE organization.

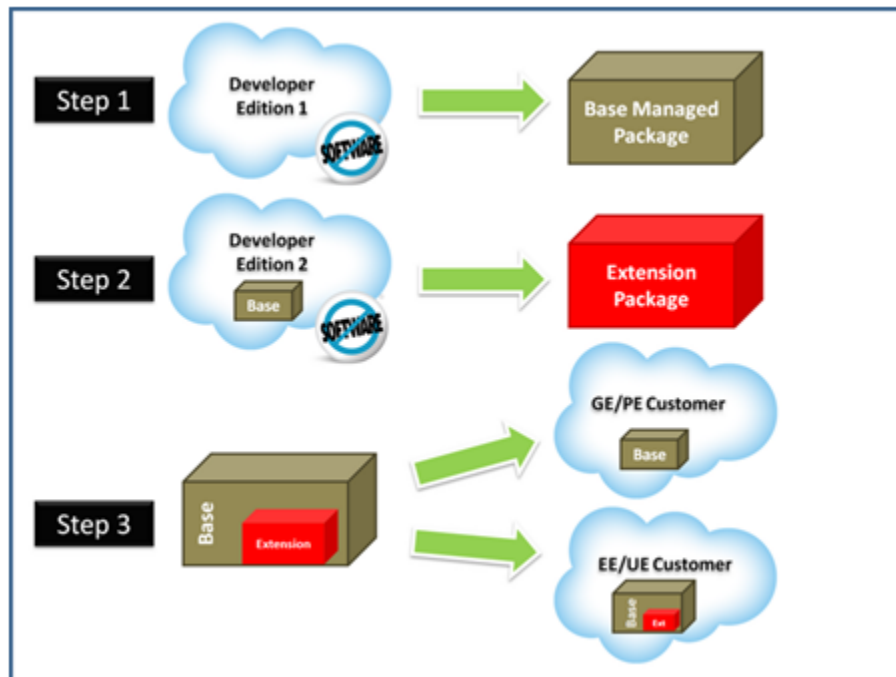
Designing Your App to Support Multiple Editions

Supporting multiple editions provides the opportunity to release richer versions of your app that can support more advanced features found in EE, UE, and PXE. There are two technologies that can be leveraged to support multiple editions. The first approach uses extension packages and the second leverages Dynamic Apex. There are benefits to both, so be sure to review both strategies before designing your app.

Supporting Multiple Editions using an Extension Package

This approach uses a base managed package that contains core app functionality. The base package only contains features supported in GE/PE. You then use a second managed package, or extension package, that extends and enhances the base package. The extension package adds additional features supported in EE/UE/PXE. For example, you have a warehouse application that tracks inventory and an extension to this app includes workflow (which isn't available in Group). Your Group and Professional Edition customers can install the base warehouse application, while your other customers install the base package and then the extension package with workflow components.

Using a base and extension package to support multiple editions



Using extension packages enables you to avoid multiple code sets and to upsell your customers. Upgrading a customer only requires installing the extension package.

Here is the process for creating an extension package.

1. Create your base managed package that leverages features supported by GE/PE.
2. Install your base managed package in a separate DE organization.
3. In this organization create your extension package that includes additional functionality supported in GE/PE. You can reference the base managed package to avoid duplicating functionality. Any component that references the base managed package will automatically trigger this package to be an extension package.

Since your extension package depends on your base package, it's important to spend time designing your app and the interfaces between the packages. For example, if the extension package needs to call an Apex class in the base package, you must make sure the desired Apex class is made global.

It's also important to consider the entire application lifecycle. For example, If you want to add new features, they should be included in the appropriate package. Updates to the base package should not break the extension package.

Supporting Multiple Editions using Dynamic Apex

Using dynamic Apex, dynamic SOQL, and dynamic DML, it's possible to create one managed package for all editions you plan to support without having to use extension packages. Your app behavior can change dynamically based on the features available in your customer's edition. This is useful when designing an app with the intent to support multiple editions.

Make sure that Apex, workflows, etc. in your package do not contain any strongly-typed reference to a feature that isn't supported by GE/PE. This can include adding a custom field on an unsupported standard object, such as Campaigns, or making an Apex reference to features like multi-currency or territory management. When you reference a feature in your package not supported by GE/PE, this package dependency will cause the installation to fail.

Instead, if you use dynamic Apex to first check if these features are available before referencing them, you can install your managed package in GE/PE. The important piece to consider is you must code your Dynamic Apex in a way that can support both use cases. This ensures that if your customer doesn't have a specific feature or object, your app will still function.

Sample Design Scenarios for Group and Professional Editions

Here are some scenarios to help you understand when and how to build for GE and PE.

Scenario 1: You want to build an app that uses record types

Since record types are not available in GE or PE, decide if you want to support these editions. Assuming you do, you can build a base managed package that does not include record types. After uploading this managed package in a released state, you can install it into another DE organization to start building the extension. Your extension can add record types that your EE, UE, and PXE customers can install and leverage.

Scenario 2: You want to build an app with 80 custom objects and 25 custom tabs

Typically this scenario would present a problem for GE and PE because of their objects and tabs limit. However, if you make your app available on the AppExchange, it does not count toward custom objects, tabs, and apps limits. So even if your app has 80 custom objects and 25 custom tabs, it installs and works normally in GE and PE.

Scenario 3: You want to build an app that makes Apex callouts to a Web service

Apex doesn't generally run in GE or PE, but if you get your managed package authorized during the security review, your Apex executes as expected. So for this scenario you build your Apex callout to invoke your external Web service and then include this class in your package.

Scenario 4: You want to build an app that leverages Campaigns

Campaigns are not supported by default in GE. For this scenario, you have two options.

- Option 1 - Build a base managed package that does not reference Campaigns. Once complete, upload and install into another DE organization to build the Campaign functionality as an extension package. Now your GE customers can install the base, while the rest can also install the extension to get extra features.
- Option 2 - This option requires only one package if you use Dynamic Apex as the only reference to Campaigns (as described earlier) and do not include a custom field on the Campaign. Your app can then be installed in GE and higher. If Campaigns is in your customer's edition, then your Dynamic Apex can manipulate Campaigns as expected.

Scenario 5: You want to build a composite app where you receive inbound API calls

You have a separate hosted app that you want to integrate with Salesforce, so you need to make API calls to GE and PE customers. Such calls are not possible by default, but if you are an eligible partner, request a special API Token that allows your SOAP calls to integrate with GE and PE. Be sure to embed the Client ID in the SOAP header of your external code.

Connected Apps Overview

USER PERMISSIONS

To read:	"Customize Application"
To create, update, or delete:	"Customize Application" AND either "Modify All Data" OR "Manage Connected Apps"
To update all fields except Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application"

EDITIONS

Connected Apps can be created in: **Group, Professional, Enterprise, Performance, Unlimited,** and **Developer** Editions

Connected Apps can be installed in: **All** Editions

To update Profiles, Permission Sets, and Service Provider SAML “Customize Application” AND “Modify All Data” Attributes:

To uninstall:

“Download AppExchange Packages”

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, connected apps allow administrators to set various security policies and have explicit control over who may use the corresponding applications.

A developer or administrator defines a connected app for Salesforce by providing the following information.

- Name, description, logo, and contact information
- A URL where Salesforce can locate the app for authorization or identification
- The authorization protocol: OAuth, SAML, or both
- Optional IP ranges where the connected app might be running
- Optional information about mobile policies the connected app can enforce

For connected apps that use OAuth service providers, define the OAuth scopes and callback URL for the connected app. In return, Salesforce provides an OAuth Consumer Key and a Consumer Secret for authorizing the connected app.

For connected apps that use SAML service providers, define the Entity ID, ACS (assertion consumer service) URL, Subject Type, Name ID Format and Issuer (these should be available from the service provider) for authorizing the connected app.

There are two deployment modes:

- The app is created and used in the same organization. This is a typical use case for IT departments, for example.
- The app is created in one organization and installed on other organizations. This is how an entity with multiple organizations or an ISV would use connected apps.

Administrators can install the connected app into their organization, enable SAML authentication, and use profiles, permission sets, and IP range restrictions to control which users can access the application. They can set the connected app to be exposed as a canvas app for tighter integration with the Salesforce UI. Administrators can also uninstall the connected app and install a newer version when a developer updates the remote app and notifies administrators that there is a new version available.



Note: In a Group Edition organization, you can’t manage individual user access using profiles. However, you can set policies when you edit an OAuth connected app’s settings in a Group Edition organization to control access to the connected app for all users.

And, Salesforce-managed connected apps packages like those for the Salesforce1 downloadable apps can’t be uninstalled. They are automatically updated when the next user’s session refreshes.

Connected apps can be added to managed packages, only. Connected apps are not supported for unmanaged packages.

Creating a Connected App

USER PERMISSIONS

To read:	"Customize Application"
To create, update, or delete:	"Customize Application" AND either "Modify All Data" OR "Manage Connected Apps"
To update all fields except Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application"
To update Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application" AND "Modify All Data"
To uninstall:	"Download AppExchange Packages"

EDITIONS

Connected Apps can be created in: **Group, Professional, Enterprise, Performance, Unlimited,** and **Developer** Editions

Connected Apps can be installed in: **All** Editions

To create a connected app:

1. From Setup, click **Create > Apps**.
2. In the Connected Apps section, click **New**.


The information you enter to create a connected app is divided into these parts:

- [Basic Information](#)
- [API \(Enable OAuth Settings\)](#)
- [Web App Settings](#)
- [Mobile App Settings](#)
- [Canvas App Settings](#)

You can create a connected app without specifying any authorization, canvas, or mobile settings. This kind of connected app behaves like a "bookmark" to the specified URL that appears in the user's App Launcher and the drop-down app menu. Simply enter basic information and provide a `start URL` in the Web App Settings. If the destination requires authentication, the service hosting the destination URL should prompt users to provide login credentials when they navigate to it.

When you've finished entering the information, click **Save** to save your new app. You can now publish your app, make further edits, or delete it. If you're using OAuth, saving your app gives you two new values the app uses to communicate with Salesforce:

- **Consumer Key:** A value used by the consumer to identify itself to Salesforce. Referred to as `client_id` in OAuth 2.0.
- **Consumer Secret:** A secret used by the consumer to establish ownership of the consumer key. Referred to as `client_secret` in OAuth 2.0.

 **Important:** As you update fields for a connected app, be aware that changes to some fields immediately apply to all installed versions of the connected app, too. These are version-independent fields that bypass the packaging or installation lifecycle. Users of the connected app will see things like the description change. The following fields have this version-independent behavior.

- Description
- Info URL
- Logo Image URL

- Callback URL

Basic Information

Specify basic information about your app in this section, including the app name, logo, and contact information.

1. Enter the `Connected App Name`. This name is displayed in the list of connected apps.



Note: The name must be unique for the current connected apps in your organization. You can reuse the name of a deleted connected app if the connected app was created using the Spring '14 release or later. You cannot reuse the name of a deleted connected app if the connected app was created using an earlier release.

2. Enter the `API Name`, used when referring to your app from a program. It defaults to a version of the name without spaces. Only letters, numbers, and underscores are allowed, so you'll need to edit the default name if the original app name contained any other characters.
3. Provide the `Contact Email` that Salesforce should use for contacting you or your support team. This address is not provided to administrators installing the app.
4. Provide the `Contact Phone` for Salesforce to use in case we need to contact you. This number is not provided to administrators installing the app.
5. Enter a `Logo Image URL` to display your logo in the list of connected apps and on the consent page that users see when authenticating. The URL must use HTTPS. The logo image can't be larger than 125 pixels high or 200 pixels wide, and must be in the GIF, JPG, or PNG file format with a 100 KB maximum file size. The default logo is a cloud. You have several ways to add a custom logo.
 - You can upload your own logo image by clicking **Upload logo image**. Select an image from your local file system that meets the size requirements for the logo. When your upload is successful, the URL to the logo appears in the `Logo Image URL` field. Otherwise, make sure the logo meets the size requirements.
 - You can also select a logo from the samples provided by clicking **Choose one of our sample logos**. The logos available include ones for Salesforce apps, third-party apps, and standards bodies. Click the logo you want, and then copy and paste the displayed URL into the `Logo Image URL` field.
 - You can use a logo hosted publicly on Salesforce servers by uploading an image that meets the logo file requirements (125 pixels high or 200 pixels wide, maximum, and in the GIF, JPG, or PNG file format with a 100 KB maximum file size) as a document using the Documents tab. Then, view the image to get the URL, and enter the URL into the `Logo Image URL` field.
6. Enter an `Icon URL` to display a logo on the OAuth approval page that users see when they first use your app. The logo should be 16 pixels high and wide, on a white background. Sample logos are also available for icons.
 You can select an icon from the samples provided by clicking **Choose one of our sample logos**. Click the icon you want, and then copy and paste the displayed URL into the `Icon URL` field.
7. If there is a Web page with more information about your app, provide a `Info URL`.
8. Enter a `Description` to be displayed in the list of connected apps.

Prior to Winter '14, the `Start URL` and `Mobile Start URL` were defined in this section. These fields can now be found under Web App Settings and Mobile App Settings below.

API (Enable OAuth Settings)

This section controls how your app communicates with Salesforce. Select `Enable OAuth Settings` to configure authentication settings.

1. Enter the `Callback URL` (endpoint) that Salesforce calls back to your application during OAuth; it's the OAuth `redirect_uri`. Depending on which OAuth flow you use, this is typically the URL that a user's browser is redirected to after successful authentication. As this URL is used for some OAuth flows to pass an access token, the URL must use secure HTTP (HTTPS) or a custom URI scheme. If you enter multiple callback URLs, at run time Salesforce matches the callback URL value specified by the application with one of the values in `Callback URL`. It must match one of the values to pass validation.
2. If you're using the JWT OAuth flow, select `Use Digital Signatures`. If the app uses a certificate, click **Choose File** and select the certificate file.
3. Add all supported OAuth scopes to `Selected OAuth Scopes`. These scopes refer to permissions given by the user running the connected app, and are followed by their OAuth token name in parentheses:

Access and manage your Chatter feed (`chatter_api`)

Allows access to Chatter REST API resources only.

Access and manage your data (`api`)

Allows access to the logged-in user's account using APIs, such as REST API and Bulk API. This value also includes `chatter_api`, which allows access to Chatter REST API resources.

Access your basic information (`id, profile, email, address, phone`)

Allows access to the Identity URL service.

Access custom permissions (`custom_permissions`)

Allows access to the custom permissions in an organization associated with the connected app, and shows whether the current user has each permission enabled.

Allow access to your unique identifier (`openid`)

Allows access to the logged in user's unique identifier for OpenID Connect apps.

Full access (`full`)

Allows access to all data accessible by the logged-in user, and encompasses all other scopes. `full` does not return a refresh token. You must explicitly request the `refresh_token` scope to get a refresh token.

Perform requests on your behalf at any time (`refresh_token, offline_access`)

Allows a refresh token to be returned if you are eligible to receive one. This lets the app interact with the user's data while the user is offline. The `refresh_token` scope is synonymous with `offline_access`.

Provide access to custom applications (`visualforce`)

Allows access to Visualforce pages.

Provide access to your data via the Web (`web`)

Allows the ability to use the `access_token` on the Web. This also includes `visualforce`, allowing access to Visualforce pages.

If your organization had the `No user approval required for users in this organization` option selected on your remote access prior to the Spring '12 release, users in the same organization as the one the app was created in still have automatic approval for the app. The read-only `No user approval required for users in this organization` checkbox is selected to show this condition. For connected apps, the recommended procedure after you've created an app is for administrators to install the app and then set `Permitted Users` to `Admin-approved users`. If the remote access option was not checked originally, the checkbox doesn't display.

Web App Settings

Enter a `Start URL` for your app to direct users to a specific location after they've authenticated. If you don't enter a `Start URL`, users will be 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 don't need to enter a value for this field. The `Canvas App URL` field contains the URL that gets called for the connected app.


If your connected app will use a SAML service provider, select `Enable SAML`. Enter the `Entity Id`, `ACS URL`, `Subject Type`, `Name ID Format` and `Issuer`, available from your service provider. Select `Verify Request Signatures` if the service provider gave you a security certificate. Browse your system for the certificate. This is only necessary if you plan to initiate logging into Salesforce from the service provider and the service provider signs their SAML requests.


 **Important:** If you upload a certificate, all SAML requests must be signed. If no certificate is uploaded, all SAML requests are accepted.

Optionally, select `Encrypt SAML Response` to upload a certificate and select an encryption method for encrypting the assertion. Valid encryption algorithm values are `AES-128` (128-bit key), `AES-256` (256-bit key), and `Triple-DES` (Triple Data Encryption Algorithm).

Mobile App Settings

1. Enter the `Mobile Start URL` to direct users to a specific location when the app is accessed from a mobile device. If you don't enter a `Mobile Start URL`, users will be sent to the `Start URL` defined under Web App Settings. If the connected app you're creating is a canvas app, you don't need to enter a value for this field. The Canvas App URL field contains the URL that gets called for the connected app.
2. Select `PIN Protect`, if your app supports PIN protection. This gives an administrator the option of setting the session timeout and PIN length for mobile applications after installing the connected app. PIN protection is automatically supported by the Salesforce Mobile SDK (https://developer.salesforce.com/page/Mobile_SDK). You can also implement it manually by reading the `mobile_policy` object from the user's Identity URL.
3. Specify the `App Platform` by choosing iOS or Android from the drop-down list.
4. Specify the supported device form factor(s) for the mobile app from the `Restrict to Device Type` drop-down list. The possible values are Phone, Tablet, or Mini-Tablet. If the app is universal (that is, supports all form factors), don't choose any value.
5. Enter the `App Version` number of the mobile app.
6. Enter the `Minimum OS Version` required for the app.
7. Select `Private App` to confirm this app is for internal (non-public) distribution only. This is required because Apple doesn't allow distribution of public mobile apps outside of its app store.
8. If the mobile app is private, specify the location of the `Mobile App Binary` file. This is an IPA file for iOS and an APK file for Android.
9. For iOS apps only:
 - a. Specify the location of the Application Icon. This is the icon displayed during download and installation of the app on an iOS device.
 - b. Specify the iOS Bundle Identifier.
10. If the mobile connected app is a public app and you haven't uploaded its binary file to Salesforce, enter the `App Binary URL` here.

 **Note:** For iOS 7 and higher, you must specify the same bundle identifier that you used for developing the app in XCode. Otherwise, the end user will see two app icons on app installation.

 **Note:** If you remove mobile integration from a new version of an existing connected app, mobile integration is no longer included in any version of the connected app. For example, imagine publishing a package containing version 1.0 of your connected app with mobile integration. Then remove mobile integration from the app, repackage it, and publish it as version 1.1. If a customer installs the earlier package with version 1.0 at this point, the version 1.0 connected app will not contain mobile integration.

Your connected app can receive push notifications if:

- Your app is built with Salesforce Mobile SDK.
- Your app implements the Mobile SDK push notification protocol for your platform.
- You are a registered developer with the mobile platform provider (Apple or Google).
- Your app is registered with Apple Push Notification Service (APNS) for iOS push notifications or with Google Cloud Messaging (GCM) for Android push notifications.
- You've implemented Apex handlers for push notifications.



Note: A push-enabled connected app can support only one mobile platform. If you provide Android and iOS versions of your mobile app and need to support push notifications on both versions, create a connected app for each platform.

To learn how to fulfill these requirements, see the [Salesforce Mobile Push Notifications Implementation Guide](#).

To configure push notifications for APNS (iOS):

1. Select **Push Messaging Enabled**.
2. For Supported Push Platform, select **Apple**.
3. Select the Apple environment that is valid for your APNS push notifications certificate.
4. For Certificate, select the `.p12` certificate file that you received from APNS when you registered your app for push notifications (for example, `appkey.p12`).
5. Enter the password for your `.p12` certificate file.

To configure push notifications for GCM (Android):

1. Select **Push Messaging Enabled**.
2. For Supported Push Platform, select **Android GCM**.
3. For Key for Server Applications (API Key), enter the key that you obtained during developer registration with Google.

To change the mobile platform that you've configured for push notifications:


1. Deselect **Push Messaging Enabled**.
2. Save the connected app, and then click **Edit**.
3. Change **App Platform** and associated values in Mobile Settings to reflect the new platform.
4. Reconfigure push notifications for the new platform.


Canvas App Settings

Two types of canvas apps are available:

- *Canvas apps* that are installed by the organization administrator.
 - *Canvas personal apps* that are installed by end users across organizations. Users access a canvas personal app from the Chatter tab, and are prompted to allow the app to connect to their Salesforce data. These steps include optionally making an app a canvas personal app. For more information, see "Canvas Personal Apps" in the *Force.com Canvas Developer's Guide*.
1. If your connected app will be exposed as a canvas app, select `Force.com Canvas`.
 2. Type the `Canvas App URL` to the third-party app. The user is directed to this URL when they click the link to your canvas app.
 3. Select an `Access Method`. This specifies how the canvas app initiates the OAuth authentication flow.
 - `Signed Request (POST)`: OAuth authentication is used, but when the administrator installs the canvas app, they implicitly allow access for users. Therefore, the user won't be 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.

If your canvas app uses signed request authentication, then be sure you don't add `Perform requests on your behalf at any time` to the `Selected OAuth Scopes`.

- `OAuth Webflow (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.
4. If you're using SAML single sign-on (SSO) for canvas app authentication, select the `SAML Initiation Method` field. This field is enabled if you select `Enable SAML` in the Web App Settings section. The options for this field are:
 - **Identity Provider Initiated**—Salesforce makes the initial request to start the SSO flow.
 - **Service Provider Initiated**—The canvas app starts the SSO flow after the app is invoked.
 5. Under `Locations`, select where the canvas app appears to users.
 - **Chatter Feed**—The canvas app appears in the feed. If this option is selected, you must create a CanvasPost feed item and ensure that the current user has access to the canvas app.
 - **Chatter Tab**—The canvas app appears in the app navigation list on the Chatter tab. If this option is selected, the canvas app appears there automatically.
 - **Console**—The canvas app appears in the footer or sidebars of a Salesforce console. If this option is selected, you must choose where the canvas app appears in a console by adding it as a custom console component.
 - **Layouts and Mobile Cards**—The canvas app can appear on a page layout or a mobile card. If this option is selected, you choose where the canvas app appears by adding it to the page layout.
 - **Mobile Nav**—The canvas app is accessible from the navigation menu in Salesforce1.
-  **Note:** Canvas apps do not appear in the Salesforce1 navigation menu on Android mobile devices. To see canvas apps in the navigation menu on Android, log in to the Salesforce1 mobile browser app.
- **Open CTI**—The canvas app appears in the call control tool. If this option is selected, you must specify the canvas app in your call center's definition file for it to appear.
 - **Publisher**—The canvas app appears in the publisher. If this option is selected, you must also create a canvas custom quick action and add it to the global layout or to an object layout.
 - **Visualforce Page**—The canvas app can appear on a Visualforce page. If you add an `<apex:canvasApp>` component to expose a canvas app on a Visualforce page, be sure to select this location for the canvas app; otherwise, you'll receive an error.
6. Select `Create Actions Automatically` to create a global action for your canvas app. To create a global action for the canvas app, you must select `Publisher` under `Location`; otherwise, no global actions are created. You can also create the action manually at a later time.
 7. If you've implemented your own `Canvas.CanvasLifecycleHandler` Apex class, provide the class name in `Lifecycle Class`. Providing a `CanvasLifecycleHandler` Apex class lets you customize context information and add custom behavior to your canvas app.
 8. To make your app installable by end users, select the `Enable as a Canvas Personal App` checkbox. Chatter Tab is the only `Location` that supports canvas personal apps. For details about canvas personal apps, see "Canvas Personal Apps" in the *Force.com Canvas Developer's Guide*.

 **Note:** If you don't see the `Enable as a Canvas Personal App` setting, the administrator for the app's destination organization hasn't enabled canvas personal apps. For details about this requirement, see "Enabling Canvas Personal Apps within an Organization" in the *Force.com Canvas Developer's Guide*.

Edit, Package, or Delete a Connected App

USER PERMISSIONS

To read:	"Customize Application"
To create, update, or delete:	"Customize Application" AND either "Modify All Data" OR "Manage Connected Apps"
To update all fields except Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application"
To update Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application" AND "Modify All Data"
To uninstall:	"Download AppExchange Packages"

EDITIONS

Connected Apps can be created in: **Group, Professional, Enterprise, Performance, Unlimited,** and **Developer** Editions

Connected Apps can be installed in: **All** Editions

After creating a connected app, you can edit, package, or delete it.



Note: The name must be unique for the current connected apps in your organization. You can reuse the name of a deleted connected app if the connected app was created using the Spring '14 release or later. You cannot reuse the name of a deleted connected app if the connected app was created using an earlier release.

Editing a Connected App

You can update a connected app at any time. From Setup, click **Create > Apps**. Select a connected app name in the list and click **Edit**. Save your changes by clicking **Save**.

After you've created the connected app, you can go back to the detail page to specify the allowed IP ranges.

The IP ranges work with OAuth-enabled connected apps, not SAML-enabled connected apps, and specify valid IP addresses for the connected app.

Use the following steps to set the allowed IP range.

1. From Setup, click **Create > Apps**.
2. Select a connected app name in the list.
3. In the Trusted IP Range for OAuth Web server flow section, click **New**.
4. Enter a valid IP address in the **Start IP Address** field and a higher IP address in the **End IP Address** field.

You can enter multiple, discontinuous ranges by clicking **New** to enter each range.

You can allow specific users to access the connected app from outside of the Trusted IP Range, for OAuth-enabled connected apps. For example, to allow access to some users while traveling, set the connected app to **Relax IP Restrictions with second factor**. When a user attempts to use the connected app from outside this range, the user is prompted to provide a second factor of authentication, such as a token code. After a successful second factor authentication, the user can use the connected app from outside the Trusted IP Range.

1. From Setup, click **Manage Apps > Connected Apps**.
2. Click **Edit** next to the connected app name to display the values for the app.

3. In the IP Relaxation field, select `Relax IP Restrictions` in the drop-down list.



Note: If the `Enforce login IP ranges on every request` Session Settings option is enabled, it affects the IP relaxation behavior. For more information, see [Connected App IP Relaxation and Continuous IP Enforcement](#) on page 80.

After you've created the connected app, you can go back to the detail page and specify custom attributes. Custom attributes specify SAML metadata or specify OAuth parameters that are read at OAuth runtime.

1. From Setup, click **Create > Apps**.
2. Select a connected app name in the list.
3. In the Custom Attributes section, click **New**.

Each custom attribute must have a unique key and must use fields available from the **Insert Field** menu. For example, assign a key name, such as `country` and insert the field `$Organization.Country`. When using SAML, attributes are sent as SAML attribute statements. When using OAuth, attributes are available as a `custom_attributes` object in the user's Identity URL.

The following custom attributes are available for Salesforce1 connected apps.

Table 1: Salesforce1 Connected App for Android Custom Attributes

Attribute Key	Attribute Value	Description
CALL_HISTORY	<ul style="list-style-type: none"> • <code>DISABLED</code> • <code>ADMIN_DEFINED</code> • <code>SIMPLE</code> 	<ul style="list-style-type: none"> • If set to <code>DISABLED</code>, removes call logging from the navigation menu. • If set to <code>ADMIN_DEFINED</code>, enables native Android call logging. • If set to <code>SIMPLE</code>, enables Aura call logging.

Table 2: Salesforce1 Connected App for iOS Custom Attributes

Attribute Key	Attribute Value	Description
USE_ALTERNATE_USER_PROFILE	<ul style="list-style-type: none"> • <code>TRUE</code> • <code>FALSE</code> 	<ul style="list-style-type: none"> • If set to <code>TRUE</code>, enables Aura profile home. • If set to <code>FALSE</code>, enables native iOS profile home.
SHOW_OPEN_IN	<ul style="list-style-type: none"> • <code>FALSE</code> 	<ul style="list-style-type: none"> • If set to <code>FALSE</code>, disables users from sharing a file using a link to the file, or opening a file in a third-party app.

When defining custom attributes, wrap attribute values in quotation marks.



Important: As you update fields for a connected app, be aware that changes to some fields immediately apply to all installed versions of the connected app, too. These are version-independent fields that bypass the packaging or installation lifecycle. Users of the connected app will see things like the description change. The following fields have this version-independent behavior.

- Description
- Info URL

- Logo Image URL
- Callback URL

Packaging a Connected App

After creating a connected app or a new version of an existing app, package it to make it available to users on other Salesforce organizations. You add a connected app to a managed package in the same way as, and along with, other components such as custom objects, Visualforce pages, or Apex classes. This makes it easy to distribute a connected app to other Salesforce organizations. As a packageable component, connected apps can also take advantage of all other features of managed packages, such as listing on the AppExchange, push upgrades, post-install Apex scripts, license management, and enhanced subscriber support.



Note: You can only package a connected app from a Developer Edition organization. Connected apps can be added to managed packages, only. Connected apps are not supported for unmanaged packages.

Deleting a Connected App

To delete a connected app, click the **Connected App Name** in the list of apps. Click **Delete** on the editing page and confirm by clicking **Delete** again. Even though the app is removed from your list, you cannot reuse the app name.

If you delete a connected app that has been included in a package, the app remains available in the package until you update the package.



Note: If user provisioning has been configured for a connected app, you can't delete the connected app or uninstall a package that contains it until an administrator removes the user provisioning configuration details. Be aware that deselecting the `Enable User Provisioning` checkbox on the connected app detail page doesn't remove the configuration details from the organization. To remove the configuration details, see the instructions for Salesforce administrators in [this known issue](#).

Installing a Connected App

USER PERMISSIONS


To read:	"Customize Application"
To create, update, or delete:	"Customize Application" AND either "Modify All Data" OR "Manage Connected Apps"
To update all fields except Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application"
To update Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application" AND "Modify All Data"
To uninstall:	"Download AppExchange Packages"

EDITIONS

Connected Apps can be created in: **Group, Professional, Enterprise, Performance, Unlimited, and Developer** Editions

Connected Apps can be installed in: **All** Editions

You install a connected app by installing a managed package that includes the connected app as a component.

 **Note:** Connected apps created prior to Summer '13 can be installed using an install URL, as long as the connected app is not updated. Once the developer uploads a package with an updated version of the connected app, the install URL will no longer work.

View Connected App Details

USER PERMISSIONS

To read:	"Customize Application"
To create, update, or delete:	"Customize Application" AND either "Modify All Data" OR "Manage Connected Apps"
To update all fields except Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application"
To update Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application" AND "Modify All Data"
To uninstall:	"Download AppExchange Packages"


EDITIONS

Connected Apps can be created in: **Group, Professional, Enterprise, Performance, Unlimited,** and **Developer** Editions


Connected Apps can be installed in: **All** Editions

The Connected App Detail page shows you information about the connected app, including its version and scopes. You can edit and check usage of the connected app, and associate profiles and permissions with the app.

- Click **Edit** to change the app configuration on the Connected App Edit page.
- Click **Download Metadata** to get the service provider SAML login URLs and endpoints that are specific to your community or custom domain configuration. This button only appears if your organization is enabled as an Identity Provider, and only with connected apps that use SAML.
- Instead of downloading metadata, you can access the metadata via a URL in Metadata Discovery Endpoint. Your service provider can use this URL to configure single sign-on to connect to Salesforce.
- Click **View OAuth Usage** to see the usage report for connected apps in your organization.
- You can enable user provisioning for a connected app on this page. Once enabled, use the User Provisioning Wizard to configure or update the user provisioning settings. After you run the User Provisioning Wizard, the User Accounts section lets you manage the linkage between user accounts and their account settings on the third-party system, individually.
- Click **Manage Profiles** to select the profiles for the app from the Application Profile Assignment page. Select the profiles to have access to the app (except in Group Edition).

 **Important:** This option won't appear if the OAuth policy for **Permitted Users** is set to *All users may self-authorize* because this option isn't needed when users can authorize themselves.

- Click **Manage Permission Sets** to select the permission sets for the profiles for this app from the Application Permission Set Assignment page. Select the permission sets to have access to the app.

 **Important:** This option won't appear if the OAuth policy for **Permitted Users** is set to *All users may self-authorize* because this option isn't needed when users can authorize themselves.

- Click **New** in Service Provider SAML Attributes to create new attribute key/value pairs. You can also edit or delete existing attributes.

Only the users with at least one of the selected profiles or permission sets can run the app if you selected `Admin-Approved Users` for the `Permitted Users` value on the Connected App Edit page. If you selected `All Users` instead, profiles and permission sets are ignored.

Managing a Connected App

USER PERMISSIONS


To read:	"Customize Application"
To create, update, or delete:	"Customize Application" AND either "Modify All Data" OR "Manage Connected Apps"
To update all fields except Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application"
To update Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application" AND "Modify All Data"
To uninstall:	"Download AppExchange Packages"

EDITIONS

Connected Apps can be created in: **Group, Professional, Enterprise, Performance, Unlimited,** and **Developer** Editions

Connected Apps can be installed in: **All** Editions

To view and update properties of a connected app, find the app under Setup, in **Manage Apps > Connected Apps** and click **Edit** next to it. To view information, usage, and policies for a connected app, or add custom attributes, click the app's name.

 **Note:** Sessions refresh automatically between every 15 minutes and 12 hours while a user is in the app based upon the session `Timeout` value set for your organization; this is often undetected by the user.



Connected Apps Installed by Salesforce

Some Salesforce client apps are implemented as connected apps and automatically installed in your organization, such as Salesforce1 or Salesforce for Outlook. So you might see more connected apps in your list of installed apps than you expected.

These Salesforce connected apps are distributed in two managed packages: one for Salesforce1-related apps and one for non-Salesforce1-related apps. The list of included apps can change with each release. However, to simplify administration, each package is asynchronously installed in your organization the first time any user in the organization accesses one of these apps.

If you want to install (or reinstall) the Salesforce1 package for connected apps, proactively, you can [install it from the AppExchange](#).

The packages appear in Setup under the Installed Packages List.

Installed Packages									
Action	Package Name	Publisher	Version Number	Namespace Prefix	Status	Allowed Licenses	Used Licenses	Expiration Date	Install Date
Uninstall 	Salesforce Connected Apps	Salesforce.com	1.5	sf_com_apps	Free	N/A	N/A	N/A	10/24/2013 4:54 PM
Description This package contains Connected Applications for all the officially supported Salesforce client applications such as Touch, Salesforce for Outlook, Sa...									
Uninstall 	Salesforce1 and Chatter Apps	Salesforce.com	1.6	sf_chtr_apps	Free	N/A	N/A	N/A	9/25/2013 11:40 PM
Description This package contains Connected Applications for all the officially supported Salesforce1 and Chatter applications on your desktop and mobile devices!									

Click on each Package Name to see the list of components. The following are some of the components for the Salesforce Connected Apps package.

Package Components			
Action	Name	Parent Object	Type
	Forcecom IDE		Connected App
	Forcecom Migration Tool		Connected App
	Dataloader Bulk		Connected App
	Dataloader Partner		Connected App
	Salesforce Touch		Connected App
	Salesforce Mobile Dashboards		Connected App
	Workbench		Connected App
	Salesforce for Outlook		Connected App



Note: The Force.com IDE, Force.com Migration Tool, Dataloader Bulk, and Dataloader Partner are “wrapper” connected apps that use the SOAP API to connect to Salesforce, instead of OAuth like other connected apps. But, they use the connected apps framework to allow or deny users access to the apps in an organization.

The following are some of the components for the Salesforce1 and Chatter Apps package.

Package Components			
Action	Name	Parent Object	Type
	SalesforceA		Connected App
	Salesforce Chatterbox		Connected App
	Chatter for BlackBerry		Connected App
	Chatter for iOS		Connected App
	Chatter Desktop		Connected App
	Chatter for Android		Connected App

To manage these installed connected apps, from Setup, click **Manage Apps > Connected Apps** and you’ll see the automatically installed Salesforce connected apps appear in the list as managed package installed apps along with your other installed connected apps.

Action	Master Label +	Application Version	Permitted Users
Edit	Chatter Desktop	7.0	All users may self-authorize
Edit	Chatter Mobile for BlackBerry	7.0	All users may self-authorize
Edit	[Redacted]	1.0	All users may self-authorize
Edit	[Redacted]	1.0	Admin approved users are pre-authorized
Edit	Salesforce Files	5.0	All users may self-authorize
Edit	Salesforce1 for iOS	7.0	All users may self-authorize
Edit	Salesforce1/Chatter for Android	7.0	All users may self-authorize
Edit	SalesforceA	1.0	All users may self-authorize
Edit	[Redacted]	1.0	All users may self-authorize

Edit a Connected App

USER PERMISSIONS

To read:	"Customize Application"
To create, update, or delete:	"Customize Application" AND either "Modify All Data" OR "Manage Connected Apps"
To update all fields except Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application"
To update Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application" AND "Modify All Data"
To uninstall:	"Download AppExchange Packages"


EDITIONS

Connected Apps can be created in: **Group, Professional, Enterprise, Performance, Unlimited,** and **Developer** Editions

Connected Apps can be installed in: **All** Editions

You can modify settings and permissions for a connected app.

1. From Setup, click **Manage Apps > Connected Apps**.
 2. Click **Edit** next to the name of the app you want to modify. (To review information about an app on the connected app Detail page, click the app name.)
- The following OAuth policies are available for every OAuth-enabled connected app.
 - **Permitted Users** determines who can run the app.
 - **All Users may self-authorize:** Default. Anyone in the organization can self-authorize the app. This setting means that each user has to approve the app the first time they access it.
 - **Admin-approved users are pre-authorized:** Access is limited to those users with a profile or permission set specified, but these users don't need to approve the app before they can access it. In Group Edition, this setting prevents access to the app for all users. Manage profiles for the app by editing each profile's Connected App Access list (except in Group Edition). Manage permission sets for the app by editing each permission set's Assigned Connected Apps list.
 - ⚠ **Warning:** If you switch from **All Users may self-authorize** to **Admin-approved users are pre-authorized**, anyone currently using the app loses access, unless the user belongs to a permission set or profile that you have specified for the app.
 - 📝 **Note:** If the user's profile or permission set has the "Use Any API Client" user permission enabled, the **Admin-approved users are pre-authorized** policy can be bypassed. This user permission is available only if the "Admin Approved apps only" organization permission is enabled. The "Use Any API Client" user permission allows a non-Admin-approved user to access and run the app, even if the connected app's settings require Admin-approved users and the "Admin Approved apps only" organization permission is enabled. This permission scheme allows specific users, such as short-term contractors, to access a connected app temporarily.
 - **IP Relaxation** refers to the IP restrictions that the users of the connected app are subject to. IP ranges work with OAuth-enabled connected apps, not SAML-enabled connected apps. An administrator can choose to either enforce or bypass these restrictions by choosing one of the following options.
 - **Enforce IP restrictions:** Default. A user running this app is subject to the organization's IP restrictions, such as IP ranges set in the user's profile.

- `Relax IP restrictions with second factor`: A user running this app bypasses the organization's IP restrictions when either of these conditions are true:
 - The app has IP ranges whitelisted and is using the Web server OAuth authentication flow. Only requests coming from the whitelisted IPs are allowed.
 - The app has no IP range whitelist, is using the Web server or user-agent OAuth authentication flow, and the user successfully completes Identity Confirmation.
 - `Relax IP restrictions`: A user running this connected app is not subject to any IP restrictions.
-  **Note:** If the `Enforce login IP ranges on every request` Session Settings option is enabled, it affects the IP relaxation behavior. For more information, see [Connected App IP Relaxation and Continuous IP Enforcement](#) on page 80.
- `Refresh Token Policy` specifies the validity period for a refresh token. Refresh tokens are used by the OAuth-enabled connected app to obtain new sessions without requiring the user to provide their credentials. The connected app simply exchanges the refresh token for a new session. Using refresh token policies, administrators control how long a refresh token is used. Options include the following.
 - `Refresh token is valid until revoked`. This setting is the default behavior. It specifies that the token is used indefinitely, unless revoked by the user or administrator. Revoke tokens in a user's detail page under OAuth Connected Apps or in the OAuth Connected Apps Usage report.
 - `Immediately expire refresh token`. This setting specifies that the token is immediately invalid. The user can use the current session (access token) already issued, but cannot use the refresh token to obtain a new session.
 - `Expire refresh token if not used for n`. This setting invalidates the token if it is not used for the amount of time specified. For example, if the field value states `7 days`, and the refresh token is not exchanged for a new session within seven days, the next attempt to use the token fails. The token expired and can no longer generate new sessions. If the refresh token is successfully used before 7 days, monitoring the period of inactivity resets, and the token is valid for another 7 days.
 - `Expire refresh token after n`. This setting invalidates the refresh token after a fixed amount of time. For example, if the policy states `1 day`, the refresh token can be used to obtain new sessions for 24 hours. After 24 hours, the token can't be used.


A user's session can be maintained by usage. Its validity period is defined by the Timeout Value for the connected app, user profile, or organization's session settings (in that order). The `Refresh Token Policy` is evaluated only during usage of the issued refresh token and does not affect a user's current session. Refresh tokens are required only when a user's session has expired or is no longer available. For example, if you set a `Refresh Token Policy` to `Expire refresh token after 1 hour`, and the user uses the application for 2 hours, the user isn't forced to authenticate after 1 hour. The user is required to re-authenticate when the session expires and the client attempts to exchange its refresh tokens for a new session.

- `Timeout Value` is available for OAuth-enabled connected apps, only. This value sets the expiration of the access tokens for the connected app's session. If you don't set a value or `None` is selected (the default), Salesforce uses the Timeout Value in the user's profile. If the profile has no value set, Salesforce uses the Timeout Value in the organization's Session Settings.
- The current permissions for the connected app are also listed here.

If your connected app is a canvas app that uses signed request authentication, be sure to:

- Set `Permitted Users` to `Admin-approved users are pre-authorized`.
 - Set `Expire Refresh Tokens` to `The first time they use this application`.
 - Give users access via profiles and permission sets.
- `Session Level Policy` is available for all connected apps. Select `High Assurance session required` to require users to enter a time-based token during login to access the app.

- Basic Information is available for all connected apps. However, if your app is a canvas app, these field values aren't used. Instead, the canvas app URL that was specified when the connected app was created is used.
 - `Start URL` is used if the connected app uses single sign-on. In this case, set the URL to the page where the user starts the authentication process. This location also appears in the application switcher menu.
 - `Mobile Start URL` is used to direct users to a specific location when the app is accessed from a mobile device.
- Mobile App settings are available for mobile connected apps that enforce pin protection.
 - `Require PIN after` specifies how much time can pass while the app is idle before the app locks itself and requires the PIN before continuing. Allowable values are none (no locking), 1, 5, 10, and 30 minutes. This policy is only enforced if a corresponding `Pin Length` is configured. Enforcement of the policy is the responsibility of the connected app. Apps written using the Salesforce Mobile SDK can enforce this policy, or the app can read the policy from the `UserInfo` service and enforce the policy.

 **Note:** This setting does not invalidate a user's session. When the session expires due to inactivity, this policy only requires that the user enter a PIN to continue using the current session.

 - `Pin Length` sets the length of the identification number sent for authentication confirmation. The length can be from 4 to 8 digits, inclusive.
- Custom Attributes are available for all connected apps. Developers can set custom SAML metadata or custom OAuth attributes for a connected app. Administrators can delete or edit those attributes or add custom attributes. Attributes deleted, edited, or added by administrators override attributes set by developers. For more information, see [Edit, Package, or Delete a Connected App](#) on page 72.

Connected App IP Relaxation and Continuous IP Enforcement

This topic describes how the `Enforce login IP ranges on every request` Session Settings option affects OAuth-enabled connected app IP relaxation settings.

If you relaxed IP restrictions for your OAuth-enabled connected app, and your organization has the `Enforce login IP ranges on every request` option enabled, the access to your connected app can change. This access change applies to client access, including mobile devices, for all OAuth-enabled connected apps. IP relaxation does not apply to SAML-enabled connected apps.

Table 3: Connected App IP Relaxation Settings and Continuous IP Enforcement

IP Relaxation	When Continuous IP Enforcement Is Disabled (Default)	When Continuous IP Enforcement Is Enabled
Enforce IP restrictions	A user running this app is subject to the organization's IP restrictions, such as IP ranges set in the user's profile.	A user running this app is subject to the organization's IP restrictions, such as IP ranges set in the user's profile.
Relax IP restrictions with second factor	A user running this app bypasses the organization's IP restrictions when either of these conditions is true: <ul style="list-style-type: none"> • The app has IP ranges whitelisted and is using the Web server OAuth authentication flow. Only requests 	A user running this app bypasses the organization's IP restrictions when either of the OAuth conditions in the previous column is true. However, the user can't access the following for security reasons: <ul style="list-style-type: none"> • Change password

EDITIONS

Connected Apps can be created in: **Group, Professional, Enterprise, Performance, Unlimited, and Developer** Editions

Connected Apps can be installed in: **All** Editions

IP Relaxation	When Continuous IP Enforcement Is Disabled (Default)	When Continuous IP Enforcement Is Enabled
	<p>coming from the whitelisted IPs are allowed.</p> <ul style="list-style-type: none"> The app has no IP range whitelist, is using the Web server or user-agent OAuth authentication flow, and the user successfully completes identity confirmation. 	<ul style="list-style-type: none"> Add a time-based token Any pages in a login flow
Relax IP restrictions	A user running this connected app is not subject to any IP restrictions.	<p>A user running this connected app is not subject to any IP restrictions. However, the user can't access the following for security reasons:</p> <ul style="list-style-type: none"> Change password Add a time-based token Any pages in a login flow

Monitoring Usage for a Connected App

USER PERMISSIONS

To read:	"Customize Application"
To create, update, or delete:	"Customize Application" AND either "Modify All Data" OR "Manage Connected Apps"
To update all fields except Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application"
To update Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application" AND "Modify All Data"
To uninstall:	"Download AppExchange Packages"

EDITIONS

Connected Apps can be created in: **Group, Professional, Enterprise, Performance, Unlimited,** and **Developer** Editions

Connected Apps can be installed in: **All** Editions

To view information on the usage of any connected apps in the organization, from Setup, click **Manage Apps > Connected Apps OAuth Usage**. A list of connected apps and information about each appears.

Connected App

The name of the app. Connected apps that are installed but haven't been used by anyone don't appear in the list.

View App Info

Click **View App Info** to go to the detail page of the connected app. Alternatively, if the connected app isn't yet installed, click **Install**.

User Count

The number of users who have run the app. Click a User Count value to see information about each user, including:

- When they first used the app
- The most recent time they used the app
- The total number of times they used the app

On the Connected App User's Usage page, you can end a user's access to their current session by clicking the **Revoke** action on that person's row. Or, click the **Revoke All** button at the top of the page to log out everyone currently using the connected app.

Action

Click **Block** to end all current user sessions with the connected app and block all new sessions. Blocking an app is not permanent. You can click **Unblock** to allow users to log in and access the app at another time.

Uninstalling a Connected App

USER PERMISSIONS


To read:	"Customize Application"
To create, update, or delete:	"Customize Application" AND either "Modify All Data" OR "Manage Connected Apps"
To update all fields except Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application"
To update Profiles, Permission Sets, and Service Provider SAML Attributes:	"Customize Application" AND "Modify All Data"
To uninstall:	"Download AppExchange Packages"

EDITIONS

Connected Apps can be created in: **Group, Professional, Enterprise, Performance, Unlimited,** and **Developer** Editions

Connected Apps can be installed in: **All** Editions

You remove a connected app from your organization by uninstalling the package the app is part of.

 **Note:** When a connected app is uninstalled, the access and refresh tokens of all users of the application are removed. This prevents a user from running the application later, using an existing access token, without explicitly approving the application themselves.

Environment Hub

The Environment Hub lets you view, connect, create, and log in to multiple Salesforce organizations from one location. This can be especially convenient if you use a large number of organizations for business, development, and testing.

You must choose one organization as the Environment Hub (or hub organization), and can then connect all your other organizations (or member organizations) to the hub. You can establish single sign-on between the hub and member organizations, enabling users to seamlessly switch between them without having to provide login credentials.

When you connect an organization to the hub, related organizations are automatically discovered so you don't have to manually connect them. The following types of related organizations are auto-discovered.

- For any organization, all sandbox organizations created from it.
- For a release organization, all its related patch organizations.
- For a Trialforce Management Organization, all Trialforce Source Organizations created from it.
- For a License Management Organization (LMO), that is, an organization with the License Management App installed, any release organization (and therefore all its associated patch organizations) that has a managed package registered in the LMO.

To access the Environment Hub:

1. In the Force.com App menu, click **Environment Hub**.
2. Click the **Environment Hub** tab.



Note: To find the hub organization for any member organization, from Setup, click **Company Profile > Company Information**. The organization ID of the hub organization is listed in the Organization Details section.

The Environment Hub main page displays a list of all member organizations connected to the hub. For each organization, its name, description, edition, organization ID, single sign-on status, and other details are displayed. From the Environment Hub tab, you can perform these actions.

- Click **Connect Organization** to add another organization.
- Click **Create Organization** to create a new organization.
- Click the name of any organization to view details about it, including other organizations related to it.
- Click **Edit** next to an organization to update its details.
- Click **Del** next to an organization to delete it from the Environment Hub.
- Click **Login** next to an organization to log in to it. If you've previously enabled single sign-on for that organization, you'll be logged in without providing login credentials.

If you've added a large number of organizations to the Environment Hub, you can zero in on organizations of a specific type, for example, development organizations, or organizations created after a certain date. To create a custom view, click **Create New View** at the top of the page. You can filter organizations based on multiple criteria such as edition, creation date, instance, origin, SSO status, and so on.

Each member organization corresponds to an EnvironmentHubMember object. This is a standard object, similar to Accounts or Contacts. Hence, you can use the full functionality of the Force.com platform to extend or modify the Environment Hub, both via the user interface or the API. For example, you can create custom fields, set up workflow rules, or define user mappings and enable single sign-on using the API, for any member organization.

EDITIONS

Available in:

- Enterprise
- Performance
- Unlimited

USER PERMISSIONS

To set up and configure the Environment Hub:

- "Manage Environment Hub"

To connect an organization to the Environment Hub:

- "Connect Organization to Environment Hub"

Set Up the Environment Hub

Follow these steps to set up the Environment Hub for the first time.

1. Choose which organization you want to use as your hub organization. Use the organization that most employees log in to regularly.
2. Optionally, [set up and deploy My Domain](#) for your hub organization.



Note: You can enable the Environment Hub and use it to create organizations without using My Domain. However, My Domain deployment is required to enable single sign-on or connect existing organizations to the hub.

3. Contact Salesforce support to get the Environment Hub enabled for your hub organization.
4. After the feature is enabled, log in to your hub organization via the new My Domain URL.
5. Edit profiles or permission sets to assign users access to specific features of the Environment Hub.
 - a. From Setup, click **Manage Users > Profiles**.
 - b. Click **Edit** next to the appropriate profile.
 - c. Select the settings you want to enable on the profile editing page.
 - d. Click **Save**.

This table summarizes the settings for the Environment Hub in the profile editing page.

Section of Profile	Settings for Environment Hub
Custom App Settings	Enable the Environment Hub custom app, so it's available in the Force.com App menu.
Connected App Access	The Environment Hub connected app is for internal use only and doesn't need to be enabled for any profiles. Unless advised by Salesforce, don't delete the connected app or adjust its settings.
Service Provider Access	A new entry appears here when single sign-on is enabled in a member organization. Enabling access to a service provider allows single sign-on login access to the corresponding member organization. The service provider is named [OrganizationID] Service Provider, where [OrganizationID] is the organization ID of the member organization. Users who don't have access to the service provider sometimes see this message when attempting to log in via single sign-on: "User [UserID] does not have access to sp [ServiceProviderID]."
Administrative Permissions	The "Manage Environment Hub" permission controls which profiles can enable, create, and edit single sign-on configuration for member organizations. It also controls which profiles can create organizations from the hub (via the Create Organization button).
General User Permissions	The "Connect Organization to Environment Hub" permission controls which profiles can connect existing organizations to Environment Hub.
Standard Object Permissions	The Hub Members settings control access to the Environment Hub Member entities as follows: <ul style="list-style-type: none"> • Read: The ability to view existing Hub Member records.

EDITIONS

Available in:

- Enterprise
- Performance
- Unlimited

USER PERMISSIONS

To set up and configure the Environment Hub:

- "Manage Environment Hub"

Section of Profile	Settings for Environment Hub
	<ul style="list-style-type: none"> • Create: This setting has no impact on the ability to create Hub Member records. That's because record creation is handled either by connecting an existing organization or creating an organization from Environment Hub. • Edit: The ability to edit various fields on existing Hub Member records (Organization, Description, and so on). All fields are editable, because most are managed internally. • Delete: The ability to disconnect an organization from the Hub and delete its corresponding Environment Hub Member record and Service Provider record (if SSO was enabled on the Member). • View All: The administrative ability to read all Hub Member records, regardless of who created them. • Modify All: The administrative ability to read, edit, and delete all Hub Member records, regardless of who created them. <p>Also, the Hub Invitations settings are used to manage connecting organizations to Environment Hub. When enabling the "Connect Organization to Environment Hub" permission, also enable Create, Read, Update, and Delete for the Hub Invitations object. Otherwise, it can safely be disabled.</p>

6. Define user mappings to configure which users have single sign-on access to specific member organizations. For more information, see [Enabling Single Sign-On](#) and [Defining a SSO User Mapping](#).

After you've completed these steps, any user with the appropriate profile can access the Environment Hub by clicking **Environment Hub** in the App menu. The types of actions a user can perform in the Environment Hub depend on the settings in that user's profile.

Setting up My Domain for the Environment Hub

You can create new organizations in Environment Hub without using My Domain. However, if you want to enable single sign-on or connect existing organizations to Environment Hub, you need to set up and deploy My Domain first.

- Find a domain name that's available and sign up for it.
 - From Setup, click **Domain Management > My Domain**.
 - Enter the subdomain name you want to use within the sample URL. You can use up to 40 characters.
 - Click **Check Availability**. If your name is already taken, choose a different one.
 - Click **Terms and Conditions** to review your agreement, then select the checkbox.
 - Click **Register Domain**.

You'll receive an email when your domain name is ready for testing. It can take from 10 minutes to 24 hours.

- Test your domain name and deploy it to your entire organization.
 - From Setup, click **Domain Management > My Domain**, then click **Click here to login**, or click the URL in the confirmation email, to login to Salesforce using your new domain name.
 - Test the new domain name by clicking tabs and links within your application. You'll notice that all pages show your new domain name. If you've customized your Salesforce UI with features such as custom buttons or Visualforce pages, make sure you test custom elements thoroughly before deploying your domain name. Your customizations should not use instance-based URLs.

USER PERMISSIONS

To set up a domain name:

- "Customize Application"

- c. To roll out the new domain name to your organization, from Setup, click **Domain Management > My Domain** and then click **Deploy to Users**.

The domain is activated immediately and all users are redirected to pages with new domain addresses.

3. Set the domain login policy for users accessing your pages.
 - a. From Setup, click **Domain Management > My Domain**.
 - b. Under My Domain Settings, click **Edit**.
 - c. To turn off authentication for users who do not use your domain-specific login page, select the login policy. For example, this will prevent users from logging in at the generic `https://<instance>.salesforce.com/` login page, and being redirected to your pages after login. This option enhances security by preventing login attempts by anyone who does not know your domain name.
 - d. Choose a redirect policy, based on the level of security desired. You have three options, in order of increasing security:
 - Redirect users to the same page within the domain.
 - Redirect users with a warning.
 - Prevent redirecting, so users have to enter the new domain name.
4. Optionally, customize your login page and add or change identity providers available on your login page. For details, see the Salesforce online help.

Environment Hub Best Practices

These guidelines can help you use the Environment Hub effectively.

- The Environment Hub connected app is for internal use only. You don't need to enable it for any profiles. Unless advised by Salesforce, don't delete the connected app or adjust its settings.
- Choose the organization that most employees log in to regularly as your hub organization.
- Set up My Domain for each member organization, in addition to the hub organization.
- Edit profiles or permission sets to assign users access to specific features of the Environment Hub. Some key settings are listed below.
 - The Environment Hub custom app must be enabled for it to appear in the Force.com App menu.
 - “Manage Environment Hub” is required to create organizations or configure single sign-on (SSO) access. Assign this permission only to admin users, because it is powerful functionality.
 - “Connect Organization to Environment Hub” is required to add member organizations to the hub.
 - Service Provider Access must be enabled for each member organization to which you want to allow SSO access.

For details of all the profile settings, see: [Set Up the Environment Hub](#).

- Decide on a strategy for enabling SSO access based on your organization's security requirements. You can then choose which combination of the three SSO methods (explicit mapping, Federation ID, or custom formula) to use for your specific needs. For more information, see [Enabling Single Sign-On](#).
- Because each member organization is a standard object (of type EnvironmentHubMember), you can modify its behavior or access it programmatically. For example, you can create custom fields, set up workflow rules, or define user mappings and enable single sign-on using the API, for any member organization.
- SSO doesn't work for newly added users or for SSO user mappings defined in a sandbox organization. Only add users, edit user information, or define SSO user mappings in the parent organization for the sandbox.

EDITIONS

Available in:

- Enterprise
- Performance
- Unlimited

Connecting an Organization to the Environment Hub

To connect an organization to the Hub:

1. On the Environment Hub main page, click **Connect Organization**.
2. Enter the admin username for the member organization, that is, the organization you want to connect to the hub.
3. Optionally, enter a description for the member organization. Adding a description makes it easier for you to find a particular organization later on, especially if your hub has many members.
4. By default, single sign-on (SSO) is enabled for new member organizations. To disable SSO, deselect `Auto-enable SSO for the newly connected org.`
5. Click **Connect Organization**.
6. Enter the admin username and password of the member organization in the popup window.
7. Click **Log in to Salesforce**.
8. Click **Allow** in the next popup window.

The organization is now connected to the hub and appears in the list of member organizations in the Environment Hub.

EDITIONS

Available in:

- Enterprise
- Performance
- Unlimited

USER PERMISSIONS

To connect an organization to the Environment Hub:

- "Connect Organization to Environment Hub"

Viewing an Environment Hub Member's Details

Click the name of any organization in the Environment Hub to view details about it, such as its name, description, organization ID, related organizations, and linked users.

The detail page has the following sections.

- [Hub Member Detail](#)
- [Parent Organizations](#)
- [Child Organizations](#)
- [Single Sign-On User Mappings](#)

From the Hub Member Detail page, you can do any of the following.

- Click **Edit** to edit information about this organization.
- Click **Delete** to remove this organization from the Environment Hub.
- Click **Enable SSO** to enable single sign-on for this organization. This lets linked users log in to this organization from the Environment Hub without providing credentials.
- Click **Log into Organization** to log in. If you've enabled SSO for this organization, you'll be logged in without entering login credentials.
- Click **New SSO User Mapping** to link a username in the member organization with a username in the hub organization for single sign-on access.

EDITIONS

Available in:

- Enterprise
- Performance
- Unlimited

Hub Member Detail

This section displays the following attributes (in alphabetical order).

Attribute	Description
Description	A brief description of this organization.

Attribute	Description
Edition	The organization's edition, for example, DE or EE.
Origin	The method by which this organization was added to the Environment Hub. Possible values are <code>Auto Discovered</code> and <code>User Added</code> .
Organization	The name of this organization.
Organization ID	The organization ID of this organization.
Sandbox	If this is a sandbox organization.
Service Provider	A Details link here indicates that the organization has been activated for Environment Hub single sign-on by salesforce.com support. You need to enable service provider access for this organization in all profiles that will be using it for single sign-on.
Status	The licensing or creation status of this organization. Possible values include <code>Trial</code> , <code>Free</code> , <code>Active</code> , <code>Suspended</code> , and <code>Deleted</code> .
SSO	If single sign-on has been enabled for this organization. Possible values are: <ul style="list-style-type: none"> • <code>On</code>—Single sign-on is enabled. • <code>Off</code>—Single sign-on is disabled. • <code>Pending</code>—Single sign-on is in the process of being enabled. • <code>Failed</code>—Single sign-on enablement failed. Contact salesforce.com support for assistance.
SSO Method 1 — Mapped Users	The number of SSO user mappings defined between this organization and the environment hub.
SSO Method 2 — Federation ID	If single sign-on has been enabled based on matching Federation ID.
SSO Method 3 — User Name Formula	If single sign-on has been enabled using a custom formula to match usernames.



Note: All fields might not be visible by default. This is because the default layout is customer data, so isn't updated automatically. You might need to edit your layout to add in the new fields, by clicking **Edit Layout** at the top of the page.

Parent Organizations and Child Organizations

When you connect an organization to the Environment Hub, the following types of related organizations are automatically discovered and connected.

- For any organization, all sandbox organizations created from it.
- For a release organization, all its related patch organizations.
- For a Trialforce Management Organization, all Trialforce Source Organizations created from it.

- For a License Management Organization (LMO), that is, an organization with the License Management App installed, any release organization (and therefore all its associated patch organizations) that has a managed package registered in the LMO.

These sections display a list of the organizations related to this organization. For each organization in the list, its name and relationship to this organization are displayed.

Single Sign-On User Mappings

This section displays a list of mapped users, that is, users in the hub organization who are associated with a corresponding user in the member organization. If single sign-on has been enabled for the member organization, all mapped users can log in from the Environment Hub without needing to provide credentials.

From this section, you can do any of the following:

- Click **New SSO User Mapping** to define a new user mapping.
- Click **Del** next to a user to disable single sign-on access.

User mappings can be many-to-one (but not one-to-many). This means you can associate multiple users in the hub organization to the same user in a member organization. This can be useful, for example, if you want a group of users to log in to a test organization as the same user, so they experience exactly the same configuration.

Note:

- If a user mapping has been defined and the user is still unable to use single sign-on, check that the user's profile has the appropriate permissions enabled. For details, see [Setting up Environment Hub](#).
- SSO doesn't work for newly added users or for SSO user mappings defined in a sandbox organization. Only add users, edit user information, or define SSO user mappings in the parent organization for the sandbox.

Editing an Environment Hub Member's Details

To edit details of a member organization in the Environment Hub:

1. Click **Edit** next to the organization's name in the Environment Hub main page, or on its detail page.
2. In the page that displays, edit the name and description for the organization. It's helpful to specify a meaningful name and description. This lets you easily recognize the organization in the list of members in the Environment Hub.
3. Optionally, specify one or more single sign-on methods (see below for details).
4. Click **Save**.

To match users in the member and hub organizations for single sign-on, you can use any of three methods.

EDITIONS

Available in:

- Enterprise
- Performance
- Unlimited

SSO Method	Description
Mapped Users	Match users in the hub organization to users in a member organization manually. This method is on by default if you've defined any SSO user mappings from the member detail page. For details, see Defining a SSO User Mapping .
Federation ID	Match users who have the same Federation ID in both organizations. To enable this method, select the checkbox next to it.
User Name Formula	Define a custom formula for matching users in the hub and member organizations. This allows you the most flexibility. To enable this method, type a custom formula into the

SSO Method	Description
	<p>text box provided. For example, the following formula matches the first part of the user name (the part before the "@" sign) with an explicit domain name.</p> <pre>LEFT(\$User.Username, FIND("@", \$User.Username)) & ("mydev.org")</pre>

If you specify more than one single sign-on method, they're evaluated in the order of precedence listed above at the time a user tries to log in. The first method that results in a match is used to log the user in, and the other methods are ignored. If no matching user can be identified, you're directed to the standard salesforce.com login page.

Creating a New Organization from the Environment Hub

You can create new organizations directly from the Environment Hub, for the following purposes.

- **Development** — Development organizations for creating and uploading managed packages.
- **Test/Demo** — Trial organizations created for test and demonstration purposes, similar to organizations created from www.salesforce.com/trial. You can specify a Salesforce template for creating trial organizations. This enables you to configure and test exactly how your prospects experience the trial.
- **Salesforce** — You can create Salesforce Source Organizations (TSOs) from the Environment Hub, as an alternative to using a Salesforce Management Organization (TMO).



Note: Custom branding is not supported in TSOs created using the Environment Hub. If you plan to brand your emails or login page, create the TSO from a TMO instead.

To create a new organization in the Environment Hub:

1. On the Environment Hub, click **Create Organization**.
2. On the page that appears, from the pulldown menu, choose the type of organization you want to create. Possible choices are: **Development**, **Test/Demo**, and **Salesforce**.
3. On the page that appears, specify these details.
 - organization's name and My Domain
 - name, user name, and email address of the admin user
 - edition (for development and Salesforce organizations)
 - edition or a Salesforce template ID (for test/demo organizations)
4. Read the Master Subscription Agreement and select the checkbox.
5. Click **Create**.

Once the organization is created, it'll appear in the Environment Hub and you'll get an email confirmation.

EDITIONS

Available in:


- Enterprise
- Performance
- Unlimited

USER PERMISSIONS

To set up and configure the Environment Hub:

- "Manage Environment Hub"

Enabling Single Sign-On in the Environment Hub

 **Note:** You can enable the Environment Hub and use it to create organizations without using My Domain. However, My Domain deployment is required to enable single sign-on or connect existing organizations to the hub.

To enable single sign-on for an organization:

1. On the Environment Hub main page, click the name of the organization.
2. Click **Enable SSO** on the organization's detail page.
3. On the page that displays, click **Enable SSO**.

You're taken to the detail page of the organization. Its single sign-on status displays both at the top of the page, and next to the SSO field in the Hub Member Details section.

To match users in the member and hub organizations for single sign-on, you can use any of three methods.

SSO Method	Description
Mapped Users	Match users in the hub organization to users in a member organization manually. This method is on by default if you've defined any SSO user mappings from the member detail page. For details, see Defining a SSO User Mapping .
Federation ID	Match users who have the same Federation ID in both organizations. To enable this method, select the checkbox next to it.
User Name Formula	Define a custom formula for matching users in the hub and member organizations. This allows you the most flexibility. To enable this method, type a custom formula into the text box provided. For example, the following formula matches the first part of the user name (the part before the "@" sign) with an explicit domain name. <code>LEFT(\$User.Username, FIND("@", \$User.Username)) & ("mydev.org")</code>

If you specify more than one single sign-on method, they're evaluated in the order of precedence listed above at the time a user tries to log in. The first method that results in a match is used to log the user in, and the other methods are ignored. If no matching user can be identified, you're directed to the standard salesforce.com login page.

 **Note:**

- If a user mapping has been defined and the user is still unable to use single sign-on, check that the user's profile has the appropriate permissions enabled. For details, see: [Setting up Environment Hub](#).
- SSO doesn't work for newly added users or for SSO user mappings defined in a sandbox organization. Only add users, edit user information, or define SSO user mappings in the parent organization for the sandbox.

EDITIONS

Available in:

- Enterprise
- Performance
- Unlimited

USER PERMISSIONS

To set up and configure the Environment Hub:

- "Manage Environment Hub"

Disabling Single Sign-On in the Environment Hub

To disable single sign-on for an organization:

1. On the Environment Hub main page, click the name of the organization.
2. Click **Disable SSO** on the organization's detail page.
3. On the page that displays, click **Disable SSO**.

You're taken to the detail page of the organization. Its single sign-on status displays both at the top of the page, and next to the SSO field in the Hub Member Details section.

EDITIONS

Available in:

- Enterprise
- Performance
- Unlimited

USER PERMISSIONS

To set up and configure the Environment Hub:

- "Manage Environment Hub"

Mapping Users for Single Sign-On in the Environment Hub

You can define a mapping between a user in the hub organization and one or more users in a member organization. If single sign-on has been enabled for the member organization, all mapped users can log in to it from the Environment Hub without needing to provide credentials.

User mappings can be many-to-one (but not one-to-many). This means you can associate multiple users in the hub organization to the same user in a member organization. This can be useful, for example, if you want a group of users to log in to a test organization as the same user, so they experience exactly the same configuration.

To define a new single sign-on user mapping in Environment Hub:

1. On the Environment Hub main page, click the name of the organization.
2. Click **New SSO User Mapping** on the hub member detail page.
3. In the page that appears, enter the username for the member organization and specify the corresponding user for the hub organization using the lookup field.
4. Click **Save** (or **Save & New** to save and add a new mapped user).



Note:

- If a user mapping has been defined and the user is still unable to use single sign-on, check that the user's profile has the appropriate permissions enabled. For details, see [Setting up Environment Hub](#).
- SSO doesn't work for newly added users or for SSO user mappings defined in a sandbox organization. Only add users, edit user information, or define SSO user mappings in the parent organization for the sandbox.

EDITIONS

Available in:

- Enterprise
- Performance
- Unlimited

USER PERMISSIONS

To set up and configure the Environment Hub:

- "Manage Environment Hub"

CHAPTER 4 Packaging and Testing Your App

In this chapter ...

- [About Managed Packages](#)
- [Installing a Package](#)
- [Uninstalling a Package](#)
- [Installing Managed Packages using the API](#)
- [Resolving Apex Test Failures](#)
- [Running Apex on Package Install/Upgrade](#)
- [Running Apex on Package Uninstall](#)
- [Publishing Extensions to Managed Packages](#)

This section contains information on packaging and testing your app during development. The general procedure is as follows:




1. Create and upload a beta package.
2. Install the beta package in a partner testing organization (Enterprise, Professional or Group Editions are available). These can be created in the Environment Hub.
3. Test the package.
4. Fix bugs and make changes in your development organization.
5. Repeat these steps until you're ready to release a managed package.


SEE ALSO:

[Creating and Uploading a Beta Package](#)
[Installing a Package](#)

About Managed Packages

A managed package is a collection of application components that are posted as a unit on AppExchange, and are associated with a namespace and a License Management Organization.

- You must use a Developer Edition organization to create and work with a managed package.
- Managed packages are depicted by the following icons:
 -  Managed - Beta
 -  Managed - Released
 -  Managed - Installed


 **Tip:** To prevent naming conflicts, Salesforce recommends using managed packages for all packages that contain Apex. This way, all of the Apex objects contain your [namespace prefix](#). For example, if there is an Apex class called `MyHelloWorld` and the namespace for your organization is `OneTruCode`, the class is referenced as `OneTruCode.MyHelloWorld`.

Configuring Your Developer Settings

The developer settings in a Developer Edition organization allow you to create a single managed package, upload that package to the AppExchange, allowing other users to install and upgrade the package in their organization. After configuring your developer settings the first time, you can no longer modify them. Regardless of the developer settings, you can always create an unlimited number of unmanaged packages.

To configure your developer settings:

1. From Setup, click **Create > Packages**.
2. Click **Edit**.

 **Note:** This button doesn't appear if you've already configured your developer settings.

3. Review the selections necessary to configure developer settings, and click **Continue**.
4. [Register a namespace prefix](#).
5. Choose the package you want to convert to a managed package. If you do not yet have a package to convert, leave this selection blank and update it later.
6. Click **Review My Selections**.
7. Click **Save**.

 **Tip:** You may want to [specify a License Management Organization \(LMO\)](#) for your managed package; to find out more, go to <http://sites.force.com/appexchange/publisherHome>.

EDITIONS

Available in: **Developer Edition**

Package uploads and installs are available in **Group, Professional, Enterprise, Performance, Unlimited, and Developer Editions**

EDITIONS

Available in: **Developer Edition**

Package uploads and installs are available in **Group, Professional, Enterprise, Performance, Unlimited, and Developer Editions**

USER PERMISSIONS

To configure developer settings:

- "Customize Application"

To create packages:

- "Create AppExchange Packages"

To upload packages:

- "Upload AppExchange Packages"

Registering a Namespace Prefix

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 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 example, if your namespace prefix is abc and your managed package contains a custom object with the API name, Expense__c, use the API name abc__Expense__c to access this object using the API. The namespace prefix is displayed on all component detail pages.

EDITIONS

Available in: **Developer** Edition

Package uploads and installs are available in **Group, Professional, Enterprise, Performance, Unlimited, and Developer** Editions



Warning: S-controls stored in the s-control library or the Documents tab that do not use the Force.com API still function properly after you register a namespace prefix. However, s-controls stored outside of your organization or s-controls that use the Force.com API to call Salesforce may require some fine-tuning. For more information, see [S-control](#) in the *Object Reference*.

Your namespace prefix must:

- Begin with a letter
- Contain one to 15 alphanumeric characters
- Not contain two consecutive underscores

To register a namespace prefix:

1. From Setup, click **Create > Packages**.
2. Click **Edit**.



Note: This button doesn't appear if you've already configured your developer settings.

3. Review the selections that are required for configuring developer settings, and then click **Continue**.
4. Enter the namespace prefix you want to register.
5. Click **Check Availability** to determine if the namespace prefix is already in use.
6. If the namespace prefix that you entered isn't available, repeat the previous two steps.
7. Click **Review My Selections**.
8. Click **Save**.

Specifying a License Management Organization

A license management organization is a Salesforce organization that you use to track all Salesforce users who install your managed package. The license management organization receives notification (in the form of a lead record) when a user installs or uninstalls your package and tracks each package upload on Force.com AppExchange.

Your license management organization can be any Salesforce Enterprise, Unlimited, Performance, or Developer Edition organization that has installed the free License Management Application (LMA) from AppExchange. To specify a License Management Organization, go to <http://sites.force.com/appexchange/publisherHome>.

EDITIONS

Available in: **Developer Edition**

Package uploads and installs are available in **Group, Professional, Enterprise, Performance, Unlimited, and Developer Editions**

What are Beta Versions of Managed Packages?

A beta package is an early version of a managed package that is uploaded in a Managed - Beta state. The purpose of a Managed - Beta package is to allow the developer to test their application in different Salesforce organizations and to share the app with a pilot set of users for evaluation and feedback.

Before installing a beta version of a managed package, review the following notes:

- Beta packages can be installed in sandbox or Developer Edition organizations, or test organizations furnished through the Environment Hub only.
- The components of a beta package are editable by the developer's organization until a Managed - Released package is uploaded.
- Beta versions aren't considered major releases, so the package version number doesn't change.
- Beta packages are not upgradeable. Because developers can still edit the components of a beta package, the Managed - Released version might not be compatible with the beta package installed. Uninstall the beta package and install a new beta package or released version. For more information, see [Uninstalling a Package](#) on page 105 and [Installing a Package](#) on page 103.

Creating and Uploading a Beta Package

Use the following procedure to create and upload a beta package.

1. Create a package:
 - a. From Setup, click **Create > Packages**.
 - b. Click **New**.
 - c. Enter a name for your package. This does not have to be the same name that appears on AppExchange.
 - d. From the drop-down menu, select the default language of all component labels in the package.
 - e. Optionally, in the `Notify on Apex Error` field, enter the username of the person who should receive an email notification if an exception occurs in Apex code that is not caught by the code. If you don't specify a username, all uncaught exceptions generate an email notification that is sent to Salesforce.
 - f. Optionally, choose a custom link from the `Configure Custom Link` field to display configuration information to installers. The custom link displays as a **Configure** link within Salesforce on the Installed Packages page and package detail page of the subscriber's organization.
 - g. Optionally, enter a description that describes the package. You will have a chance to change this description before you upload it to AppExchange.

USER PERMISSIONS


To create packages:


- "Create AppExchange Packages"


To upload packages:

- "Upload AppExchange Packages"

- h. Optionally, specify a post install script. This is an Apex script that runs in the subscriber organization after the package is installed or upgraded. For more information, see [Running Apex on Package Install/Upgrade](#) on page 108.
 - i. Optionally, specify an uninstall script. This is an Apex script that runs in the subscriber organization after the package is uninstalled. For more information, see [Running Apex on Package Uninstall](#) on page 111.
 - j. On the right side of the screen, select the **Managed** checkbox.
 - k. Click **Save**.
2. Optionally, change the API access privileges. By default, API access is set to **Unrestricted**, but you can change this setting to further restrict API access of the components in the package.
 3. Add the necessary components for your app.
 - a. Click **Add Components**.
 - b. From the drop-down list, choose the type of component.
 - c. Select the components you want to add.

 **Note:** Some components cannot be added to Managed - Released packages. For a list of packageable components, see [Components Available in Managed Packages](#) on page 21. Also, S-controls cannot be added to packages with restricted API access.
 - d. Click **Add To Package**.
 - e. Repeat these steps until you have added all the components you want in your package.

 **Note:** Some related components are automatically included in the package even though they might not display in the Package Components list. For example, when you add a custom object to a package, its custom fields, page layouts, and relationships with standard objects are automatically included. For a complete list of components, see [Components Automatically Added to Packages](#) on page 34.
 4. Optionally, click **View Dependencies** and review a list of components that rely on other components, permissions, or preferences within the package. For more information on dependencies, see [About Dependencies](#) on page 46. Click **Done** to return to the Package detail page.
 5. Click **Upload**.
 6. On the Upload Package page, do the following:
 - a. Enter a **Version Name**, such as *Spring 11 - Beta*.
 - b. Enter a **Version Number**, such as *1.0*. All beta packages use the same version number until you upload a Managed - Released package.
 - c. Select a **Release Type** of Managed - Beta.

 **Note:** Beta packages can only be installed in Developer Edition, sandbox, or test organizations requested through the Environment Hub, and thus can't be pushed to customer organizations.
 - d. Optionally, enter and confirm a password to share the package privately with anyone who has the password. Don't enter a password if you want to make the package available to anyone on AppExchange and share your package publicly.
 - e. Salesforce automatically selects the requirements it finds. In addition, select any other required components from the **Package Requirements** and **Object Requirements** sections to notify installers of any requirements for this package.
 - f. Click **Upload**.

You will receive an email that includes an installation link when your package has been uploaded successfully.

Creating and Uploading a Managed Package

Creating a managed package is just as easy as creating an unmanaged package. The only requirement to create a managed package is that you're using a Developer Edition organization.

Before creating a managed package:

- Determine if you want to create and upload a managed or unmanaged package.
- Optionally, install the License Management Application (LMA) from <http://sites.force.com/appexchange>. Search for *License Management App* to locate it. The License Management Application (LMA) tracks information about each user who installs your app. It allows you to track what users have which version, giving you a means of distributing information about upgrades.

The License Management Application (LMA) can be installed in any Salesforce organization except a Personal, Group, or Professional Edition organization and does not need to be the same Salesforce organization that you use to create or upload the package, although it can be. You can also use the same License Management Application (LMA) to manage an unlimited number of your managed packages in different Developer Edition organizations.

- [Configure your developer settings](#) on page 94. Your developer settings specify your [namespace prefix](#) on page 95, the Salesforce organization where you install the License Management Application (LMA), and the unmanaged package you want to convert into a managed package.

Use the following procedure to create and upload a managed package. The procedure assumes you have already created a namespace and beta package. If you're uploading a beta package for testing, see [Creating and Uploading a Beta Package](#) on page 96.

1. Create a package:
 - a. From Setup, click **Create > Packages**.
 - b. Click **New**.
 - c. Enter a name for your package. This does not have to be the same name that appears on AppExchange.
 - d. From the drop-down menu, select the default language of all component labels in the package.
 - e. Optionally, in the `Notify on Apex Error` field, enter the username of the person who should receive an email notification if an exception occurs in Apex code that is not caught by the code. If you don't specify a username, all uncaught exceptions generate an email notification that is sent to Salesforce.
 - f. Optionally, choose a custom link from the `Configure Custom Link` field to display configuration information to installers. The custom link displays as a **Configure** link within Salesforce on the Installed Packages page and package detail page of the subscriber's organization.
 - g. Optionally, enter a description that describes the package. You will have a chance to change this description before you upload it to AppExchange.
 - h. Optionally, specify a post install script. This is an Apex script that runs in the subscriber organization after the package is installed or upgraded. For more information, see [Running Apex on Package Install/Upgrade](#) on page 108.
 - i. Optionally, specify an uninstall script. This is an Apex script that runs in the subscriber organization after the package is uninstalled. For more information, see [Running Apex on Package Uninstall](#) on page 111.
 - j. On the right side of the screen, select the **Managed** checkbox.
 - k. Click **Save**.

EDITIONS

Available in: **Developer Edition**

Package uploads and installs are available in **Group, Professional, Enterprise, Performance, Unlimited, and Developer Editions**

USER PERMISSIONS

To enable managed packages:

- "Customize Application"


To create packages:


- "Create AppExchange packages"


To upload packages:


- "Download AppExchange packages"

2. Optionally, change the API access privileges. By default, API access is set to `Unrestricted`, but you can change this setting to further restrict API access of the components in the package.
3. Add the necessary components for your app.
 - a. Click **Add Components**.
 - b. From the drop-down list, choose the type of component.
 - c. Select the components you want to add.

 **Note:** Some components cannot be added to Managed - Released packages. For a list of packageable components, see [Components Available in Managed Packages](#) on page 21. Also, S-controls cannot be added to packages with restricted API access.
 - d. Click **Add To Package**.
 - e. Repeat these steps until you have added all the components you want in your package.

 **Note:** Some related components are automatically included in the package even though they might not display in the Package Components list. For example, when you add a custom object to a package, its custom fields, page layouts, and relationships with standard objects are automatically included. For a complete list of components, see [Components Automatically Added to Packages](#) on page 34.
4. Optionally, click **View Dependencies** and review a list of components that rely on other components, permissions, or preferences within the package. For more information on dependencies, see [About Dependencies](#) on page 46. Click **Done** to return to the Package detail page.
5. Click **Upload**.
6. On the Upload Package page, do the following:
 - a. Enter a `Version Name`, such as *Spring 12*. The version name is the marketing name for a specific release of a package and allows you to create a more descriptive title for the version than just a number.
 - b. Enter a `Version Number`, such as *1.0*. For more information on versions, see [Upgrading Your App](#) on page 215.
 - c. Select a `Release Type` of Managed - Released.
 - d. Change the `Description`, if necessary.
 - e. Optionally, specify a link to release notes for the package. Click **URL** and enter the details in the text field that appears. This link will be displayed during the installation process, and on the Package Details page after installation.

 **Note:** As a best practice, this should point to an external URL, so you can make the information available to customers in advance of the release, and update it independently of the package.
 - f. Optionally, specify a link to post install instructions for the package. Click **URL** or **Visualforce page** and enter the details in the text field that appears. This link will be displayed on the Package Details page after installation.

 **Note:** As a best practice, this should point to an external URL, so you can update the information independently of the package.
 - g. Optionally, enter and confirm a password to share the package privately with anyone who has the password. Don't enter a password if you want to make the package available to anyone on AppExchange and share your package publicly.
 - h. Salesforce automatically selects the requirements it finds. In addition, select any other required components from the `Package Requirements` and `Object Requirements` sections to notify installers of any requirements for this package.
 - i. Click **Upload**.
7. Once your upload is complete, you can do any of the following.

- Click **Change Password** link to change the password option.
- Click **Deprecate** to prevent new installations of this package while allowing existing installations to continue operating.

 **Note:** You cannot deprecate the most recent version of a managed package.

When you deprecate a package, remember to remove it from AppExchange as well. See “Removing Apps from AppExchange” in the AppExchange online help.

- Click **Undeprecate** to make a deprecated version available for installation again.

You will receive an email that includes an installation link when your package has been uploaded successfully.

 **Note:**

- When using the install URL, the old installer is displayed by default. You can customize the installation behavior by modifying the installation URL you provide your customers.
 - To access the new installer, append the text `&newui=1` to the installation URL.
 - To access the new installer with the "All Users" option selected by default, append the additional text `&p1=full` to the installation URL.
- If you uploaded from your Salesforce production organization, notify installers who want to install it in a sandbox organization to replace the “login.salesforce.com” portion of the installation URL with “test.salesforce.com.”

Viewing Package Details

From Setup, click **Create > Packages** and click the name of a package to view its details, including any added components, whether it is a managed package, if the package has been uploaded, and so on.

The detail page has the following sections:

- [Package Details](#) on page 100
- [Components](#) on page 101
- [Versions](#) on page 102
- [Patch Organizations](#) on page 102


From the Package Detail page, you can do any of the following:

- Click **Edit** to change the package name, custom link that displays when users click Configure, or description.
- Click **Delete** to delete the package. This does not delete the components contained in the package but the components will no longer be bundled together within this package.
- Click **Upload** to upload the package. You will receive an email when the upload is complete.
- Optionally, you can enable, disable, or change the dynamic Apex and API access that components in the package have to standard objects in the installing organization by using the links next to **API Access**.

Viewing Package Details

For package developers, the package detail section displays the following package attributes (in alphabetical order):


Attribute	Description
API Access	The type of access that the API and dynamic Apex that package components have. The default setting is Unrestricted , which

Attribute	Description
	means that all package components that access the API have the same access as the user who is logged in. Click Enable Restrictions or Disable Restrictions to change the API and dynamic Apex access permissions for a package.
Created By	The name of the developer that created this package, including the date and time.
Description	A detailed description of the package.
Language	The language used for the labels on components. The default value is your user language.
Last Modified By	The name of the last user to modify this package, including the date and time.
Notify on Apex Error	<p>The username of the person who should receive an email notification if an exception occurs in Apex that is not caught by the code. If you don't specify a username, all uncaught exceptions generate an email notification that is sent to Salesforce. This is only available for managed packages.</p> <p> Note: Apex can only be packaged from Developer, Enterprise, Unlimited, and Performance Edition organizations.</p>
Package Name	The name of the package, given by the publisher.
Post Install Script	The Apex code that runs after this package is installed or upgraded. For more information, see Running Apex on Package Install/Upgrade on page 108.
Type	Indicates whether this is a managed or unmanaged package.
Uninstall Script	The Apex code that runs after this package is uninstalled. For more information, see Running Apex on Package Uninstall on page 111.

Viewing Package Components

For package developers, the Components tab lists every package component contained in the package, including the name and type of each component.

Click **Add** to add components to the package.

 **Note:** Some related components are automatically included in the package even though they may not display in the Package Components list. For example, when you add a custom object to a package, its custom fields, page layouts, and relationships with standard objects are automatically included. For a complete list of components Salesforce automatically includes, see [Components Automatically Added](#) on page 34.

Click **View Dependencies** to review a list of components that rely on other components, permissions, or preferences within the package. An entity may include such things as an s-control, a standard or custom field, or an organization-wide setting like multicurrency. Your

package cannot be installed unless the installer has the listed components enabled or installed. For more information on dependencies, see [Understanding Dependencies](#) on page 46. Click **Back to Package** to return to the Package detail page.

Click **View Deleted Components** to see which components were deleted from the package across all of its versions.

Viewing Version History

For package developers, the Versions tab lists all the previous uploads of a package.


Click **Push Upgrades** to [automatically upgrade subscribers to a specific version](#).

Click the Version Number of any listed uploads to manage that upload. For more information, see [Managing Versions](#) on page 226.



Note: **Push Upgrades** is available for patches and major upgrades. Registered ISV partners can request Push Major Upgrade functionality by logging a case in the [Partner Community](#).

The versions table displays the following package attributes (in alphabetical order):

Attribute	Description
Action	<p>Lists the actions you can perform on the package. The possible actions are:</p> <ul style="list-style-type: none"> • Deprecate: Deprecates a package version. <p> Warning: Users will no longer be able to download or install this package. However, existing installations will continue to work.</p> <ul style="list-style-type: none"> • Undeprecate: Enables a package version to be installed by users once again.
Status	<p>The status of the package. The possible statuses are:</p> <ul style="list-style-type: none"> • Released: The package is Managed - Released. • Beta: The package is Managed - Beta. • Deprecated: The package version is deprecated.
Version Name	<p>The version name for this package version. The version name is the marketing name for a specific release of a package. It is more descriptive than the <code>Version Number</code>.</p>
Version Number	<p>The version number for the latest installed package version. The format is <code>majorNumber.minorNumber.patchNumber</code>, such as 2.1.3. The version number represents a release of a package. The <code>Version Name</code> is a more descriptive name for the release. The <code>patchNumber</code> is generated only when you create a patch. If there is no <code>patchNumber</code>, it is assumed to be zero (0).</p>

Viewing Patch Development Organizations

Every patch is developed in a *patch development organization*, which is the organization where patch versions are developed, maintained, and uploaded. To start developing a patch, you need to create a patch development organization. To do this, see [Creating and Uploading](#)

[Patches](#) on page 217. Patch development organizations are necessary to permit developers to make changes to existing components without causing incompatibilities between existing subscriber installations. Click **New** to create a new patch for this package.

The Patch Organizations table lists all the patch development organizations created. It lists the following attributes (in alphabetical order):

Attribute	Description
Action	Lists the actions you can perform on a patch development organization. The possible actions are: <ul style="list-style-type: none"> • Login: Log in to your patch development organization. • Reset: Emails a new temporary password for your patch development organization.
Administrator Username	The login associated with the patch organization.
Patching Major Release	The package version number that you are patching.

Installing a Package

During the development and testing cycle, you might need to periodically install and uninstall packages before you install the next beta. Follow these steps to install a package.

Pre-Installation


1. In a browser, type in the installation URL you received when you uploaded the package.
2. Enter your username and password for the Salesforce organization in which you want to install the package, and then click the login button.
3. If the package is password-protected, enter the password you received from the publisher.

Default Installation

Click **Install**. You'll see a message that describes the progress and a confirmation message after the installation is complete.

Custom Installation

Follow these steps if you need to modify the default settings, as an administrator.

1. Choose one or more of these options, as appropriate.
 - Click **View Components**. You'll see an overlay with a list of components in the package. For managed packages, the screen also contains a list of connected apps (trusted applications that are granted access to a user's Salesforce data after the user and the application are verified). Review the list to confirm that the components and any connected apps shown are acceptable, and then close the overlay.
-  **Note:** Some package items, such as validation rules, record types, or custom settings might not appear in the Package Components list but are included in the package and installed with the other items. If there are no items in the Package Components list, the package might contain only minor changes.

- If the package contains a remote site setting, you must approve access to websites outside of Salesforce. The dialog box lists all the websites that the package communicates with. We recommend that a website uses SSL (secure sockets layer) for transmitting data. After you verify that the websites are safe, select **Yes, grant access to these third-party websites** and click **Continue**, or click **Cancel** to cancel the installation of the package.



Warning: By installing remote site settings, you're allowing the package to transmit data to and from a third-party website. Before using the package, contact the publisher to understand what data is transmitted and how it's used. If you have an internal security contact, ask the contact to review the application so that you understand its impact before use.

- Click **API Access**. You'll see an overlay with a list of the API access settings that package components have been granted. Review the settings to verify they're acceptable, and then close the overlay to return to the installer screen.
- In Enterprise, Performance, Unlimited, and Developer Editions, choose one of the following security options.



Note: Depending on the type of installation, you might not see this option. For example, in Group and Professional Editions, or if the package doesn't contain a custom object, Salesforce skips this option, which gives all users full access.

Install for Admins Only

Specifies the following settings on the installing administrator's profile and any profile with the "Customize Application" permission.

- Object permissions—"Read," "Create," "Edit," "Delete," "View All," and "Modify All" enabled
- Field-level security—set to visible and editable for all fields
- Apex classes—enabled
- Visualforce pages—enabled
- App settings—enabled
- Tab settings—determined by the package creator
- Page layout settings—determined by the package creator
- Record Type settings—determined by the package creator

After installation, if you have Enterprise, Performance, Unlimited, or Developer Edition, set the appropriate user and object permissions on custom profiles as needed.

Install for All Users

Specifies the following settings on all internal custom profiles.

- Object permissions—"Read," "Create," "Edit," "Delete," "View All," and "Modify All" enabled
- Field-level security—set to visible and editable for all fields
- Apex classes—enabled
- Visualforce pages—enabled
- App settings—enabled
- Tab settings—determined by the package creator
- Page layout settings—determined by the package creator
- Record Type settings—determined by the package creator



Note: The Customer Portal User, Customer Portal Manager, High Volume Customer Portal, Authenticated Website, Partner User, and standard profiles receive no access.

Install for Specific Profiles...

Enables you to choose the usage access for all custom profiles in your organization. You can set each profile to have full access or no access for the new package and all its components.

- Full Access—Specifies the following settings for each profile.

- Object permissions—"Read," "Create," "Edit," "Delete," "View All," and "Modify All" enabled
 - Field-level security—set to visible and editable for all fields
 - Apex classes—enabled
 - Visualforce pages—enabled
 - App settings—enabled
 - Tab settings—determined by the package creator
 - Page layout settings—determined by the package creator
 - Record Type settings—determined by the package creator
- No Access—Specifies the same settings as Full Access, *except* all object permissions are disabled.

You might see other options if the publisher has included settings for custom profiles. You can incorporate the settings of the publisher's custom profiles into your profiles without affecting your settings. Choose the name of the profile settings in the drop-down list next to the profile that you need to apply them to. The current settings in that profile remain intact.

Alternatively, click **Set All** next to an access level to give this setting to all user profiles.

2. Click **Install**. You'll see a message that describes the progress and a confirmation message after the installation is complete.

Post-Installation Steps

If the package includes post-installation instructions, they're displayed after the installation is completed. Review and follow the instructions provided. In addition, before you deploy the package to your users, make any necessary changes for your implementation. Depending on the contents of the package, you might need to perform some of the following customization steps.

- If the package includes permission sets, assign the included permission sets to your users who need them. In managed packages, you can't make changes to permission sets that are included in the package, but subsequent upgrades happen automatically. If you clone a permission set that comes with a managed package or create your own, you can make changes to the permission set, but subsequent upgrades won't affect it.
- If you're re-installing a package and need to re-import the package data by using the export file that you received after uninstalling, see "Importing Package Data" in the Salesforce Help.
- If you installed a managed package, click **Manage Licenses** to assign licenses to users.
- Configure components in the package as required. For more information, see "Configuring Installed Packages" in the Salesforce Help.

Component Availability After Deployment

Many components have an **Is Deployed** attribute that controls whether they are available for end users. After installation, all components are immediately available if they were available in the developer's organization.

For tips on customizing the installed package and components, see "Configuring Installed Packages" in the Salesforce Help. Installed packages are available to users in your organization with the appropriate permissions and page layout settings.

Uninstalling a Package

To remove a package:

1. From Setup, click **Installed Packages**.
2. Click **Uninstall** next to the package that you want to remove.

3. Select **Yes, I want to uninstall...** and click **Uninstall**.
4. After an uninstall, Salesforce automatically creates an export file containing the package data, as well as any associated notes and attachments. When the uninstall is complete, Salesforce sends an email containing a link to the export file to the user performing the uninstall. The export file and related notes and attachments are listed below the list of installed packages. We recommend storing the file elsewhere because it's only available for a limited period of time after the uninstall completes.



Tip: If you reinstall the package later and want to reimport the package data, see “Importing Package Data” in the Salesforce Help.

When you uninstall packages, consider the following:

- If you're uninstalling a package that includes a custom object, all components on that custom object are also deleted. This includes custom fields, validation rules, s-controls, custom buttons and links, as well as workflow rules and approval processes.
- You can't uninstall a package whenever any component in the package is referenced by a component that will not get included in the uninstall. For example:
 - When an installed package includes any component on a standard object that another component references, Salesforce prevents you from uninstalling the package. This means that you can install a package that includes a custom user field and build a workflow rule that gets triggered when the value of that field is a specific value. Uninstalling the package would prevent your workflow from working.
 - When you have installed two unrelated packages that each include a custom object and one custom object component references a component in the other, Salesforce prevents you from uninstalling the package. This means that you can install an expense report app that includes a custom user field and create a validation rule on another installed custom object that references that custom user field. However, uninstalling the expense report app prevents the validation rule from working.
 - When an installed folder contains components you added after installation, Salesforce prevents you from uninstalling the package.
 - When an installed letterhead is used for an email template you added after installation, Salesforce prevents you from uninstalling the package.
- You can't uninstall a package if a field added by the package is being updated by a background job, such as an update to a roll-up summary field. Wait until the background job finishes, and try again.
- Uninstall export files contain custom app data for your package, excluding some components, such as documents and formula field values.

Installing Managed Packages using the API

You can install, upgrade, and uninstall managed packages using the API, instead of the user interface. Automating these repeated tasks can help you can work more efficiently and to speed up application development.

To install, upgrade, or uninstall a package, use the standard Metadata API `deploy()` call with the `InstalledPackage` metadata type. The following operations are supported.

- Deploying an `InstalledPackage` installs the package in the deploying organization.
- Deploying a newer version of a currently installed package upgrades the package.
- Deploying an `InstalledPackage` using a manifest called `destructiveChanges.xml`, instead of `package.xml`, uninstalls it from the organization.



Note: You can't deploy a package along with other metadata types. Hence, `InstalledPackage` must be the only metadata type specified in the manifest file.

This is a typical project manifest (`package.xml`) for installing a package. The manifest must not contain a `fullName` or `namespacePrefix` element.

```
<?xml version="1.0" encoding="UTF-8"?>
  <Package xmlns="http://soap.sforce.com/2006/04/metadata">
    <types>
      <members>*</members>
      <name>InstalledPackage</name>
    </types>
    <version>28.0</version>
  </Package>
```

The package is specified in a file called **`MyNamespace.installedPackage`**, where **`MyNamespace`** is the namespace prefix of the package. The file must be in a directory called `installedPackages`, and its contents must have this format.

```
<?xml version="1.0" encoding="UTF-8"?>
  <InstalledPackage xmlns="http://soap.sforce.com/2006/04/metadata">
    <versionNumber>1.0</versionNumber>
    <password>optional_password</password>
  </InstalledPackage>
```

To uninstall a package, deploy this `destructiveChanges.xml` manifest file in addition to the `package.xml` file.

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

Retrieving an `InstalledPackage`, using the `retrieve()` call creates an XML representation of the package installed in an organization. If the installed package has a password, the password isn't retrieved. Deploying the retrieved file in a different organization installs the package in that organization.

For more information on the `deploy()` and `retrieve()` commands, see the [Metadata API Developer's Guide](#).

Resolving Apex Test Failures

Package installs or upgrades may fail for not passing Apex test coverage. However, some of these failures can be ignored. For example, a developer might write an Apex test that makes assumptions about a subscriber's data.

If you're a subscriber whose installation is failing due to an Apex test, contact the developer of the package for help.

If you're a developer and an install fails due to an Apex test failure, check for the following:

- Make sure that you are staging all necessary data required for your Apex test, instead of relying on subscriber data that exists.
- If a subscriber creates a validation rule, required field, or trigger on an object referenced by your package, your test might fail if it performs DML on this object. If this object is created only for testing purposes and never at runtime, and the creation fails due to these conflicts, you might be safe to ignore the error and continue the test. Otherwise, contact the customer and determine the impact.

EDITIONS

Available in:

- Developer

Running Apex on Package Install/Upgrade


App developers can specify an Apex script to run automatically after a subscriber installs or upgrades a managed package. This makes it possible to customize the package install or upgrade, based on details of the subscriber's organization. For instance, you can use the script to populate custom settings, create sample data, send an email to the installer, notify an external system, or kick off a batch operation to populate a new field across a large set of data. For simplicity, you can only specify one post install script. It must be an Apex class that is a member of the package.

The post install script is invoked after tests have been run, and is subject to default governor limits. It runs as a special system user that represents your package, so all operations performed by the script appear to be done by your package. You can access this user by using `UserInfo`. You will only see this user at runtime, not while running tests.

If the script fails, the install/upgrade is aborted. Any errors in the script are emailed to the user specified in the **Notify on Apex Error** field of the package. If no user is specified, the install/upgrade details will be unavailable.

The post install script has the following additional properties.

- It can initiate batch, scheduled, and future jobs.
- It can't access Session IDs.
- It can only perform callouts using an async operation. The callout occurs after the script is run and the install is complete and committed.

 **Note:** You can't run a post install script in a new trial organization provisioned using Trialforce. The script only runs when a subscriber installs your package in an existing organization.

How does a Post Install Script Work?

A post install script is an Apex class that implements the `InstallHandler` interface. This interface has a single method called `onInstall` that specifies the actions to be performed on installation.

```
global interface InstallHandler {  
    void onInstall(InstallContext context)  
}
```

The `onInstall` method takes a context object as its argument, which provides the following information.

- The org ID of the organization in which the installation takes place.
- The user ID of the user who initiated the installation.
- The version number of the previously installed package (specified using the `Version` class). This is always a three-part number, such as 1.2.0.
- Whether the installation is an upgrade.
- Whether the installation is a push.

The context argument is an object whose type is the `InstallContext` interface. This interface is automatically implemented by the system. The following definition of the `InstallContext` interface shows the methods you can call on the context argument.

```
global interface InstallContext {  
    ID organizationId();  
    ID installerId();  
    Boolean isUpgrade();  
    Boolean isPush();  
    Version previousVersion();  
}
```

Version Methods and Class

You can use the methods in the `System.Version` class to get the version of a managed package and to compare package versions. 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 non-patch release. Major and minor number increases will always use a patch number of 0.

The following are instance methods for the `System.Version` class.

Method	Arguments	Return Type	Description
<code>compareTo</code>	<code>System.Version version</code>	Integer	<p>Compares the current version with the specified version and returns one of the following values:</p> <ul style="list-style-type: none"> Zero if the current package version is equal to the specified package version An Integer value greater than zero if the current package version is greater than the specified package version An Integer value less than zero if the current package version is less than the specified package version <p>If a two-part version is being compared to a three-part version, the patch number is ignored and the comparison is based only on the major and minor numbers.</p>
<code>major</code>		Integer	Returns the major package version of the calling code.
<code>minor</code>		Integer	Returns the minor package version of the calling code.
<code>patch</code>		Integer	Returns the patch package version of the calling code or <code>null</code> if there is no patch version.

The `System` class contains two methods that you can use to specify conditional logic, so different package versions exhibit different behavior.

- `System.requestVersion`: Returns a two-part version that contains the major and minor version numbers of a package. Using this method, you can determine the version of an installed instance of your package from which the calling code is referencing your package. Based on the version that the calling code has, you can customize the behavior of your package code.
- `System.runAs(System.Version)`: Changes the current package version to the package version specified in the argument.

When a subscriber has installed multiple versions of your package and writes code that references Apex classes or triggers in your package, they must select the version they are referencing. You can execute different code paths in your package's Apex code based on the version setting of the calling Apex code making the reference. You can determine the calling code's package version setting by calling the `System.requestVersion` method in the package code.

Example of a Post Install Script

The following sample post install script performs these actions on package install/upgrade.

- If the previous version is null, that is, the package is being installed for the first time, the script:
 - Creates a new Account called “Newco” and verifies that it was created.
 - Creates a new instance of the custom object Survey, called “Client Satisfaction Survey”.
 - Sends an email message to the subscriber confirming installation of the package.
- If the previous version is 1.0, the script creates a new instance of Survey called “Upgrading from Version 1.0”.
- If the package is an upgrade, the script creates a new instance of Survey called “Sample Survey during Upgrade”.
- If the upgrade is being pushed, the script creates a new instance of Survey called “Sample Survey during Push”.

```
global class PostInstallClass implements InstallHandler {
  global void onInstall(InstallContext context) {
    if(context.previousVersion() == null) {
      Account a = new Account(name='Newco');
      insert(a);

      Survey__c obj = new Survey__c(name='Client Satisfaction Survey');
      insert obj;

      User u = [Select Id, Email from User where Id =:context.installerID()];
      String toAddress= u.Email;
      String[] toAddresses = new String[]{toAddress};
      Messaging.SingleEmailMessage mail =
        new Messaging.SingleEmailMessage();
      mail.setToAddresses(toAddresses);
      mail.setReplyTo('support@package.dev');
      mail.setSenderDisplayName('My Package Support');
      mail.setSubject('Package install successful');
      mail.setPlainTextBody('Thanks for installing the package.');
```

```
      Messaging.sendEmail(new Messaging.Email[] { mail });
    }
    else
      if(context.previousVersion().compareTo(new Version(1,0)) == 0) {
        Survey__c obj = new Survey__c(name='Upgrading from Version 1.0');
        insert(obj);
      }
    if(context.isUpgrade()) {
      Survey__c obj = new Survey__c(name='Sample Survey during Upgrade');
      insert obj;
    }
    if(context.isPush()) {
      Survey__c obj = new Survey__c(name='Sample Survey during Push');
      insert obj;
    }
  }
}
```

You can test a post install script using the new `testInstall` method of the `Test` class. This method takes the following arguments.

- A class that implements the `InstallHandler` interface.
- A `Version` object that specifies the version number of the existing package.

- An optional Boolean value that is `true` if the installation is a push. The default is `false`.

This sample shows how to test a post install script implemented in the `PostInstallClass` Apex class.

```
@isTest
static void testInstallScript() {
    PostInstallClass postinstall = new PostInstallClass();
    Test.testInstall(postinstall, null);
    Test.testInstall(postinstall, new Version(1,0), true);
    List<Account> a = [Select id, name from Account where name = 'Newco'];
    System.assertEquals(a.size(), 1, 'Account not found');
}
```

Specifying a Post Install Script

Once you have created and tested the post install script, you can specify it in the **Post Install Script** lookup field on the Package Detail page. In subsequent patch releases, you can change the contents of the script but not the Apex class.

The class selection is also available via the Metadata API as `Package.postInstallClass`. This is represented in package.xml as a `<postInstallClass>foo</postInstallClass>` element.

Running Apex on Package Uninstall

App developers can specify an Apex script to run automatically after a subscriber uninstalls a managed package. This makes it possible to perform cleanup and notification tasks based on details of the subscriber's organization. For simplicity, you can only specify one uninstall script. It must be an Apex class that is a member of the package.

The uninstall script is subject to default governor limits. It runs as a special system user that represents your package, so all operations performed by the script will appear to be done by your package. You can access this user by using `UserInfo`. You will only see this user at runtime, not while running tests.

If the script fails, the uninstall continues but none of the changes performed by the script are committed. Any errors in the script are emailed to the user specified in the **Notify on Apex Error** field of the package. If no user is specified, the uninstall details will be unavailable.

The uninstall script has the following restrictions. You can't use it to initiate batch, scheduled, and future jobs, to access Session IDs, or to perform callouts.

How does an Uninstall Script Work?

An uninstall script is an Apex class that implements the `UninstallHandler` interface. This interface has a single method called `onUninstall` that specifies the actions to be performed on uninstall.

```
global interface UninstallHandler {
    void onUninstall(UninstallContext context)
}
```

The `onUninstall` method takes a context object as its argument, which provides the following information.

- The org ID of the organization in which the uninstall takes place.
- The user ID of the user who initiated the uninstall.

The context argument is an object whose type is the `UninstallContext` interface. This interface is automatically implemented by the system. The following definition of the `UninstallContext` interface shows the methods you can call on the context argument.

```
global interface UninstallContext {
    ID organizationId();
    ID uninstallerId();
}
```

Example of an Uninstall Script

The sample uninstall script below performs the following actions on package uninstall.

- Inserts an entry in the feed describing which user did the uninstall and in which organization
- Creates and sends an email message confirming the uninstall to that user

```
global class UninstallClass implements UninstallHandler {
    global void onUninstall(UninstallContext ctx) {
        FeedItem feedPost = new FeedItem();
        feedPost.parentId = ctx.uninstallerID();
        feedPost.body = 'Thank you for using our application!';
        insert feedPost;

        User u = [Select Id, Email from User where Id =:ctx.uninstallerID()];
        String toAddress= u.Email;
        String[] toAddresses = new String[] {toAddress};
        Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
        mail.setToAddresses(toAddresses);
        mail.setReplyTo('support@package.dev');
        mail.setSenderDisplayName('My Package Support');
        mail.setSubject('Package uninstall successful');
        mail.setPlainTextBody('Thanks for uninstalling the package.');
```

```
        Messaging.sendEmail(new Messaging.Email[] { mail });
    }
}
```

You can test an uninstall script using the `testUninstall` method of the `Test` class. This method takes as its argument a class that implements the `UninstallHandler` interface.

This sample shows how to test an uninstall script implemented in the `UninstallClass` Apex class.

```
@isTest
static void testUninstallScript() {
    Id UninstallerId = UserInfo.getUserId();
    List<FeedItem> feedPostsBefore =
        [SELECT Id FROM FeedItem WHERE parentId=:UninstallerId AND CreatedDate=TODAY];
    Test.testUninstall(new UninstallClass());
    List<FeedItem> feedPostsAfter =
        [SELECT Id FROM FeedItem WHERE parentId=:UninstallerId AND CreatedDate=TODAY];
    System.assertEquals(feedPostsBefore.size() + 1, feedPostsAfter.size(),
        'Post to uninstaller failed.');
```

```
}
```

Specifying an Uninstall Script

Once you have created and tested the uninstall script and included it as a member of your package, you can specify it in the **Uninstall Script** lookup field on the Package Detail page. In subsequent patch releases, you can change the contents of the script but not the Apex class.

The class selection is also available via the Metadata API as `Package.uninstallClass`. This is represented in `package.xml` as an `<uninstallClass>foo</uninstallClass>` element.


Publishing Extensions to Managed Packages

An *extension* is any package, component, or set of components that adds to the functionality of a managed package. An extension requires that the base managed package be installed in the organization. For example, if you have built a recruiting app, an extension to this app might include a component for performing background checks on candidates.

The community of developers, users, and visionaries building and publishing apps on Force.com AppExchange is part of what makes Force.com such a rich development platform. Use this community to build extensions to other apps and encourage them to build extensions to your apps.

To publish extensions to a managed package:

1. Install the base package in the Salesforce organization that you plan to use to upload the extension.
2. Build your extension components.



Note: To build an extension, install the base package and include a dependency to that base package in your package. The extension attribute will automatically become active.
3. Create a new package and add your extension components. Salesforce automatically includes some related components.
4. Upload the new package that contains the extension components.
5. Proceed with the publishing process as usual. For information on creating a test drive or registering and publishing your app, go to <http://sites.force.com/appexchange/publisherHome>.



Note: Packages cannot be upgraded to Managed - Beta if they are used within the same organization as an extension.

EDITIONS

Available in:

- Group
- Professional
- Enterprise
- Performance
- Unlimited
- Developer

USER PERMISSIONS

To create packages:

- "Create AppExchange Packages"

To upload packages:

- "Upload AppExchange Packages"

CHAPTER 5 Passing the Security Review

In this chapter ...

- [Security Review](#)
- [Security Review Steps](#)
- [Security Review Wizard](#)
- [Submit a Client or Mobile App for Security Review](#)
- [Submit an Extension Package for Security Review](#)
- [Security Review Resources](#)
- [Security Review FAQ](#)

We want Force.com to be a safe and reliable platform for our customer's core business applications. To promote trust, all apps and components that are distributed to customers undergo a comprehensive security review. Your offering must meet or exceed the requirements set by our security review team before it can be distributed. After approval, apps and components are also periodically retested.

These guidelines and tasks will help you make sure that your offering passes the security review.

Security Review

The security review ensures that the app or component you publish on the AppExchange meets industry best security standards. For the latest information on the security review, visit: <http://p.force.com/security>.

The AppExchange security review:

- Assures customers that your app or component works securely with Salesforce.
- Helps you deliver apps and components that span multiple systems and meet the needs of AppExchange customers.
- Allows Salesforce to facilitate open relationships between customers, developers, and providers by providing a secure ecosystem.

The scope of the security review depends on the type of offering.

Type	Description	Scope of Review
Force.com	Offerings where the data, logic, and user interface is built entirely on the Force.com platform.	<ul style="list-style-type: none"> • Automated code scan • Manual code review and black-box testing • Client-side components (Flash, JavaScript) • Integrations and web services
Client and Mobile Apps	Offerings that run outside of the Salesforce environment. It treats the Force.com platform as a data source, using the development model of the tool and platform for which it's designed. Examples include iPhone apps and Microsoft Outlook connectors.	<ul style="list-style-type: none"> • Manual hands-on testing • Integrations and web services • Architecture review and web server testing
Web Apps	Offerings that run in a third-party hosted environment and integrate with Salesforce, leveraging the Force.com Web-services API. The data, logic, and user interface can be stored outside of Force.com.	<ul style="list-style-type: none"> • Automated testing and manual black-box testing • Client-side components (Flash, JavaScript) • Integrations and web services • Architecture review and web server testing

Security Review Steps

Follow these steps to have your offering reviewed for security compliance.

1. Prepare for the security review.
 - Read the security guidelines.
 - Review the preparation tips found at <http://p.force.com/security>.
 - Review the free resources listed on our [Secure Cloud Development](#) site.
 - Review the [Requirements Checklist](#).
 - Review the [OWASP Top Ten Checklist](#).
 - Run a free [self-service source code analysis](#) against code developed on the Force.com platform.
 - Run a [web-application scan](#) against your external web application that is integrated with Force.com.

- Manually test your offering to ensure that it meets review requirements not found by tools. For details, see the [OWASP Testing Guide](#).
- Fix any issues found during testing.

In general, be as thorough as you can in your testing. During the development phase of your app or component, run the code scanner several times to avoid fixing issues at the end. If you have questions, schedule office hours with the security review team at: <http://security.force.com/security/contact/ohours>.

2. Initiate the security review.



Note: Before you initiate the security review, configure a test environment that Salesforce can use to test your offering. For information about setting up a test environment, see [Required Testing Information for the ISV Security Review](#).

- Log in to the Partner Community.
- Open the security review wizard.
 - If your offering is a managed package, launch the wizard as follows.
 - On the Publishing page, click the **Packages** tab.
 - Find the offering that you want to submit and click **Start Review**.
 - If you have an API-only offering, launch the wizard as follows.
 - On the Publishing page, click the **Listings** tab.
 - Find the offering that you want to submit and click it to open the AppExchange publishing console.
 - Click the **App** tab and choose the API-only option.
 - Click **Start Security Review**.
- Follow the steps outlined in the security review wizard, which guides you through the rest of the submission.
- If this is a paid offering, pay the annual listing fee and the one-time security review fee. If your app or component is free, these fees are waived.

After you submit your package, the security review team runs tests to identify potential vulnerabilities. If necessary, they will contact you to discuss their findings. The review team performs both application and network security testing and sends you the results.


3. Review the results. There are three possible outcomes.

- **Approved**—You can list your app or component on the AppExchange and distribute it to customers immediately.
- **Provisional Pass**—The security review team identified low- or medium-risk issues that can be addressed easily and do not pose significant risks. You can create a public listing for your offering on the AppExchange and distribute it to customers. If you don't fix the issues identified within the specified time period, your app or component is removed from the AppExchange.
- **Not Approved**—The security review team identified high-risk issues during the testing phase. You can't list your offering on the AppExchange or distribute it to customers until all issues have been addressed and your offering has been reviewed again. If the app or component is already listed on the AppExchange, you must address the issues within 60 days. Because the security review is a black-box, time-limited process, we can't list every instance in which a particular issue occurred. Interpret these findings as representative examples of the types of issues you must fix across the offering.

Key Steps to Follow after Passing the Security Review

When you have passed the security review, you can:

- [List your offering publicly on the AppExchange](#) and distribute it to customers.
- [Request API access for your offering in Professional and Group Editions](#).

 **Important:** Salesforce reserves the right to conduct periodic reviews of all offerings. If we find that the app or component doesn't meet our security requirements, we notify you and provide time to remedy the issue. In extreme cases, we pull the AppExchange listing from public viewing. In this case, you must cease distribution of your offering.

Security Review Wizard

Use the online security review wizard to submit information about your offering to Salesforce for testing. The wizard is comprehensive, so give yourself plenty of time to respond to the questions. Be as thorough as you can, and remember that your responses are saved as you go—you can always return later to complete the process. The more information you provide, the faster we can test and approve your app or component.

The wizard consists of a series of screens that guide you through the information required.

1. Preparation

View tips and links to resources to help you prepare for the security review.

2. General Information

Add information for the person at your company who we can contact with security-related questions.

3. Policies and Certifications

Attach your company's information security policy and certifications that you've earned. For example, ISO 27001.

4. Components

List the technologies used by your app or component. You can select relevant items in a checklist based on the type. Examples include:

- Force.com — Apex, Visualforce, API, SSO
- Web app — frameworks and languages (Java, .NET, Rails, SSO, Heroku, and so on)
- Client app — desktop app, browser plug-in, Salesforce CTI toolkit implementation
- Mobile app — iOS, Android, BlackBerry, Windows

5. Test Environments

Provide fully configured environments for testing, including login credentials, install links, and sample data.

- Force.com — usernames and passwords for all user levels (admin, end user, and so on) in a test organization
- Web app — URLs, usernames, and passwords for all user levels, API keys, SSO, and OAuth/SAML settings
- Client app — install URLs, configuration data and instructions, required license files, associated sample data, config guides, credentials
- Mobile app — separate install link for each type of mobile app

6. Reports

Upload reports from your testing.

- Force.com — Security Code Scanner report



Note: Makes sure that the code scanner results are clean. If you're aware of issues in the scanner report that are false positives, provide the details.


- Web — Web App Scanner report
- Other — any other reports or documentation that you want to provide

7. Review Details

Review a summary of the information you've provided to verify that your submission is correct and complete. If there's something you'd like to change, you can modify it.

8. Payment

Pay for the security review using Recurly. Salesforce saves your payment information, so you only need to provide it once. If your app or component is free, no payment is required.

 **Important:** If you've already paid the security review fee for your offering, you aren't charged again. However, you're still asked to confirm the payment information every time you run the security review wizard.

Submit a Client or Mobile App for Security Review

Most of the client app requirements, and some of the web app requirements, apply to mobile apps. Here are typical scenarios:

- The mobile app has a Force.com component that sits on the customer's organization. The Force.com component is a managed package and follows the security review process for a packaged app.
- The client app only uses APIs for communicating with Salesforce. In this case, follow the process for an API-only app for security review.

For testing, we ask that you provision us an app for all the platforms that you plan to distribute. We can accept a test flight or an ad hoc deployment for iOS. For other platforms, we can accept the app in a file (APK, COR, and so on). Similar to a composite app, if there are callouts to anything other than Salesforce, we ask for a web application scanner report. We accept Zed Attack Proxy (ZAP) and Burp reports. If the mobile app has a web component, even if it's optional, Salesforce requires a web application scanner report.

Submit an Extension Package for Security Review

ISVs create extension packages when they want to provide add-on features to their apps. The extension packages also help when ISVs want to support Salesforce editions like PE and GE with their app. Another use case is creating a "bridge" package that enables the ISV's app to work with another app.

All packages, whether base or extension, require a security review. The same process needs to be followed for review of an extension package as for a base package.

Some extension packages are very small, for example, a few links or buttons to call base package components. Regardless of the size of the extension package, the same process needs to be followed. The only difference is that the review process is faster for smaller packages.

The process for submitting an extension package for the security review is similar to that for the base package.

1. Upload your extension package (it should be managed-release like your base package). Of course, the extension package can only be uploaded from an organization separate from that for the base package.
2. In your AppExchange listing, link the organization where the extension package was created. The extension package should appear in the list of packages under your listings.
3. Initiate the security review. Make sure your test account includes both the base and extension packages.

It's important that every extension package is reviewed and approved by the Salesforce security team. Even small packages can introduce vulnerabilities to the platform. Follow the same process of doing a self-scan of the code before submitting for a review. If the extension package has components that interface with an external application, run a web application scan, such as Zed Attack Proxy (ZAP) or Burp, and submit the corresponding results.

Security Review Resources

These resources can help you prepare for the security review.

- [Security Review Process](#)

- [Security Review Requirements Checklist](#)
- [Force.com Secure Cloud Development Resources](#)
- [Force.com Secure Coding Guidelines](#)
- [Open Web Application Security Project \(OWASP\)](#)
- [OWASP Top 10 Issues](#)
- [OWASP Testing Guide](#)
- [OWASP Secure Coding Guide](#)
- [OWASP Secure Coding Practices Quick Reference](#)

Security Review FAQ

This section contains a list of frequently asked questions about the security review.

- [Is an AppExchange security review required?](#)
- [What happens during a security review?](#)
- [Why do I need to have a security review?](#)
- [How long does the security review take? How often is it required?](#)
- [Is there a fee for the security review?](#)
- [Why do I have to test my offering before the review if the security team is going to test it anyway?](#)
- [What are the typical reasons why I would not pass the security review?](#)
- [Can I submit my offering before it's complete to get the security review process done early?](#)
- [If I have any "No" responses in the security review wizard, or no formal and detailed documentation, do I fail the review?](#)
- [Why does the review team need to test the X or Y part of my offering?](#)
- [Do I have to fix all the issues that the security review team reported?](#)
- [Why can't the review team send me every instance of every finding for my review?](#)
- [What happens after I pass the security review?](#)
- [What happens if my offering isn't approved?](#)
- [What's the difference between Approved, Provisional Pass, and Not Approved?](#)
- [When I update my offering, do I need to pay the security review fee again to have it reviewed?](#)
- [When I create a managed package to upgrade my offering, do I need to pay the security review fee again?](#)
- [Why perform periodic security reviews?](#)
- [How do reviewed solutions work with PE and GE organizations?](#)

Is an AppExchange security review required?

Yes. Every app and component that is publicly listed on the AppExchange must pass a security review.

What happens during a security review?

The security review process includes two kinds of assessments:

- Qualitative—question and answer round-to-review policies and procedures
- Quantitative—network and application penetration tests (see [Scope](#))

Why do I need to have a security review?

We want the AppExchange to be a trusted on-demand application ecosystem. The security review helps build this culture of trust by ensuring that our offerings adhere to security standards and best practices. This review lends credibility to the AppExchange and, in turn, helps increase customer adoption.

How long does the security review take? How often is it required?

The review process takes about 6–8 weeks, provided that:

- Your documentation is complete and accurate.
- The test environment is complete, fully configured, and includes all necessary information.
- You've met all the requirements.
- You're within the agreement guidelines.

Periodically, we review apps or components again. The timeline varies depending on the security risk of the offering, but it's usually between six months and two years since the last review. Around the expiry date, Salesforce contacts you to arrange another review.

Is there a fee for the security review?

If your offering is paid, Salesforce charges a fee for the initial review, and a small annual fee for subsequent reviews. If your offering is free, these fees are waived. Pricing information is available in the [Partner Community](#).

Why do I have to test my offering before the review if the security team is going to test it anyway?

By testing your offering before you submit, you're more likely to pass the review on the first round. Applicants who don't test beforehand rarely pass and must resubmit after addressing security concerns. Resubmitting significantly delays the publishing process.

What are the typical reasons why I would not pass the security review?

In no particular order, here's a list of the top reasons for not being approved. For more information, see [OWASP Top 10](#).

- Injection (SQL, XML, and so on)
- Cross-site scripting
- Broken authentication and session management
- Insecure direct object references
- Cross-site request forgery
- Security misconfiguration
- Insecure cryptographic storage
- Failure to restrict URL access
- Insufficient transport layer protection
- Unvalidated redirects and forwards

Can I submit my offering before it's complete to get the security review process done early?

No. It's important that the offering you submit is complete and the version that you intend to distribute. If the offering is not what you intend to distribute or is incomplete, we can't properly identify issues. We will need to review again, which delays your offering.

If I have any "No" responses in the security review wizard, or no formal and detailed documentation, do I fail the review?

No. Although the questions ask for formal and detailed documentation, we understand that, depending on the size and maturity of your company, this might not be possible. We designed the wizard to cover a broad spectrum of companies, from small to large. Smaller companies can provide us with a checklist instead of formal documentation of their security policies. If you have to answer "No" to some questions, use the comments box to explain. We understand that the scoring isn't black and white, and we factor in elements like company size and maturity to guide our decision-making.

Why does the review team need to test the X or Y part of my offering?

Our approach is to test all parts of the offering to ensure that our mutual customers and their data are not put at risk. This includes external web applications or services that are required or optional, client/mobile applications that are required or optional, and all Apex and Visualforce (packaged or unpackaged) that is included in the offering. If you're not sure if you should include part of your offering, include it anyway. The review team will not test parts of the offering that we determine are not in scope, but if a required part is not included, your review will be delayed.

Do I have to fix all the issues that the security review team reported?

Yes. Unless otherwise noted on the test reports, you're required to fix all classes of issues that the review team detected across the entire app or component.

Why can't the review team send me every instance of every finding for my review?

The security review is a black-box, time-limited review, and it would be impractical to provide that information given the visibility and time the review team has for each review. The findings should be interpreted as examples and all issues in the provided categories should be fixed across the offering.

What happens after I pass the security review?

After you pass the security review, you can list your offering publicly on the AppExchange and distribute it to customers. You're also eligible to [request an API token](#) if your app uses the SOAP or REST APIs.

What happens if my offering isn't approved?

If your offering isn't approved, you must address the security review team's findings and submit the app or component for a follow-up review. When your offering is approved, you can publish on the AppExchange and distribute it to customers.

What's the difference between Approved, Provisional Pass, and Not Approved?

If you're granted Approved or Provisional Pass on your security review, it means we didn't find any high-risk vulnerabilities in your offering. A Provisional Pass indicates that we likely found medium-risk vulnerabilities, and we work out a mutually acceptable timeline for you to remedy these items. In the meantime, you can list your app or component on the AppExchange and distribute it to customers. For a review that wasn't approved, you must remedy the issues found before you can list your offering on the AppExchange or get access to the API token.

When I update my offering, do I need to pay the security review fee again to have it reviewed?

No. When you upload a new package version to the AppExchange and associate it with your listing, we run a source code analysis to identify security vulnerabilities. This scan is included in the annual listing fee, so there is no extra charge. However, you are asked to confirm your payment information when you run the security review wizard.

When I create a managed package to upgrade my offering, do I need to pay the security review fee again?

No. If you developed the new version with a package that we've previously approved, it's automatically approved when you submit it for review. However, you are asked to confirm your payment information when you run the security review wizard.

Why perform periodic security reviews?

We require periodic security reviews for all apps and components published on the AppExchange. These reviews usually occur six months to two years after the initial approval, depending on the risk of the offering. Periodic reviews ensure that upgraded apps and components continue to meet security best practices and aren't subject to the latest security vulnerabilities. We also update our review process so that we're aligned with industry requirements.

How do reviewed solutions work with PE and GE organizations?

Eligible partners can apply for access to the API in the [Partner Community](#), which enables approved offerings to make API calls. If you're an eligible partner and have passed the security review, request API access by logging a case in the Partner Community. See [Use of ClientID](#) for code examples.

CHAPTER 6 Publish Your Offering on the AppExchange

In this chapter ...

- [What Is the AppExchange?](#)
- [Publish on the AppExchange](#)
- [Email Notifications](#)
- [Manage Billing and Subscriptions with AppExchange Checkout](#)
- [Work with AppExchange Leads](#)
- [Analytics Reports for Publishers](#)
- [Update the Package in an AppExchange Listing](#)
- [AppExchange FAQ](#)

The AppExchange publishing experience is managed from the Publishing page in the Partner Community. From the Publishing page, you can:

- Create listings or edit existing ones
- Connect the organizations that contain your packaged app, component, or trial template
- Manage license settings or start the security review process
- View the analytics for your published listings

What Is the AppExchange?

The AppExchange is an online marketplace for Salesforce apps, components, and consulting services. If you're a Force.com developer or consultant, the AppExchange is the gateway for connecting customers to your business solution. If you're a Salesforce administrator or user, visit the AppExchange to find tools and talent to unleash your company's productivity.

How Does the AppExchange Work?

An AppExchange listing is your primary marketing tool for promoting your app or component. In the listing, you can describe your solution, pricing, support, and other details so that customers can determine if your offering is right for them. You also have a chance to upload videos, white papers, and other content to help customers understand what you are delivering. Based on the information you provide, an AppExchange curator categorizes the listing into one or more business areas, like sales, marketing, or analytics.

After you've created a provider profile and uploaded your package, you can create a listing. You can create only one listing per app or component. This approach has several advantages. As the provider, it's easier to maintain and upgrade your offering over its lifecycle. Having one listing also helps you achieve a higher ranking, because the metrics that the AppExchange uses to rank apps and components, like page views, aren't diluted across multiple listings. Customers benefit, too, because your offering is easier to find, all your reviews are in one place, and there aren't several similar listings to cause confusion.

Who Can Use the AppExchange?

Anyone can browse listings and test-drive apps or components. You need the "Download Packages" permission to install apps or components. To create a package and upload it to the Partner Community, you must have "Create Packages" and "Upload Packages" permissions. To create and publish a listing, you must have the "Manage Listings" permission.

Publish on the AppExchange

To publish an app, component, or consulting service on the AppExchange, follow these high-level steps.

1. If your listing is an app or component, connect your packaging organization to the AppExchange.
2. Create a provider profile.
3. Review tips for creating a listing that excites and engages customers.
4. Create the listing.
5. If your listing is an app or component, submit the package for security review.
6. After your app is approved, publish the listing on the AppExchange.
7. Review the analytics to see how your listing is performing.

Connect a Packaging Organization to the AppExchange

To publish a listing on the AppExchange, first connect the packaging organization, which is the organization that contains your packaged app or component.

1. Log in to the Partner Community.
2. On the Publishing page, click the **Organizations** tab.
3. Click **Connect Organization**.
4. Enter the login credentials for the organization that contains the package you want to list.

5. Click **Submit**.

If the AppExchange finds any packages, they appear on the **Packages** tab on the Publishing page. From the **Packages** tab, create an app or component listing, or begin the security review.

Create or Edit Your Provider Profile

A polished, accurate provider profile is a key part of establishing customer trust in your app, component, or consulting service. On your profile, you can share a mission statement and tell customers where you're located, how many employees you have, and so on. People browsing listings see this information on the Provider tab.

To create or edit your profile, open the Publishing page in the Partner Community, and then click the **Company Info** tab.

Create or Edit an AppExchange Listing

Market your app, component, or consulting service with an AppExchange listing. Create a listing or edit an existing one in the AppExchange publishing console, which guides you through the process.

To create or edit a listing, open the Publishing page in the Partner Community, and then click the **Listings** tab.

Here are the tabs you navigate when creating or editing a listing.

Tab	What you do:	Available on these listings types:
Text	<ul style="list-style-type: none"> Describe your offering Provide contact information so that customers and Salesforce can get in touch 	App, Component, Consulting Service
Media	<ul style="list-style-type: none"> Add branding Upload images, videos, and other resources to help customers understand your offering 	App, Component, Consulting Service
App	<ul style="list-style-type: none"> Upload the package that contains your app (or the link to your app if you're only using the Salesforce API) 	App
Component	<ul style="list-style-type: none"> Upload the package that contains your component 	Component
Trials	<ul style="list-style-type: none"> Set up a test drive or free trial so that customers can see your offering in action 	App, Component
Leads	<ul style="list-style-type: none"> Choose how Salesforce collects leads when customers interact with the listing 	App, Component, Consulting Service
Pricing	<ul style="list-style-type: none"> Choose whether your offering is free or paid and provide pricing information 	App, Component
Service Offering	<ul style="list-style-type: none"> Choose listing categories, like services offered and industry focus 	Consulting Service

Make Your AppExchange Listing Effective

A great app, component, or consulting service deserves a listing to match. We gathered feedback from customers and Salesforce marketing experts to provide a list of tips to make your listing stand out.

Tell Customers, Then Show Them

An effective listing combines concise, customer-oriented writing with compelling visuals. As you craft your listing, keep the following tips in mind.

- **Emphasize a use case**—When customers read your listing, they want to understand the problem you’re solving, whether they’re part of the target audience, and what makes your offering different. As you explain your solution, put things in terms the customer cares about. For example, if your component helps support reps resolve cases 10% faster, say so.
- **Add screenshots, videos, and demos**—Customers are more likely to interact with listings that have visuals. Most people like to at least see how something works before making a purchase.
- **Make the listing easy to read**—Like you, the typical AppExchange customer is busy. Help customers understand what’s important by making your listing easy to read. Keep sentences short and use formatting, like bullets, to draw attention to key points. If you’ve added screenshots or a video, use zooming and annotations to highlight features.

Aim for Clean and Simple Design

An effective listing tends to have a clean and simple design. When making design decisions, keep the following tips in mind.

- **Find a design reference**—Before you create a logo, banner, or other graphic, find a design that you like and think about what it does well. For example, does it use a visually pleasing font? Keep these ideas in mind as you begin designing.
- **Preview before publishing**—The AppExchange lets you preview your listing before publishing, and you can see exactly how your offering will appear to customers. Put yourself in the customer’s shoes and ask, “If I saw this listing, would I feel comfortable buying this app or component?”

For more tips, see *Partner Logo and Branding Usage Guidelines* in the Education section of the [Partner Community](#).

Choose an Installation Option

The easier it is for people to install your offering, the more likely it is they will become paying customers. Choose the installation option that gives customers the best experience.

Option	When to choose this:
Directly from the AppExchange	If your offering is packaged, this option provides the simplest installation experience. It allows people to install your offering into their Salesforce sandbox or production environment through the AppExchange installation sequence without assistance from you. This option is required for components and recommended for apps.
From your website	If your app is a downloadable client or needs additional information to be installed, this option is the best. After users click Get It Now on your listing and agree to the terms and conditions, they are directed to your website to complete the installation process. Make sure that you’ve provided clear download instructions and performed the required setup or configuration.

Option	When to choose this:
They should contact us to install it	If your installation or selection process requires your assistance, you must choose this option. After agreeing to terms and conditions, the customer is told that you'll be in touch shortly to help with installation. Make sure that your company has the resources to assist potential customers.

Register Your Package and Choose License Settings

If you register a package and set up the License Management App (LMA), you receive a license record each time a customer installs your app or component. Licenses let you track who is using your app or component and for how long.



Note: Before you register a package, make sure that:

- Your app or component is in a managed package.
- You have installed the LMA. In most cases, the LMA is installed in your partner business organization.

1. Log in to the Partner Community.
2. On the Publishing page, click the **Packages** tab.
3. Click **Manage Licenses** next to the package that you want to register.
4. Click **Register**. Enter the login credentials for the organization where the LMA is installed. Usually, the organization is your partner business organization.
5. Select whether your default license is Free Trial or Active.
6. If you selected a free-trial license, enter the length of the trial, up to 90 days.
7. Enter the number of seats associated with your default license, or select **License is site-wide** to offer the license to all users in the installer's organization.
8. Click **Save**.

Submit Your Listing for Security Review

To distribute your app or component to customers, or provide a free trial, you must pass the AppExchange security review. This review ensures that your offering is safe for customers to install.



Important: You are contractually required to keep security review information current. For example, if you upgrade your component to use a new web service, you must edit the information in your security review submission.

Submit an App or Component Listing for Review

1. Log in to the Partner Community.
2. On the Publishing page, click the **Packages** tab.
3. Find the package that you want to submit, and then click **Start Review**.
4. Follow the steps in the security review wizard to complete the submission.

Within two days, a member of the partner operations team will contact you with next steps.

Submit a Trial Template for Review

Before you submit a trial template for security review, make sure that the packages that it contains have passed the security review.

1. Log in to the Partner Community.
2. On the Publishing page, click the **Trial Templates** tab.
3. Find the trial template that you want to submit, and then click **Start Review**.
4. Follow the on-screen prompts to create a profile in your trial organization that the security review team can use for testing. If you already have a profile, click **Skip to the next step**.
5. Enter the login credentials for the test profile.
6. Click **Submit Security Review**.

Email Notifications

Installation Notification Emails

Salesforce emails your subscribers 30 days after they install your app or component. The email thanks subscribers and encourages them to share their experiences with others by writing a review. We only send emails when:

- The subscriber has a valid email address.
- The subscriber hasn't already received a notification.
- The subscriber hasn't yet posted a review.

Review Notification Emails

When subscribers post reviews and comments on your listings, Salesforce emails parties who are likely to be interested. The notification that subscribers receive depends on their role in the conversation (provider, author, or commenter).

Type of Email Notification	Sent to	Details
New Review on Your Listing	You, the provider	Sent whenever someone posts a review of your listing.
New Comment on Your Review	The review author	Sent only if someone other than the review author comments on the review and if the author has opted to receive email notifications on their profile. If the author replies to the notification, the reply is posted as a new comment on the review.
Also Commented on the Review	The people who commented on the review	Sent to people who have commented on a review, are not the review author or the author of this comment, and have opted to receive email notifications on their profiles. At most, one email notification is sent to each commenter for each new comment. If the person replies to the notification, the reply is posted as a new comment on the review.
New Comment on the Review of Your Listing	You, the provider	Sent whenever someone writes a new comment on a review of your listing.

Manage Billing and Subscriptions with AppExchange Checkout

Checkout is a Salesforce service that lets your customers pay for apps and components directly on the AppExchange. The service bills your customer's credit card and deposits the money into your account. Checkout also handles your subscriptions, saving you the pain of managing recurring billing and licenses.

Checkout is powered by [Stripe](#), a simple, developer-friendly way to accept payments online. Stripe provides no-touch recurring billing and subscription management, including trials, coupons, upgrades and downgrades, and automatic renewals. You have no setup fees, monthly service charges, or card storage fees, and you can begin accepting payments immediately. And because Checkout integrates with the License Management App (LMA), licenses are always up to date.

What You Need to Know About Checkout

- Checkout collects credit card information, manages trial periods, processes charges, and updates licenses in the LMA.
- Checkout supports six pricing models.
 - Per user, billed monthly
 - Per user, billed annually
 - Per user, one-time payment
 - Per company, billed monthly
 - Per company, billed annually
 - Per company, one-time payment
- To use Checkout, you must distribute your app or component as a managed package. Checkout isn't compatible with OEM apps.
- The fee to use Checkout is 15% plus 30 cents per successful transaction.
- Stripe is working to expand global support. See the list of supported countries at <https://stripe.com/global>. If your country isn't listed, sign up to be notified when Stripe is available.

Sign Up for Checkout

Sign up for Checkout in the AppExchange publishing console.

1. Log in to the Partner Community.
2. In the publishing console, create an app or component listing. Alternately, edit an existing listing.
3. Click the **Pricing** tab.
4. Select **Paid, using Checkout**.
5. Click **Connect to Stripe**. You're taken to Stripe's website, where you can create an account and manage your subscription plans.

AppExchange Checkout FAQ

The following lists frequently asked questions about AppExchange Checkout.

- [Can I have multiple listings for one application?](#)
- [Do you support pricing that is not per user per month, for example, a price of \\$5,000 per year per Salesforce instance?](#)
- [Can I have multiple pricing plans available on my listing?](#)
- [How do I manage my pricing plans for Checkout?](#)

- Do customers have to purchase my app through the AppExchange or can I add them as customers in Stripe?
- Can my customer switch between payment plans on the AppExchange?
- Does a license automatically become inactive if a credit card is declined?
- Does Checkout replace the LMA? Will the LMA still control license status (Active/Trial) or are licenses active for any purchase made through Checkout?
- Does Checkout support payment in multiple currencies?
- Is Checkout available for all countries?
- Can I add value-added tax for transactions processed through Checkout?
- Is there a minimum revenue share? For example, if the 15% falls below \$1.00, is the revenue share still 15%?
- Does the revenue share apply to non-Checkout processed transactions?
- How does billing work when customers add/remove licenses during the month?
- How does Checkout affect existing processes, such as lead management, Trialforce and trial licenses? For example, how will Checkout signup interact with lead processing?
- How should I handle invoices for my customers?
- If an Admin installs the app, can someone else in the company go to their AppExchange account to Buy Now?

Can I have multiple listings for an app or component?

No, you can associate an app or component with only one listing. In addition, you can't duplicate a package (or create a new package version) just to list the app or component in a new listing. This behavior is to your advantage, because it's easier for you to maintain and upgrade the app or component over its lifecycle. It also helps your listing achieve a higher ranking in the AppExchange, because the metrics that Salesforce uses to rank apps and components, like page views, aren't diluted across multiple listings.

Do you support pricing that is not per user per month?

Yes, Checkout supports the following pricing models.

- Per user, billed monthly
- Per user, billed annually
- Per user, one-time payment
- Per company, billed monthly
- Per company, billed annually
- Per company, one-time payment

Create your pricing plans in Stripe, and then configure which plans to offer to customers on the **Pricing** tab for your listings on the AppExchange.

Can I have multiple pricing plans available on my listing?

Yes. AppExchange Checkout now supports multiple pricing plans on a single listing. For example, you can offer your app per user per month and per company per month, or offer a recurring plan as well as a one-time plan.

How do I manage pricing plans on a listing that uses Checkout?

To get your app or component ready for Checkout, first create the subscription plans in Stripe. After you've configured your plans, open the listing in the AppExchange publishing console. Then click the **Pricing** tab and choose your subscription plans. You can also add a one-time payment option, which is billed through Stripe.

Do customers have to purchase my app or component through the AppExchange or can I add them as customers in Stripe?

Don't add customers to plans in Stripe, because the AppExchange can't associate them with your listing or perform provisioning through the License Management App. Customers must always purchase your app or component through the AppExchange.

Can my customer switch between payment plans on the AppExchange?

No, customers can't switch payment plans through the AppExchange. They must contact you and you'll need to manually switch them to the new plan within Stripe.

Does a license automatically become inactive if a credit card is declined?

You specify what happens if a credit card is declined in your Stripe account settings. You can choose whether to retry the payment or deactivate the subscription. If the subscription is deactivated, the license will also become inactive.

Does Checkout replace the LMA? Does the LMA still control license status or are licenses active for purchases made through Checkout?

Checkout does not replace the LMA. Checkout works with the LMA to control whether an app or component is active or inactive, as well as how many licenses an organization has. Checkout creates and updates license records in the LMA, as necessary. You cannot directly edit the license records created by Checkout. To change a record, update the corresponding subscription information in Stripe. Checkout then updates the record in the LMA.

Does Checkout support purchases in multiple currencies?

For the list of currencies that Stripe supports, go to <https://support.stripe.com/questions/which-currencies-does-stripe-support>.

When you sign up for Stripe, you choose a default payment currency based on your country (for example, USD if you're in the United States). You can enable other currencies in your Stripe account settings. When customers purchase your app or component, Checkout charges them in your specified currency and deposits the money into your bank account.

Is Checkout available for all countries?

Stripe is working quickly to expand global support. See the list of supported countries here: <https://stripe.com/global>. You can use Checkout if you are based in a country listed on this page. If your country is not currently listed, sign up to be notified when Stripe is available in your country.

Can I add value-added tax for transactions processed through Checkout?

Yes. To charge value-added tax (VAT) on purchases, you enable VAT on your profile after signing up for Stripe.

Open the Publishing page in the Partner Community and click the **Company Info** tab. Select **Collect and remit Value Added Tax (VAT) on app purchases**. After you enter your VAT number, customers are charged VAT where applicable.



Note: VAT is not supported for one-time payments, and Stripe does not support VAT. You can apply VAT only for purchases completed through the AppExchange.

Does AppExchange Checkout have a minimum revenue share?

Apps or components sold via Checkout don't have a minimum revenue share. The fee for Checkout is always 15% plus Stripe's per transaction fee of 30 cents. The transaction fee is charged even if the price is less than \$1.00.

Does the revenue share apply to non-Checkout processed transactions?

The revenue share applies only to Checkout transactions—that is, charges in your Stripe account that are linked to an AppExchange listing where Checkout is enabled. As a Checkout partner, you agree that all purchases of your app or component take place through the AppExchange and are a part of revenue sharing.

How does billing work when customers add/remove licenses during the month?

Licenses added or removed part way through a month are charged a prorated amount for that month.

How does Checkout affect existing processes, such as lead management, Trialforce, and trial licenses? For example, how will Checkout signup interact with lead processing?

Checkout doesn't affect the way leads are sent to you, nor does it affect Trialforce configuration. However, it does change the way AppExchange trial licenses are managed. When a customer signs up for a trial using Checkout, the checkout trial user is listed as Active in the LMA.

How should I handle invoices for my customers?

Stripe generates invoices that customers can view directly on the AppExchange on the My Account page. You can also configure Stripe to email receipts to customers for each charge made to their credit card.

If an admin installs an app or component, can others in the company go to their AppExchange account to Buy Now?

Any user who has permissions to install apps or components can perform Buy Now actions, provided the user also has the "Manage Billing" permission. This permission is the same one required to view the My Account page or make purchases using Checkout inside the Salesforce app.

Work with AppExchange Leads

When a customer interacts with your listing, the AppExchange collects contact information that can be delivered to you as a lead.

You can collect leads when a customer:

- Installs your app or component

- Takes a test drive
- Watches a demo or video
- Signs up for a free trial
- Clicks the **Learn More** link

Before you enable lead collection on a listing, make sure that you've set up Web-to-Lead in the organization where you want to receive leads.

AppExchange Leads FAQ

This section contains a list of frequently asked questions about AppExchange Leads.

- [How do I receive leads from the AppExchange?](#)
- [Can I choose to receive leads for one listing but not others?](#)
- [How will I know this lead came from the AppExchange?](#)
- [Can Salesforce de-duplicate leads before they are sent to me?](#)
- [What do the lead source codes from my listing mean?](#)
- [How can I see more information in the lead record?](#)
- [What's the difference between the leads and license records my listing generates?](#)
- [What happens to my listing if I chose not to receive AppExchange leads?](#)

How do I receive leads from the AppExchange?

To receive leads from the AppExchange, edit your AppExchange listing and click the **Leads** tab. Specify the actions you want to receive leads for and to which Salesforce organization to send the leads. This organization must have Web-to-Lead enabled and be a standard Salesforce organization, not a Developer Edition organization. We recommend using your partner business org so that you can manage leads and licenses from a single, convenient location.

Can I choose to receive leads for one listing but not others?

Yes, you can enable lead collection for one listing but not others. Lead collection is enabled on a per listing basis. If you don't want to collect leads for a particular listing, don't choose that option in the publishing console.

How will I know this lead came from the AppExchange?

The lead source codes give you information about how the lead was created and can help you determine how to proceed. The lead source code always takes the form of `SFDC-XX|Listing Name` or `SFDC-dup-XX|Listing Name`. If the source code contains `-dup-`, AppExchange already sent you a lead for this user, listing, or action within the last 180 days. The `XX` identifies the action the user took to generate the lead.

Can Salesforce de-duplicate leads before they are sent to me?

A majority of the partners we polled requested that we send them all the leads. We designate duplicate AppExchange leads as follows: `SFDC-dup-XX|Listing Name`.

If the source code contains `-dup-`, then AppExchange already sent you a lead for this user, listing, or action within the last 180 days. For example, `SFDC-dup-DM|VerticalResponse for AppExchange` indicates a duplicate lead from a user who clicked to view the demo video on the VerticalResponse for AppExchange listing.

What do the lead source codes from my listing mean?

The lead source codes give you information about how the lead was created and can help you determine how to proceed. The lead source code always takes the form of `SFDC-XX|Listing Name` or `SFDC-dup-XX|Listing Name`. If the source code contains `-dup-`, AppExchange already sent you a lead for this user, listing, or action within the last 180 days. The `XX` identifies the action the user took to generate the lead.

Here's a table of the action codes and what they mean.

Lead Source Code	Description
IN	The user started the install process for your app or component by clicking Get It Now on your listing, agreeing to the terms and conditions, and clicking the install button on the confirmation page. The user might not have completed the installation or might have uninstalled the app or component. Use the License Management Application (LMA) to track who has the package installed.
DM	The user watched some or all of your demo.
LM	The user clicked Learn More on your listing. Note: Listings that previously had Learn More buttons now have Get It Now buttons and receive lead source codes with IN actions.
TS	The user clicked Get It Now on your listing and chose to start a new 30-day free trial of Salesforce and your app or component. These users might be preexisting Salesforce customers.
TD	The user took your test drive by clicking Test Drive on your listing.

How can I see more information in the lead record?

At this time, providers can't modify the lead form that customers are asked to fill out when they view a demo, access the test drive, install an app, or click to Learn More. Please indicate any improvements you wish to see on the [IdeaExchange](#).

What's the difference between the leads and license records my listing generates?

Leads and license records are generated by specific actions that a customer takes on your listing. If you've set up Web-to-Lead and enabled lead collection on your listing, leads are generated when a customer does any of the following: views a video or demo, clicks **Learn More**, takes a test drive, or installs your app or component. By contrast, license records are generated only when a customer installs your app or component. You must also have the License Management App enabled in your partner business org to receive licenses.

What happens to my listing if I chose not to receive AppExchange leads?

If you don't select a scenario for collecting leads, customers aren't prompted to fill out the lead sign-up form, and no leads are sent. Regardless of the lead settings, customers can still view the demo, take a test drive, click the **Learn More** link, and install your app or component.

Analytics Reports for Publishers

AppExchange analytics reports are powerful visual tools for understanding how your app, component, or consulting partner listing is performing. These reports provide metrics related to the web traffic, number of installations, and other user activities over time. By looking at the reports, you can quickly gain insights about the aspects of your listing that resonate with customers and which areas need refinement.

- To access a report for your listing, open the Publishing page in the Partner Community, and then click the **Analytics** tab.

Report Types

For app and component listings, the available reports are:

- Installs (Get It Now)
- Leads
- Resources & Promotions
- Test Drives, Demos & Screenshots
- Web Analytics

For a consulting partner listing, the available reports are:

- Leads
- Learn Mores, Videos & Screenshots
- Web Analytics

Report Attributes

All the reports share these common attributes.

Listing Name

The title of the listing shown at the top of every report.

Back to Publishing Home link

Returns you to the Publishing Home page.

Show Menu

Allows you to choose from one of the available reports. The reports are sorted alphabetically.

Date Range Menu

Allows you to choose the date range. Last 30 Days is selected by default.

Metrics

Report	Metrics
Installs (Get It Now)	Get it Now, Installs, Click-to-Install Ratio

Report	Metrics
Leads	Unique Leads, Duplicate Leads, Total Leads
Resources & Promotions	Case Studies, Data Sheets, Promotions, Customer Testimonials, Webinars, Customization Guides, Whitepapers
Test Drives, Demos & Screenshots	Test Drives, Demos, Screenshots
Learn Mores, Videos & Screenshots	Learn Mores, Videos, Screenshots
Web Analytics	Page Views, SEO Searches, Visits, Internal Searches, Unique Visitors

Line Graph

Shows one or more lines for each metric you've selected for display. Select the checkboxes beneath the graph for the metrics you want to see. By default, all metrics are included in the graph. The reports show metrics over time grouped by created date. When you click the graph, the date and selected metrics for that date display. Next to each metric, the number of items in the metric over the selected date range displays regardless of whether you have chosen to include the graph of that metric.

Table

Each report includes a table. The first column on all reports is the Date, and the rest of the columns correspond to the metrics associated with the report. The table shows 30 rows at a time. Click **Next** to see more data. By default, the table is sorted by date from oldest to newest. Change the sort order by clicking the column headers. Clicking the selected sort column a second time sorts the data in the opposite direction. The small triangle pointing up or down next to a column header indicates the sort direction and marks that column as the sort column.

Update the Package in an AppExchange Listing

If you add features to a published app or component, update your AppExchange listing so that new customers get access to the latest version. You can update a listing only if the app or component passed the security review within the past year. It must also share the same namespace as the version that passed the review.



Note: If the last security review was completed more than a year ago, the security review team will contact you to arrange a new review. Until then, you can continue to list the newer version.

1. Upload the new version of your package to the AppExchange.
2. Log in to the Partner Community.
3. On the Publishing page, click the **Packages** tab. If you developed the new package in the same organization as the previous version, the new package displays automatically. If you developed it in a different organization, first connect the organization that contains the new package on the **Organizations** tab.
4. Find the new package, and then click **Start Review**.
5. Fill out the self-evaluation questionnaire and click **Submit**. If your offering passed the security review within the last year, the new package is auto-approved, and its status changes to Reviewed; the status change can take up to 24 hours.
6. After your app or component is approved, navigate to the **Listings** tab and click the listing that you want to edit. This opens the AppExchange publishing console.
7. If you're updating an app, click the **App** tab. If you're updating a component, click the **Component** tab.
8. Click **Select Package**, and then find the new package you want to associate with the listing.
9. Click **Save**.

AppExchange FAQ

The following is a list of frequently asked questions about selling on the AppExchange.

- [Can I add more industries?](#)
- [Do I need an APO to publish my app or component on the AppExchange?](#)
- [Can I change my company name?](#)
- [Can I create my app or component on a Salesforce sandbox and upload it to the AppExchange?](#)
- [Can I edit a review?](#)
- [Can I keep the same listing but change the package it provides?](#)
- [Can I update my app or component with a new version or patch?](#)
- [How Do Customers Find My Listing?](#)
- [How do I edit a package after I've created a listing?](#)
- [How do I get an API token for my app?](#)
- [How do I increase my listing's popularity?](#)
- [How do I offer a free trial of my app or component?](#)
- [How do I see listings that Salesforce removed?](#)
- [How do I upgrade my customers to a new version?](#)
- [What's the difference between a free trial and test drive?](#)
- [Where can I share my ideas?](#)
- [Where can I write a review?](#)

Can I add more industries?

No. To prevent abuse, you can only specify two industries for each listing. If you cover more industries, mention them in the full or brief description of your listing.

Do I need an APO to publish my app or component on the AppExchange?

No, you no longer need an AppExchange Publishing Organization (APO) to publish your app or component on the AppExchange. You can now connect the organization where you developed the app or component directly to the AppExchange publishing console. To connect an organization, open the Publishing page in the Partner Community, and click the **Organizations** tab. Before connecting an organization, make sure that you have the "Manage Listings" permission in the Partner Community.

Can I change my company name?

Yes, you can change your company name and other aspects of your company profile. Open the Publishing page in the Partner Community, and navigate to the **Company Info** tab. You can change the company name, upload a logo, and modify other details on your company profile.

Can I create my app or component on a Salesforce sandbox and upload it to the AppExchange?

No. You can use a sandbox to install and test your app or component, but you must create and upload it using a Developer edition organization.

Can I edit a review?

You can edit reviews that you authored. You can comment on reviews that you did not write.

Can I keep the same listing but change the package it provides?

Yes, you can change the packages that are linked to your listing. First, make sure that you've uploaded the new package and, if the listing is public, that the package has passed the security review.

On the Publishing page in the Partner Community, navigate to the **Packages** tab and find the package associated with the listing that you want to update. Click **Edit Listing** to open the publishing console. If you're updating an app, you can add a package on the **App** tab. If you're updating a component, add it on the **Component** tab.

Can I update my app or component with a new version or patch?

Yes, but you must [submit the new package for an AppExchange Security Review](#) and [register the package with your License Management App \(LMA\)](#).

How Do Customers Find My Listing?

Customers can find your app, component, or consulting service in several ways. On the AppExchange, they can search using keywords or browse using categories. They can also find your listing via an external search provider, such as Google. Knowing how your listing is ranked in each of these scenarios helps you get the most visibility with potential customers.

- Most of the time, people look for apps, components, and consulting services by searching for a term (keyword) on the AppExchange home page. The AppExchange returns matching results and sorts them based on keyword relevance. Here are some tips on how this works.
 - If you include a keyword anywhere in your listing, your listing appears in the search results for that keyword.
 - Generally, a keyword's relevance is increased by appearing earlier in the listing.
 - Generally, a keyword's relevance is increased by appearing more than once in the listing. Note that listing a keyword multiple times in a row does not improve the listing's ranking.
 - When two or more keywords are searched, only listings with all keywords in the same order are returned. In addition, searches for multiple keywords also match camel-cased words (for example, a search for "Great App" matches "GreatApp").
 - Listings at the top of a search are the most relevant from both a keyword and popularity perspective.
- When people look for apps, components, or consulting services by browsing categories, the listings in a category are sorted based on popularity during the past 30 days. Popularity is based on the actions customers can take, such as installing an app or component, taking a test drive, watching a demo or other resources, and clicking the **Learn More** link. Activities that show greater commitment, such as installing an app or component, are weighted more heavily than activities showing less commitment, such as clicking screenshots. The number of reviews and the average rating on a listing do not contribute to the popularity of the listing.

- People can also sort results by rating, release date, name, or provider name. Search results ranked by rating are sorted first by the number of stars and then by the number of reviews. For example, a listing with 1 review and five stars is ranked above a listing with 20 four-star reviews.
- Because the AppExchange is a public website, search engines index your listing pages and return them in their search results. To improve your ranking with external search providers, make sure that you cross reference your listing URL on your website, blog, Facebook, and Twitter pages.


How do I edit a package after I've created a listing?

Log in to the Partner Community and navigate to the AppExchange Publishing page. Click the **Packages** tab to view a list of all packages uploaded to the AppExchange. From this list you can:

- Search for a package by keyword.
- Select **Unlisted Packages** from the drop-down list to see only the packages that haven't yet been linked to a listing.
- Click **Start Review** to begin the security review process.
- For a listed package, click **Edit Listing** to edit listing details, such as pricing information, banners, and logos.
- For an app or component in a managed package, click **Manage Licenses** to update the license settings for this package version, such as whether your offering is free or for sale, if and when it expires, and how many people in the installer's organization can access it.

How do I get an API token for my app?

To get an API token for your app, so it can use the SOAP and REST APIs, it must pass the AppExchange security review. You can then log a case in the [Partner Community](#) under the **AppExchange and Feature Requests > API Token Request** category. You need to specify the type of token (SOAP or REST) and if you're using OAuth.

 **Note:** This feature is available to eligible partners. For more information on the Partner Program, including eligibility requirements, please visit us at www.salesforce.com/partners.

How do I increase my listing's popularity?

Popularity is based on customer activity. The AppExchange measures everything users do on your listing: install, learn more, test drive, demo, view screenshots, white papers, or data sheets, and more. The AppExchange weighs the activity according to its level of importance as indications of interest and filters out attempts to abuse the system.

The AppExchange recalculates popularity daily and then summarizes and evaluates results over 30 days. When you browse by category, you see listings sorted by their relative popularity over the past 30 days.

How customers have reviewed or rated the listing does not affect popularity. AppExchange visitors can sort by rating if they're interested.

Here are a few hints on improving rankings.

- Include a test drive. People like being able to try out an app or component. The number of test drives influences popularity. You also get the added benefit of being able to collect leads.
- Add images. One of the first things that visitors do is click the **View Screenshots** button. Many people don't even look at a listing that doesn't have screenshots.
- Add resources that demonstrate how your app or component affects the customer's bottom line. For example, if you have research showing that a component helps support representatives resolve cases faster, include that information in a data sheet.
- Be up front with your pricing. If you don't include pricing on your listing, people become disinterested quickly.

How do I offer a free trial of my app or component?

When you're creating or editing a listing, the **Trials** tab asks whether you want to offer a free trial or test drive. A free trial lets customers try your app in an interactive organization that you've customized. A test drive lets customers try a read-only version of your app without logging in to Salesforce. For more information, see [Provide a Free Trial](#) on page 179.

How do I see listings that Salesforce removed?

The AppExchange doesn't allow you to view listings that Salesforce removed. However, you can view private listings, which can include listings removed by Salesforce, usually because of problems discovered during the periodic security review. To view your private listings, on the Publishing page, navigate to the **Listings** tab. Click **Private Listings** from the drop-down list.

How do I upgrade my customers to a new version?

Create a new version of your managed package and upload it in the released state. After you upload, you can share the Install URL with your existing customers so that they can upgrade. If you're deploying only a bug fix to your customers and want to upgrade them automatically, see "Scheduling Push Upgrades" in the Salesforce online help. You can use the License Management App (LMA) to find out which customers need to upgrade.

Customers can also check whether an upgrade is available by logging in to the AppExchange and viewing the My Account page. If a new version of the app or component is available, it appears on this page.

What's the difference between a free trial and test drive?

When you're creating or editing a listing, the **Trials** tab asks whether you want to offer a free trial of Salesforce and your app or component. The free trial is a non-production Salesforce organization that includes your package and sample data. If a customer chooses to purchase the app or component instead of letting the trial expire, the organization becomes a production version. We recommend that you write a data cleanup script and include a button in your app or component that gives customers the option to remove sample data.

You can also choose to offer a test drive, which is a read-only version of your app or component that all customers taking the test drive log in to. Like a free trial, a test drive uses Developer Edition organizations that include sample data and whatever configuration options you choose.

Where can I share my ideas?

You can share your ideas on how to improve the AppExchange or Salesforce partner programs in the Collaboration section of the [Partner Community](#). These ideas are only seen by Salesforce and other partners. To share ideas more publicly, please post them on the [IdeaExchange](#).

Where can I write a review?

On the listing page, click the number of reviews or **Write the first**. If there are already reviews, you are directed to the review page where you can click **Write a review**. Each user can write only one review per listing.



Important: You cannot write a review for your own listing. Please review the [Terms of Use](#) for AppExchange for additional legal information.

CHAPTER 7 Managing Orders

In this chapter ...

- [Key Objects in the Channel Order App](#)
- [Installing and Configuring the Channel Order App](#)
- [Managing Contract Terms and Product Catalogs](#)
- [Submitting and Managing Orders](#)
- [Using the Partner Order Submit API](#)

The Channel Order App (COA) allows partners to create, manage and submit orders directly to the salesforce.com Partner Operations team.

You should use this app if you are a partner with an OEM or ISVforce agreement. For OEM partners the app is for provisioning the underlying [salesforce.com](#) licenses as well as revenue sharing and for ISVforce partners it is for revenue sharing.

You will receive the COA when you get your ISV business org (it's part of the template). To start submitting orders through the COA, you need to be trained by our Partner Operations team. Once you have your ISV business org, have a finalized contract, and have passed security review for your app, you can sign up for COA training by logging a case in the [Partner Community](#) under **Orders & Contracts > Channel Order App**.



Note: Orders should be submitted based on the sales and licensing of applications to customers, as required by your partner agreement.

Key Objects in the Channel Order App

Before you start placing orders using the COA, you should be familiar with the following terms/objects.

- **Contract Terms:** This object defines how you are able to sell products to customers, according to your contract with salesforce.com.
- **Product Catalog:** This object defines the list of products you can sell under your contact and the associated revenue sharing details.
- **Service Order and Service Order Detail:** These two objects are used to submit the details of your orders to salesforce.com. The Channel Order App collects the necessary data and populates these objects automatically.
- **Customer:** This custom object in the Channel Order App package stores the name, org ID, and address information for each of your customers. The Service Order has a lookup to this object to obtain the customer information for your order.

Installing and Configuring the Channel Order App

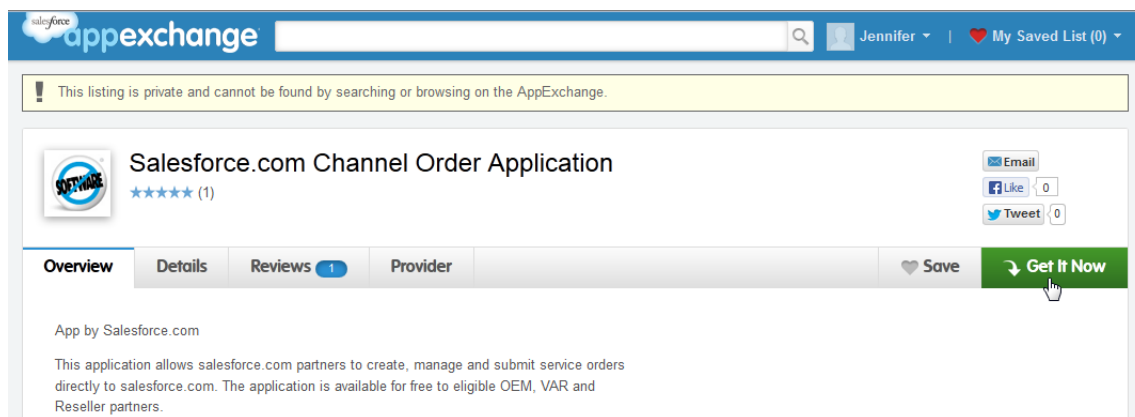
Before you use the Channel Order App, you must install it in your org and configure the associated resources. This section walks you through the installation and configuration process.

Install the Channel Order Application

This page explains how to get the Salesforce.com Channel Order Application from the AppExchange and install it in your production org.

Before you begin, confirm that you have admin access to the org you will use to submit and track production orders.

1. Go to the application listing for the Channel Order App on the AppExchange:
<https://appexchange.salesforce.com/listingDetail?listingId=a0N300000055ailEAA>
You must use the link; the listing is private and can't be found by searching or browsing.
2. If you are not already logged in, log in to the AppExchange using the credentials for the org where the app will be installed.
3. On the Salesforce.com Channel Order Application page, click **Get It Now**.



4. Click **Install in production**.
5. On the Installation Summary page, confirm that you are installing the app in the correct org. Review the Terms & Conditions and select "I have read and agree to the terms & conditions." Click **Confirm and Install**.

WHAT YOU ARE INSTALLING	WHERE YOU ARE INSTALLING
PACKAGE Salesforce.com Channel Order Application	ORGANIZATION Salesforce
VERSION SFDC Channel Order (1.17 / 1.17.0)	EDITION Developer Edition
SUBSCRIPTION Free	USER NAME devuser@de.com
DURATION Does Not Expire	
NUMBER OF SUBSCRIBERS Site-wide	

☒ I have read and agree to the [terms and conditions](#) .

[Cancel Install](#) | [Back to previous step](#) **Confirm and Install!**

- Enter your Salesforce administrative credentials again and click **Login**.
- Before beginning installation, the installer displays the Package Installation Details. Click **Continue**.

Package Installation Details [Help for this Page](#)

Package Name	SFDC Channel Order
Version Name	1.17
Version Number	1.17
Publisher	Channel Order Management
Description	

Continue [Cancel](#)

Package Components


▼ Pages (3)

Action	Component Name	Parent Object	Component Type	Installation Notes
Create	SFDCIntegrationCredentials		Visualforce Page	This is a brand new component.
Create	SendServiceOrder		Visualforce Page	This is a brand new component.
Create	PartnerTermsSyncBtn		Visualforce Page	This is a brand new component.

▼ Apps (1)

- In the Approve Third-Party Access popup window, review the list of websites and select "Yes, grant access to these third-party websites." Click **Continue**.

Approve Third-Party Access ✕

 This package may send or receive data from third-party websites. Make sure you trust these websites. [What if you are unsure?](#)

Website	SSL Encrypted
na1.salesforce.com	✓
login.salesforce.com	✓
na1-api.salesforce.com	✓

☒ Yes, grant access to these third-party web sites

Continue [Cancel](#)

- In Step 1: Approve Package API Access, the installer lists the permissions required by the Channel Order App. Review the page and approve the required access by clicking **Next**.

Package Installer
SFDC Channel Order

Help for this Page ?

Step 1. Approve Package API Access Step 1 of 3

These settings control the access that s-controls and other components in this package have to standard objects via the API. The access will still be constrained by the user's profile. You can view and edit the package API access to standard objects after the package is installed from the package detail page. [Tell me more](#)

Package Custom Objects
This Package will have the user's access (via the API) to all Custom Objects in your Organization.

Extended Object Permissions

	Read	Create	Edit	Delete
Accounts	✓	✓	✓	✓
Assets	✓	✓	✓	✓
Campaigns	✓	✓	✓	✓
Cases	✓	✓	✓	✓
Contacts	✓	✓	✓	✓
Contracts	✓	✓	✓	✓
Documents	✓	✓	✓	✓

	Read	Create	Edit	Delete
Ideas	✓	✓	✓	✓
Leads	✓	✓	✓	✓
Opportunities	✓	✓	✓	✓
Price Books	✓	✓	✓	✓
Products	✓	✓	✓	✓
Push Topics	✓	✓	✓	✓
Solutions	✓	✓	✓	✓

General User Permissions
This Package will be able to use all of the General User Permissions from the user's Profile.

Administrative Permissions
This Package will be able to use all of the Administrative Privileges from the user's Profile.

Next Cancel

10. In Step 2: Choose Security Level, you can define which user profiles can access the app. Select Grant access to admins only and click **Next**.

 **Note:** For details on how security settings affect profiles in your org, click the help link on the page.

Package Installer
SFDC Channel Order

Help for this Page ?

Step 2. Choose security level Step 2 of 3

Select security settings:

☒ Grant access to admins only Users with your profile get full access (best for limited deployments)

☐ Grant access to all users All internal custom profiles get full access

☐ Select security settings User access set by profile (recommended for most packages)


Previous Next Cancel

Next

11. Click **Install** to start installation.
12. Installation may take 2-3 minutes. If you don't see the Install Complete page, go to **Setup > Installed Packages** and look for SFDC Channel Order. If the package is not available, refresh the page every minute until you see the package.

Package Details Help for this Page ?

SFDC Channel Order (Managed)

 **Install Complete**
 Follow any remaining steps in the app install guide to complete deployment.

Installed Package Detail

[Uninstall](#)
[View Components](#)
[Manage Licenses](#)
[View Dependencies](#)

Package Name	SFDC Channel Order	Version Number	1.17
Language	English	First Installed Version Number	1.17
Version Name	1.17	Package Type	Managed
Namespace Prefix	CHANNEL_ORDERS	Allowed Licenses	Unlimited
Publisher	Channel Order Management	Used Licenses	0
Status	Active	API Access	Unrestricted [Enable Restrictions]
Expiration Date	Does not Expire	Modified By	Jennifer Shipman , 5/28/2013 10:16 AM
Description			
Installed By	Jennifer Shipman , 5/28/2013 10:16 AM		

Configure the Channel Order App: Define a New Email Service

The first step in configuring the Channel Order App is defining a new Email Service.

1. From Setup, click **Develop > Email Services > New Email Service**.
2. On the Email Service Information page, enter the following:

- Email Service Name: *SFDC Service Order*
- Apex Class: *ProcessServiceOrderStatus*
- Accept Attachments: *Text attachments only*
- Active (selected)

 **Note:** Leave all other fields blank, including Accept Email From.

Click **Save and New Email Address**.

Email Service [Help for this Page](#)

Email services let you use Apex classes to process the contents, headers, and attachments of inbound email. Use the settings below to create an email service. After saving, create one or more email service addresses to receive messages for processing.

Save Save and New Email Address Cancel

Email Service Information = Required Information

Email Service Name **SFDC Service Order**

Apex Class **ProcessServiceOrderS**

Accept Attachments **Text attachments only**

Advanced Email Security Settings ☐ [i](#)

Accept Email From

Convert Text Attachments to Binary Attachments ☐ [i](#)

Active ☒

Failure Response Settings

Configure how salesforce.com responds when an attempt to access this email service fails for the reasons shown below.

Over Email Rate Limit Action **Discard message**

Deactivated Email Address Action **Discard message**

Deactivated Email Service Action **Discard message**

Unauthenticated Sender Action **Discard message**

Unauthorized Sender Action **Discard message**

Enable Error Routing ☐ [i](#)

Route Error Emails to This Email Address

Save Save and New Email Address Cancel

3. On the Email Address Information page, enter the following:

- Email Address: `SFDC_Service_Order` (auto-populated)
- Active (selected)
- Context User: select a Salesforce admin user

 **Note:** Leave the Accept Email From field blank.

Email Service Address [Help for this Page](#)

Specify an email address for this email service. The email service processes messages sent to this address. One email service can have multiple email addresses.

Email Service Information

Email Service Name **SFDC Service Order**

Accept Email From **All email addresses (subject to security settings)**

Email Address Information = Required Information

Email address **SFDC_Service_Order**
Specify the local-part of the email address. Salesforce.com assigns the domain name part of the address.


Active ☒

Context User **Jennifer Shipman**

Accept Email From

Save Save and New Cancel

- Click **Save** or **Save and New**.

 **Note:** Copy the generated email address at the bottom of the confirmation page; you will need it in the next step.

Email Service: SFDC Service Order Help for this Page

[Edit](#) [Deactivate](#) [Cancel](#)

Email Service Name	SFDC Service Order
Apex Class	ProcessServiceOrderStatus
Accept Attachments	Text attachments only
Advanced Email Security Settings	<input type="checkbox"/> i
Accept Email From	All email addresses (subject to security settings)
Convert Text Attachments to Binary Attachments	<input type="checkbox"/> i
Active	<input checked="" type="checkbox"/>

▼ Failure Response Settings

Over Email Rate Limit Action	Discard message
Deactivated Email Address Action	Discard message
Deactivated Email Service Action	Discard message
Unauthenticated Sender Action	Discard message
Unauthorized Sender Action	Discard message
Enable Error Routing	<input type="checkbox"/> i

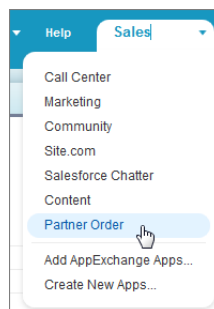
Email Addresses [New Email Address](#)

Action	Email Address	Context User
View Edit	sfdc_service_order@2wuhiooxw09h9xrtctth6wztpu1udqs3do5bewx5fdir3mrpz-i-z79qeas.il.apex.salesforce.com	Jennifer Shipman

Configure the Channel Order App: Provide Service Order Credentials

Before using the Channel Order App, you must provide Service Order Credentials and import your Contract Terms and Product Catalogs.

- In the Force.com App menu, click **Partner Order**.



- View all available tabs by clicking “+” next to the last tab.
- Click **Service Order Credentials** and enter the following information:
 - Service Type: `Production`
 - Username: Enter the user name provided to you by the Salesforce Partner Ops team. (This is different from the credentials used to log in to the org where the Channel Order App is installed.)
 - Password: Enter the password provided to you by the Salesforce Partner Ops team.
 - Login URL: This value is auto-populated for production orgs.

- Partner Org Email Address: Enter the email address you created in the previous step ([Configure the Channel Order App: Define a New Email Service](#)). If you need to find this email address again, from Setup click **Develop** > **Email Services** > **SFDC Service**.
 - End Point: This value is auto-populated for production orgs.
4. Click **Save** then **Test Connection** to verify the connection to Salesforce.
 5. For **Partner API Key**, enter the API key provided to you by the Salesforce Partner Ops team.
 6. Click **Import/Update Data** to import your Contract Terms and associated Product Catalogs.

Partner Order
SFDC Integration Credentials

Service Type: Production

Username: coa.abccorp@partnerforce.com

Password: ••••••••

Login URL: https://login.salesforce.com/services/Soap/u/26.0

Partner Org Email Address: sfdc_service_order@2wuhjooxw09h9xtctth6wztpu1udqs3do5bewox5fdir3mrpz.i-z79geas.il.apex.salesforce.co

End Point: https://na1-api.salesforce.com

Partner API Key: 382u34nereij2389u3

Test Connection Save

Import/Update Data

When the import is complete, the page will display a success message.

After you have completed the process above, your Contract Terms and Product Catalogs will appear in the Channel Order App. For details, see [Update Contract Terms and Product Catalogs](#) and [Add New Contract Terms](#).

Add Custom Fields to the Order Detail Page

You can add custom fields to the Order detail page to collect any data you want as orders are entered.

Follow the steps below to add custom fields to the Order detail page. All field types are accepted and the data entered is only stored locally in your org (not sent to salesforce.com).

1. First, create the custom fields on the Service Order object. From Setup, click **Create** > **Objects** and select the Service Order object. In Custom Fields & Relationships, click **New**.
For detailed instructions, see [Creating Custom Fields](#) in the Salesforce Help.
2. Add the new fields to the Custom Fields Field Set for the Service Order object and order them as they should appear on the Order detail page.
For details on field sets, see [Creating and Editing Field Sets](#) in the Salesforce Help.
3. When you save the additions to the Service Order object, a new section called Enter Custom Details will appear on the Order detail page that contains the custom fields.

4. Enter Service and Order Dates

Service Start Date

[12/12/2013]

Date Partner Received Customer Order

[12/12/2013]

5. Enter Custom Details

Lookup Account

Note:


Cancel

Save & Next

Enable the Channel Order App for Non-Admin Users

To make the Channel Order App accessible to users without admin permissions, modify the custom profile for those users.

You must have admin access to modify profiles.

 **Note:** The modifications required to access the Channel Order App can only be made to a custom profile. For details on creating custom profiles, see the Salesforce online help.

1. From Setup, click **Manage Users > Profiles** and find the custom profile you want to modify.
2. Click the profile name. On the Profile Detail page, click **Edit**.
3. Under Custom App Settings, find Partner Order and select the checkbox in the Visible column.

Profile Edit

Save Cancel

Name

Custom: Marketing Pro

User License

Salesforce

Description

Custom Profile

☒

Custom App Settings

Required Information

	Visible	Default		Visible	Default
Call Center	<input checked="" type="checkbox"/>	<input type="radio"/>	Sales	<input checked="" type="checkbox"/>	<input checked="" type="radio"/>
Community	<input checked="" type="checkbox"/>	<input type="radio"/>	Salesforce Chatter	<input checked="" type="checkbox"/>	<input type="radio"/>
Content	<input checked="" type="checkbox"/>	<input type="radio"/>	Sample Console	<input type="checkbox"/>	<input type="radio"/>
Marketing	<input checked="" type="checkbox"/>	<input type="radio"/>	Site.com	<input checked="" type="checkbox"/>	<input type="radio"/>
Partner Order	<input checked="" type="checkbox"/>	<input type="radio"/>			

Tab Settings

Partner Order Visible

4. Under Custom Tab Settings, set the Partner Order custom tabs as follows:

- Customers: Default On
- Orders: Default On
- Partner Contract Terms: Default On
- Partner Product Catalog: Default On
- Service Order Credentials: Tab Hidden

 **Note:** For security purposes, the Service Order Credentials tab must be hidden for non-admin profiles.


- Service Order Detail: Tab Hidden
- Service Orders: Tab Hidden

Custom Tab Settings	
Partner Contract Terms	Default On
Partner Product Catalog	Default On
Service Order Credentials	Tab Hidden

Service Order Detail	Default On
Service Orders	Default On

5. Under Custom Object Permissions, set the following access options:

- Customers: Read, Create, and Edit
- Partner Contract Terms: Read
- Partner Product Catalog: Read
- Service Orders: Read, Create, and Edit
- Service Order Detail: Read, Create, and Edit

 **Note:** You cannot update object permissions for a standard profile; to make these changes to a standard profile, you must clone the profile.

6. Click **Save**.

7. Back on the Profile Detail page, scroll down to Enable Apex Class Access and click **Edit**.

8. On the Enable Apex Class Access page, select all the classes under Available Apex Classes that begin with "CHANNEL_ORDERS" and click **Add** to move them to Enabled Apex Classes. Click **Save**.

Enable Apex Class Access

[Save](#) [Cancel](#)

Available Apex Classes	Enabled Apex Classes
--None--	CHANNEL_ORDERS.ProcessServiceOrderStatus CHANNEL_ORDERS.COAPartnerConfig CHANNEL_ORDERS.SFDCIntegrationCredentialsController CHANNEL_ORDERS.ServiceOrderProcessor

Add Rem Add

9. On the Profile Detail page, scroll to **Enable Visualforce Page Access** and click **Edit**.

10. On the Enable Visualforce Page Access page, select all the pages under **Available Visualforce Pages** that begin with "CHANNEL_ORDERS" and click **Add** to move them to **Enabled Visualforce Pages**. Click **Save**.

11. Back on the Profile Detail page, scroll to the Field-Level Security: Custom Field-Level Security section and complete the steps below for each of the following items:

- Customer
 - Partner Contract Terms
 - Partner Product Catalog
 - Service Order
 - Service Order Details
- a. Click **View** to open the field list.

Custom Field-Level Security	
Partner Contract Terms	[View]
Partner Product Catalog	[View]
Service Order	[View]
Service Order Detail	[View]
Service Order Log	[View]

- b. Click **Edit**.
- c. Select all checkboxes in the Visible column. Leave the Read-Only column as is.

Partner Contract Terms Field-Level Security for profile
Custom: Marketing Profile [Help for this Page](#)

Save Cancel

Field Name	Field Type	Visible	Read-Only
Allow Auto Renew Override	Checkbox	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Allow Contract Length Override	Checkbox	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Allow to Sell to Existing Org	Checkbox	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Created By	Lookup	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Default Auto Renew Policy	Picklist	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Default Contract Currency	Picklist	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Default Contract Length (Months)	Picklist	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Last Modified By	Lookup	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Name	Text	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Owner	Lookup	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Partner API Key	Text	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Partner Contract Term External ID	Text	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Type	Picklist	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Save Cancel

Visible

- d. Click **Save**.
- e. Click **Back to Profile**. (Repeat for the next object.)

After these steps are complete, users with the profile should be able to access the Channel Order App.

Managing Contract Terms and Product Catalogs

This section explains how to import your Contract Terms and Product Catalogs into the Channel Order App so you can create and submit orders.

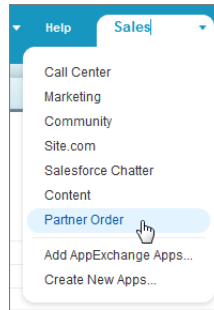
 **Note:** Contract Terms and Product Catalogs installed with the Channel Order App are owned and maintained by the Salesforce.com Partner Operations team and should not be modified.

Add New Contract Terms

To add Contract Terms to your Channel Order App, first obtain a new Partner API Key from salesforce.com, then follow the steps below to configure the new Contract Terms.

Each Contract Terms record is associated with a unique Partner API Key.

1. In the Force.com App menu, click **Partner Order**.



2. View all available tabs by clicking "+" next to the last tab.
3. Click **Service Order Credentials**. Leave all the login information as is and enter the new Partner API Key you obtained from salesforce.com.

 A screenshot of the 'Partner Order SFDC Integration Credentials' form. The form includes fields for Service Type (Production), Username (coa.abccorp@partnerforce.com), Password (masked), Login URL (https://login.salesforce.com/services/Soap/u/26.0), Partner Org Email Address (sfdc_service_order@2wuhjooxw09h9xtctth6wztpu1udqs3do5bewox5fdir3mrpz.i-z79geas.il.apex.salesforce.co), and End Point (https://na1-api.salesforce.com). There are 'Test Connection' and 'Save' buttons. Below these fields is a 'Partner API Key' field containing '382u34nereij2389u3' and an 'Import/Update Data' button.

4. Click **Import/Update Data**.
5. A confirmation message will display when the update is complete.

You can now manage the new Contract Terms as described in [Update Contract Terms and Product Catalogs](#).

Update Contract Terms and Product Catalogs

Salesforce.com Partner Operations will push any necessary updates to your Channel Order App, but if you think your Contract Terms or Product Catalogs are outdated, you can update them to the latest version yourself.

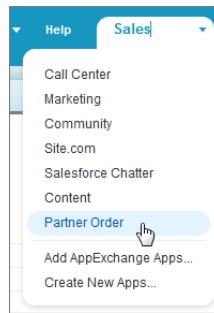
You can update your contract terms when you're submitting an order, or from a Contract Terms record.

To update your Contract Terms and Product Catalog when you're submitting an order, click **Refresh Product Details** on the order detail page.

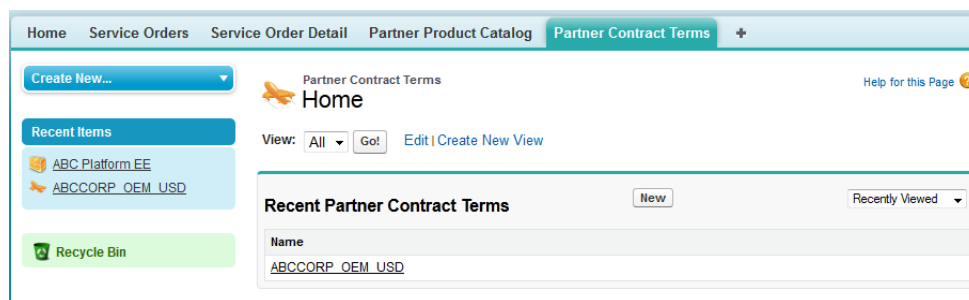
 A screenshot of the order detail page. It shows two sections: '2. Select a Contract' with a dropdown menu showing 'Salesforce_OEM_USD', and '3. Enter Order Details' with fields for Order Type (Initial), SFDC Invoice Description, and Related Opportunity. A red circle highlights the 'Refresh Product Details' button in the bottom right corner.

To update Contract Terms from a Contract Terms record, follow the steps below:

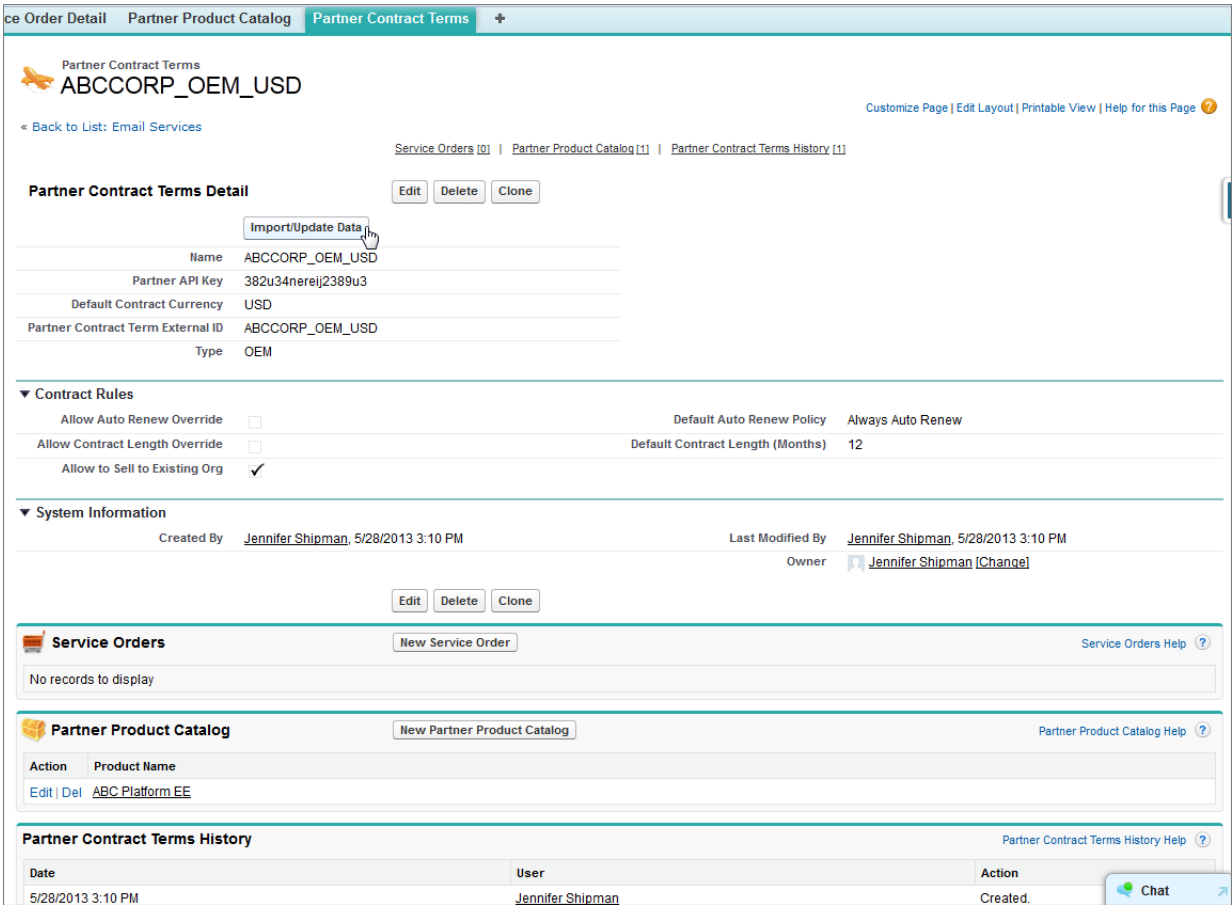
1. In the Force.com App menu, click **Partner Order**.



2. Go to the Partner Contract Terms tab and select the Contract Terms record you want to update.



3. On the Contract Terms record page, click **Import/Update Data** to update the Contract Terms and any associated Product Catalog records, and add any new Product Catalog records that have been added to your Contract Terms.



- 4. A confirmation message will display when the update is complete.

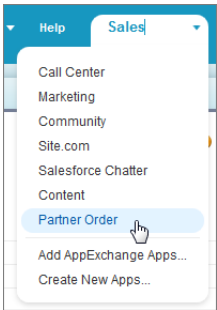
Submitting and Managing Orders

The Orders tab in the Channel Order App displays the status of your current orders and allows you to view, edit, and submit orders.

View and Manage Orders

Navigate to the Orders tab to view order status or edit and submit orders.

- 1. In the Force.com App menu, click **Partner Order**.



- Go to the Orders tab.

Order Number	Customer Name	Org ID	Contract	Order Start Date	Order Status	Action
SO-00000003	Central Park Law	00DU0000000HdJv	DO_OEM_USD1	8/8/2013	Draft	Edit
SO-00000005	Central Park Law	00DU0000000HdJv	DO_OEM_USD1	8/12/2013	Draft	Edit
SO-00000001	Central Park Law	00DU0000000HdJv	DO_OEM_USD1	8/10/2013	Received	View
SO-00000002	Central Park Law	00DU0000000HdJv	DO_OEM_USD1	8/9/2013	Received	View
SO-00000004	Central Park Law	00DU0000000HdXX	DO_OEM_USD1	8/8/2013	Received	View
SO-00000000	Central Park Law	00DU0000000HdJv	DO_OEM_USD1	8/8/2013	Received	View

- Use the search box to find orders by Customer Name or Org ID.
- Click any column to sort orders by the corresponding value.
- Use the Order Status column to track your orders.
- Click a link in the Action column to view a completed order or edit an unsubmitted order.
- Click **Submit a New Order** to enter details for a new order.

Submit an Order

Submit new orders from the Orders tab in the Channel Order App.

To start a new order, click **Submit New Order**. To edit an unfinished order or continue with an unsubmitted order, click the **Edit** link for the order in the Action column. The Order detail page allows you define the specifics of the order.

- Enter Customer Information: Enter the customer information in the fields provided. To auto-populate the details, search on the Company Name or Org ID. (If you use the License Management App to provision and control licenses, the Org ID is the same as the Subscriber ID on the license record.)

[Home](#)
[Orders](#)
[Partner Product Catalog](#)
[Partner Contract Terms](#)

Channel Order App - Submit New Order

1. Enter Customer Information

To begin your order, enter the customer information below. Customer data is stored once an order is submitted for the first time. To auto-populate the fields, search on the Company Name or Org ID.

Company Name

Customer Company

Org ID

Street

123 Main Street

City

San Francisco

Country

United States

State/Province/Region

California

ZIP/Postal Code

94121

2. Select a Contract

Salesforce_OEM_USD

3. Enter Order Details

Order Type

Initial

SFDC Invoice Description

Related Opportunity

Refresh Product Details

	Product	App	Pricing	Unit	Total Quantity	Customer Price (Unit/Month)	Total Customer Price/Month	Estimated Total SFDC Price/Month	Contract Length	Billing Frequency	Contract Auto Renew	Renewal Terms (Months)	Cancellation Terms (Days)
▼	Product 1	OEM App	% Net Revenue	Per User			--	--	1	1	Yes	12	1
▼	Product 2	OEM App	Fixed	Per User			--	--	1	1	Yes	12	1
▼	Product 3	OEM App	Fixed	Custom (Bundle)			--	--					

4. Enter Service and Order Dates

Service Start Date

[1/7/2014]

Date Partner Received Customer Order


[1/7/2014]

Date Customer Accepted SFDC Service Agmt


[1/7/2014]

Cancel

Save & Next

 **Note:** The screen you see may not look like the image above. The information and options on the page change based on the customer's contract terms and the order type.

2. Select a Contract: Choose the appropriate contract from the list.
3. Enter Order Details: Define the specifics of the order, including the type of order and the products to include.

 **Note:** If you have recently upgraded the Channel Order App, or think your Contract Terms or Product Catalogs are outdated, click **Refresh Product Details** to sync with the AppExchange and update data for the contract selected and the associated product catalogs. The page will display a Success message when the sync is complete.

- Choose the Order Type.

Order Type	Use to:
Initial	Create an order for a new customer who has never purchased under the contract before
Add-On	Add licenses or products for an existing customer
Reduction	Reduce licenses or products for an existing customer

Order Type	Use to:
Cancellation	Terminate the contractual relationship with an existing customer
Upgrade — Partner App	Upgrade an existing customer to an expanded version of your app
Upgrade — Org Edition	Upgrade an existing customer to an expanded version of Salesforce


- Enter an optional `SFDC Invoice Description`. This field should contain anything extra you want included on the invoice, including your identification information.
- Select an optional `Related Opportunity` to associate an opportunity from your org with the order. This data will not be sent to salesforce.com.
- Modify the order details as necessary using the fields provided. Quantity is always required, and customer price is required for % Net Revenue products. Many fields on this page are defined by your product catalog and contract. Use the action menu next to each product listing to view product details, add line item descriptions that will appear on your invoice from salesforce.com, or to clone a product line item if you sell it at two price points to the same customer.

3. Enter Order Details

Order Type SFDC In

Re

Product	App	Pricing	Unit	Total Quantity	Customer Price (Unit/Month)	To f
Product 1	OEM App	% Net Revenue	Per User	<input type="text"/>	<input type="text"/>	
View Product Details						
Clone This Product						
Add Invoice Description						
		Fixed	Per User	<input type="text"/>		
		Fixed	Custom (Bundle)	<input type="text"/>		

 **Note:** If the selected contract allows you to override contract or billing terms, you can update those terms on this page. The updated terms will be applied to all products in the order.

Column	Describes:
Product	The name of the product. Click a link to view details about that product.
App	The name of the app.
Pricing	The type of pricing associated with the product, according to your contract: <ul style="list-style-type: none">— Fixed— % Net Revenue
Unit	The unit associated with the pricing for the product <ul style="list-style-type: none">— per user

Column	Describes:
	<ul style="list-style-type: none"> per org custom units
Total Quantity	Enter the quantity of the product to include in the order.
Customer Price (Unit/Month)	Enter the customer price per unit per month. Only required for % Net Revenue products.
Total Customer Price/Month	The amount the customer will owe you based on the quantity and price entered for the order. Only calculated for % Net Revenue products.
Estimated Total SFDC Price/Month	The amount you will owe salesforce.com based on the quantity entered and your product catalog and contract.
Contract Length	The length of the contract in months.
Billing Frequency	The length of the Salesforce billing period in months.
Contract Auto Renew	Whether or not the contract will be renewed automatically.
Renewal Terms (Months)	The term for renewal in months.
Cancellation Terms (Days)	The term for cancellation in days.



Note: You can include custom fields to collect additional data about orders; for details, see [Add Custom Fields to the Order Detail Page](#).

4. Enter Service and Order Dates: For each field, click the text box to pick a date or click the link to choose today's date.
5. Click **Save & Next** to continue or save the order with an `Order Status` of *Draft*. If there are any errors in your order, a detailed message will be displayed. After you save an order, you can exit the application and return to the order later.
6. The Channel Order App – Order Confirmation page displays details about your order. If the details are correct and you agree to the stated terms, click **Confirm & Submit**. To make changes, click **Modify**.

After your order is successfully submitted, it will appear in the list on the Orders tab with an `Order Status` of *Received*. Orders that have not been submitted have an `Order Status` of *Draft*.

Using the Partner Order Submit API

The Channel Order App Partner Order Submit API allows you to submit orders programmatically via Apex from the ISV org where the Channel Order App is installed directly to Salesforce for provisioning of base licensing for your app.

Partner Order Submit API

The Partner Order Submit API can be used to submit orders immediately or asynchronously.

Syntax

```
channel_orders.ServiceOrderProcessor.sendOrder()
channel_orders.ServiceOrderProcessor.sendOrderAsync()
```

Usage

Use `sendOrderAsync` when you want to create or update multiple orders and send them in the same transaction. See the example in this section for more details.

Rules and Guidelines

This is an Apex implementation, so all Apex usage rules and limits apply. Currently we only support 1 order per call.

The Partner Submit API is used to send an order after it has been created, using a valid `Service Order ID`. You can create Service Order and Service Order Detail records using the Channel Order App, data loading, or automated processing.

Each order must include the following fields:





 **Note:** Field names are prefixed with `CHANNEL_ORDERS__` unless noted.

Table 4: Service Order Fields

Field Label	Field Name	Description
Created with New COA	<code>Created_with_new_COA__c</code>	Indicates you are using the latest version of the Channel Order App.  Note: Always check this field to ensure proper processing.
Contract	<code>Partner_Contract_Rules__c</code>	This field is a lookup to the Contract Terms record.
Customer Company Name	<code>Customer__c</code>	This field is a lookup to the Customer record, which includes the address and org details.  Note: You must use an existing Customer record; you can't populate the customer name and address fields using the API.
Date Partner Received Customer Order	<code>Date_Partner_Received_Customer_Order__c</code>	The date you received the order.
Date Customer Accepted SFDC Service Agreement	<code>Date_Customer_Accepted_SFDC_Svc_Agrmnt__c</code>	The date the customer accepted the salesforce.com service

Field Label	Field Name	Description
		agreement. Only required for OEM contracts.
I Certify a Corresponding Order is Rec'd	I_certify__c	Confirmation that the order has been received.
Order Type	Order_Type__c	Accepted values: <ul style="list-style-type: none"> Initial Add-On Reduction Cancellation Order Upgrade-Partner App Upgrade-Org Edition
Service Order ID	Name	The ID of the service order. (Standard field; not prefixed with CHANNEL_ORDERS__.)
Service Order Status ¹	Service_Order_Status__c	The current status of the service order: <ul style="list-style-type: none"> Draft Error Received Processed  Note: You can only submit orders with a status of Draft.
Service Start Date	Service_Start_Date__c	The date service should begin.

 **Note:**

- ¹ In certain situations, a query to the Partner Order Submit API may return `Service_Order_Status1__c` in addition to `Service_Order_Status__c` in the REST response. The `Service_Order_Status1__c` field is for internal use by Salesforce and can be ignored.

Table 5: Service Order Details Fields

Field Label	Field Name	Description
Billing Frequency	pc_Billing_Frequency__c	Must match contract with salesforce.com unless you have been granted override permissions.

Field Label	Field Name	Description
Cancellation Terms (days)	pc_Cancellation_Terms__c	Must match contract with salesforce.com unless you have been granted override permissions.
Contract Auto Renew	pc_Contract_Auto_Renew__c	Must match contract with salesforce.com unless you have been granted override permissions.
Contract Length	pc_Contract_Length__c	Must match contract with salesforce.com unless you have been granted override permissions.
Customer Price	Customer_Price_Per_Month__c	The price in units per month. Only required for PNR products.
Partner Contract Term	pc_Partner_Contract_Term__c	This field is a lookup to the Contract Terms record.
Product	Product_Name__c	This field is a lookup to the Product Catalog record.
Renewal Terms (months)	pc_Renewal_Terms__c	Must match contract with salesforce.com unless you have been granted override permissions.
Service Order	Partner_Order__c	This field is a lookup to the Service Order.
SFDC Invoice Description	Product_Line_Description__c	Optional. This field should contain anything extra you want added to the invoice, including your identification information.
Total Quantity	Quantity__c	The total quantity of Product Catalogs in the service order.

Methods

The ServiceOrderProcessor object supports the following methods:

Name	Arguments	Description
sendOrder	ID	Submit an order with a single ID immediately.
sendOrder	Set of IDs	Submit an order with a set of IDs immediately.
sendOrderAsync	ID	Submit an order with a single ID asynchronously (@future).
sendOrderAsync	Set of IDs	Submit an order with a set of IDs asynchronously (@future).

Example: Batching on the Partner Order Submit API

As noted above, only 1 call can be processed at a time. This example uses a batch job to work around this limitation.

In the example below, if you have 100 orders in `Draft` status, the code will create one batch job with 100 executions, because only one record is processed per execution, in the last line in the example.

```
//Batch Apex class
global class COABatchClass implements Database.batchable<sObject>, Database.AllowsCallouts,
Database.Stateful{
    final String DRAFT_STATUS = 'Draft';
    global final String query =
        'select Id, CHANNEL_ORDERS__Service_Order_Status__c ' +
        ' from CHANNEL_ORDERS__Service_Order__c where CHANNEL_ORDERS__Service_Order_Status__c
=: DRAFT_STATUS';

    global Database.QueryLocator start(Database.BatchableContext BC){
        return Database.getQueryLocator(query);
    }

    global void execute(Database.BatchableContext info, List<CHANNEL_ORDERS__Service_Order__c>
scope){
        for(CHANNEL_ORDERS__Service_Order__c s : scope){
            CHANNEL_ORDERS.ServiceOrderProcessor.sendOrder(s.Id);
        }
    }
    global void finish(Database.BatchableContext BC){}
}

//Batch call
Id batchInstanceId = Database.executeBatch(new COABatchClass(), 1);
```

CHAPTER 8 Managing Licenses

In this chapter ...

- [License Management App \(LMA\) Overview](#)
- [Installing the LMA](#)
- [Configuring the LMA](#)
- [Associating a Package with Your LMO](#)
- [Using the LMA](#)
- [Integrating with Sales and Marketing](#)
- [Best Practices](#)
- [License Management App FAQ](#)
- [Troubleshooting](#)


If you have installed the License Management App (LMA) you can track licenses and leads, and provide administrative support for your customers. These topics are covered in detail in the following sections.

Note that AppExchange provides a standard mechanism for presenting a license agreement to the installer and requiring acceptance of this agreement before continuing with the installation. If you want to require an agreement, paste the text of your license agreement in the `License Agreement` text field on the app listing page on AppExchange directory.

If your app requires a more interactive license negotiation, make sure your About tab splash page clearly details the required end user license agreement and states your preferred process for getting started. In all cases, Salesforce includes a disclaimer that limits our liability for the app

License Management App (LMA) Overview

The License Management App (LMA) helps developers and publishers apply licensing to their uploaded and registered AppExchange apps. Each time a customer or a prospective customer installs your managed package, a lead record and a license record are created in the LMA. By specifying defaults for the license records, you can apply licensing to control how many users in the customer organization can access your package, and for how long. Using the LMA, you can keep track of how many customers have installed a package and which version of the package they currently use. You can also use it to manage the leads associated with the licenses.

 **Note:** The LMA is available in English only.

The LMA is developed by Salesforce and is available to eligible partners. For more information on the Partner Program, including eligibility requirements, please visit us at www.salesforce.com/partners.

Understanding License Management

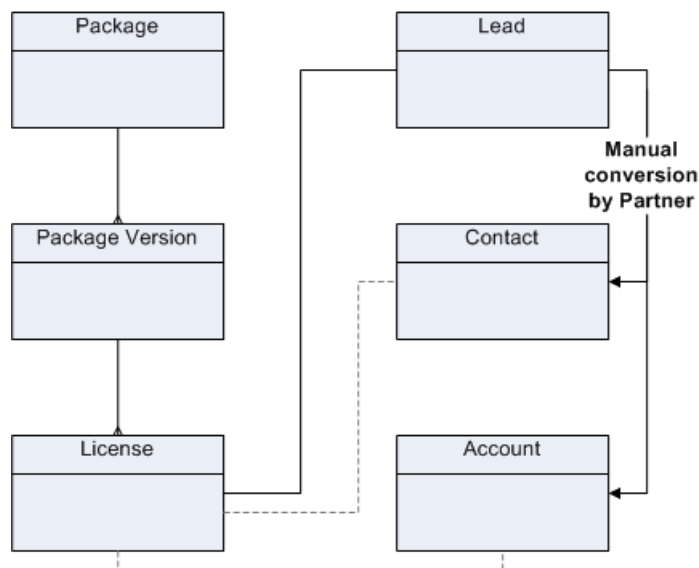
The license management process begins when someone installs an app from AppExchange. Salesforce automatically performs the following actions.

- Creates a license in the installer's organization. A copy of that license is stored in the Licenses tab of the LMA installed in your LMO.
- Updates the package version with the license information.
- Creates a lead with the installer's name, company, and email address. The lead source for installers is always *Package Installation*.

Use the LMA to manage licenses for managed packages. By modifying the license records, you can control how many users in the customer organization can access your package, and for how long.

Entity Relationship Diagram

The LMA has three custom objects that directly and indirectly relate to standard Salesforce objects. See the entity relationship diagram below to understand how the custom objects relate to other objects.



- The package object is the root object for all information in the LMA.
- Each package can have multiple package versions. Package version has a master-detail relationship with the package object.
- Each package version can have multiple licenses. License has a lookup relationship with the package version object.
- Each license has a direct relationship with a single lead. License has a lookup relationship with the lead object.
- The lead can be manually converted into an account and contact. When that happens, the license record is automatically associated with the converted account and contact records. License has a lookup relationship with the account and contact objects.

LMA Terminology

Apps

A collection of Salesforce components such as tabs, reports, dashboards, Apex classes and triggers, etc., that addresses a specific business need. You store an app in a package to upload it to AppExchange.

Developer

The developer of an app is the Salesforce user who created the app and its components. The developer of an app can be the same as or different from the [publisher](#).

Installer

The installer of an app is the person who installed the app from Force.com AppExchange into their Salesforce organization.

Lead Manager

The lead manager is the user responsible for following up on the leads that are automatically created when a managed package is installed. New leads are assigned to the license manager by default. The publisher (who owns the LMO) uses the lead manager to follow-up on customers who uninstall the managed packages.

License

An installation of a package by a Salesforce organization. Each license entry in the LMA represents an installation of a managed package.

License Manager

The license manager user is automatically created when an organization installs the LMA. A corresponding Package License Manager profile is also created. This user is the default owner for all packages managed using the LMA and all licenses created for those packages.

License Management Organization (LMO)

A license management organization is the Salesforce organization that you use to track all the Salesforce users who install your package. A license management organization must have the LMA installed. It automatically receives notification every time your package is installed or uninstalled so that you can easily notify users of upgrades.

Managed Package

Managed packages are packages that can be upgraded in the installer's organization. They differ from unmanaged packages in that some components are locked, allowing for the upgrade process.

Package

The container for an app and its components. In the LMA, the term package refers only to managed packages. A package may contain zero, one, or many apps.

Package License Manager

The Package License Manager profile gives a single user, the license manager, full access to the LMA. You cannot edit this profile or assign it to any other users.

Package Version

A specific version of a managed package. A new version is created when the developer uploads the updated package from a developer organization.

Publisher

The publisher of an app is the Salesforce user or organization that published the app on AppExchange. After registering their app, publishers are required to register using their contact information.

Installing the LMA

Before you install the LMA, you need to decide which organization is your License Management Organization (LMO). Because the LMA is an integral part of the sales, billing, and marketing of a managed package, putting the LMA in the wrong organization makes it difficult to manage licenses as a part of the sales and marketing process.

The LMA creates lead and license records when customers install a trial package. When they buy the package, those licenses are converted to paid licenses (either by site or based on a number of users). The LMA tracks who has installed a trial package, who is using it, how many licenses have been purchased, when it's up for renewal, etc. These are all important parts of the sales, billing, and marketing process ongoing in your production organization.

If you haven't already requested a CRM for ISV organization through the [Partner Community](#), do that now. CRM for ISVs is available to eligible partners. For more information on the Partner Program, including eligibility requirements, please visit us at www.salesforce.com/partners.



Note: If you have a Developer Edition organization that is being used to create a managed package, do *not* use this organization as your LMO.



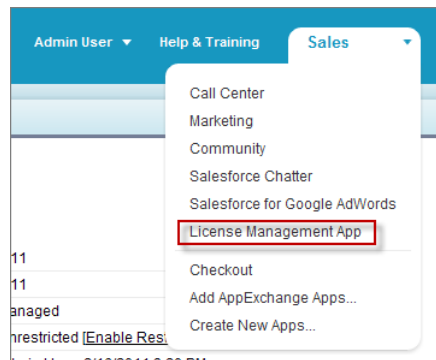
Warning: Once you begin to manage a package license out of a specific LMO, you can't move those licenses, leads, and other information to another organization. Once you install the LMA, you cannot uninstall it. You must contact Salesforce Support if you want to uninstall the LMA.

The LMA comes pre-installed with your ISV business organization. To install the LMA in a different organization, follow these steps.

1. File a case in the Partner Community under the **AppExchange and Feature Requests > License Management App** category.
2. Once the case is resolved, you will receive an email with an installation URL. Log in to the organization you plan to use as your LMO and click the LMA link in the email.
3. You can read the blurb about the app, then click **Get It Now**.
4. In the pop-up, choose **In my production Salesforce**.
5. Read and agree to the terms and conditions and then click **Install**.
6. Verify your password and then click **Submit**.
7. On the Package Installation Details page, click **Continue**.
8. Click **Next**, **Next**, and **Install**.


Click in the upper right corner and confirm you have the LMA installed.

License Management App



Configuring the LMA

After installing the LMA, configure it as follows.

1. [Assign a lead manager](#). It is critical that a user in your organization is the owner of all lead records created as a result of package installations.
2. Set the custom object permissions.
 - a. Licenses: most users in your organization do not need any permissions. Users who view licenses need the “Read” permission and users who modify license records need “Read” and “Edit” permissions.
 - b. Packages: only users who assign the lead manager need “Edit” permissions. Other users have either “Read” permissions or no permissions.
 - c. Package Versions: all users have “Read” permissions or no permissions, as there is no need to create, modify, or delete these records.
-  **Warning:**
 - The licenses, packages, and package versions managed in the LMA are different from most records in that they are created automatically by AppExchange. For this reason, license, package, and package version records should not be modified or deleted.
 - Users with the System Administrator profile can create, modify, and delete records of all of these custom objects, as they have the “Modify All Data” permission.
3. Set field-level security in user profiles or permission sets.
 - a. Licenses: settings depend on how you want to manage these fields for different users in your organization.
 - b. Packages: make all fields `Read-Only`. If you want users to edit the `Lead Manager` field, you can make this field editable.
 - c. Package Versions: make all fields `Read-Only`.
4. Override the standard **Edit** button on the license record to use the **Modify License** Visualforce page.
5. Add related lists:
 - Add the Licenses related list to the appropriate Lead page layouts. License managers can use this list on the lead detail page to view the licenses associated with a particular lead.
 - Add the Licenses related list to the appropriate Account page layouts. Users can view this list and identify the licenses associated with a particular account.

USER PERMISSIONS

To configure the LMA:

- System Administrator profile

To edit licenses and packages:

- “Read”
- AND
- “Edit”

To view licenses, packages, and package versions:

- “Read”

- Add the Licenses related list to the appropriate Contact page layouts. Users can view this list and identify the licenses associated with a particular contact.

Associating a Package with Your LMO

To receive lead and license records from customer installs, you must associate your managed package with your [License Management Organization \(LMO\)](#), the Salesforce organization where the License Management App is installed.



Warning: Once you associate a package with an LMO, that package's leads and licenses must permanently be managed out of the LMO. You can't migrate licenses.

You can associate your managed package with your LMO on AppExchange.

1. Go to <http://www.appexchange.com>
2. Click the **Publishing** tab.
3. Enter the Developer Edition organization where you created the package.
4. Agree to the terms and conditions.
5. Fill in a brief profile; a name will suffice.
6. Click **Your Uploaded Packages** tab and find your package version.
7. Click the **Manage Licenses** link for the package version.
8. Click **Register**.
9. Specify the LMO by providing user credentials for that organization and specify the other default license parameters.
10. Click **Save**. It may take 30 minutes or longer for a package record to appear as associated with the LMO.



Note: All versions of a managed package are associated with the same LMO, therefore, the LMO is associated only once with the package.

License Defaults

1. Log in to the Partner Community.
2. On the Publishing page, click the **Packages** tab.
3. Click **Manage Licenses** next to the package that you want to register.
4. Click **Register**. Enter the login credentials for the organization where the LMA is installed. Usually, the organization is your partner business organization.
5. Select whether your default license is Free Trial or Active.
6. If you selected a free-trial license, enter the length of the trial, up to 90 days.
7. Enter the number of seats associated with your default license, or select **License is site-wide** to offer the license to all users in the installer's organization.
8. Click **Save**.

When a customer installs the package, these default values are used to set the `Status`, `Expiration Date`, and `Seats` fields on the license record in the LMA and in the installer's organization. If you want to update these values, you can modify this license record in the LMA.



Note: Default values are at the package version level, therefore, different versions of the same package may have different default values.

Using the LMA

Use the License Management App (LMA) to apply licensing to your managed package to control how many users in the installer's organization can access your package and for how long. Every time your package is installed, a lead record is created in the LMA. You can use this information to contact the prospect and try to convert the lead to a customer, or to tell them about updates to your applications.

About Leads

When an installer installs a managed package, a lead is automatically created in the LMO. The lead contains the installer's name, company, and email address. The lead source defaults to `Package Installation`.

This information is important because it links the license to an actual Salesforce user that you can contact to notify about upgrades. If an installer uninstalls the package from his or her organization, a customer retention specialist for the developer's organization may want to contact the installer. Additionally, if there is an upgrade for a particular package, a license manager might want to mass email all installers with that package to let them know that an upgrade is available.

Each LMO should have a [lead manager](#) that is responsible for following up on all leads associated with licenses. When a new lead is created as a result of someone installing a package from AppExchange, the `Lead Owner` field on the lead record defaults to the [lead manager](#) specified for the package. If there is no lead manager specified for a package, the lead owner defaults to the `License Manager`.

EDITIONS

Available in:

- Enterprise
- Performance
- Unlimited
- Developer

USER PERMISSIONS

To edit licenses and packages:

- "Read"

AND

"Edit"

To view licenses, packages, and package versions:

- "Read"

About Packages

Packages in the LMA refer to managed packages that have been uploaded to AppExchange. On AppExchange, these packages are referred to as apps, regardless of what components they contain. Each package has one or more package versions, for which there can be multiple licenses. A Developer Edition organization can only contain one managed package at a time, but an LMO can manage multiple packages from multiple developers.

Click the Packages tab to display all the packages that a user has recently viewed. To view all packages that the LMO manages, select `All` from the list view. You can click a package name to view details including release date, latest version, owner, and developer name.

Package Details

From the Packages tab, click a package name to view package details. The package detail page contains information on the package, as well as relevant related lists. In the Package Version related list, you can see all the uploaded and registered package versions on AppExchange. The package version object has a master-detail relationship with package object.



Important: Do not edit, delete, clone, or create packages or package versions. Do not delete, clone, or create licenses. These records are automatically created and contain important information for tracking the licenses and packages in the License Management App. They cannot be repopulated.

Here are the attributes of a package.

Field	Description
Created By	Defaults to the License Manager user.
Developer Name	The package developer's organization name.
Developer Org ID	The 18-character ID of the package developer's organization.
Last Modified By	The name of the user that last modified this record, as well as the date and time the record was updated.
Latest Version	The most recent uploaded and registered version of the package. The developer enters this information when uploading the package.
Lead Manager	The LMO user that automatically becomes the owner of the leads created as a result of package installation. <code>Lead Manager</code> is blank when the package record is created. If you don't assign a lead manager, the License Manager user becomes the owner of the leads.
Owner	Defaults to the license manager user. You can transfer ownership at any time to any user in the organization by clicking Change next to the current owner's name.
Package ID	The 18-character GUID (Globally Unique ID) that identifies the package.
Package Name	The name of the package, as specified by the developer.
Release Date	The date the developer uploaded this package to AppExchange.

Editing Package Owners

The package owner automatically defaults to the [License Manager](#) user. You can change the owner of a package at any time, provided you have the appropriate permissions.



Note: You can assign ownership to any user in your organization. Therefore, it is very important to ensure that the user to which you are transferring ownership has access to the LMA and the custom app, and has the appropriate user permissions.

To change the owner of a single package:

1. Click the Packages tab.
2. Select a package.
3. Click the **Change** link next to the owner.
4. Click the lookup icon and choose a new owner.
5. Optionally, check the **Send Notification Email** checkbox to email the current and new owners.
6. Click **Save**.

You can also transfer ownership for multiple packages to a single owner by viewing a specific package list view. To change the owner of more than one license:

1. Click the Packages tab.
2. Select a package list view.
3. Select the packages you want to change.
4. Click **Change Owner**.

USER PERMISSIONS

To edit licenses and packages:

- "Read"
- AND
- "Edit"

- 5. Click the lookup icon and choose a new owner.
- 6. Optionally, check the **Send Notification Email** checkbox to email the current and new owners.
- 7. Click **Save**.

Editing the Lead Manager for a Package

When a new lead is created as a result of someone installing a package from AppExchange, the `Lead Owner` field on the lead record defaults to the `lead manager` specified for the package. If there is no lead manager specified for a package, the lead owner defaults to the `License Manager`.

To assign or change the lead manager for a package.

- 1. Select a package from the Packages tab.
- 2. Click **Edit**.
- 3. Click the lookup icon next to the `Lead Manager` field to search for a user.
- 4. Select a user.
- 5. Click **Save**.

USER PERMISSIONS

To edit licenses and packages:

- "Read"


AND

"Edit"

About Package Versions


Package versions are specific uploads of a package. A package version may contain updated or additional objects, representing an upgrade of a package. A package may also have a new version if the package was originally uploaded as "Managed - Beta" but has been changed to "Managed - Released."

Licenses represent the package version a user has installed. All versions of a managed package are [associated with the same LMO](#). If you view the detail for a package version, you see a list of all licenses for installers that have installed that version.

 **Note:** Do not edit, delete, clone, or create packages or package versions. Do not delete, clone, or create licenses. These records are automatically created and contain important information for tracking the licenses and packages in the License Management App. They cannot be repopulated.

Package Version Details

You can access the package version detail page by clicking the package version name in the Licenses, Packages, or Package Versions tabs. In the Licenses related list, you can see all the licenses associated with this package version. The license object has a lookup relationship with package version object.

 **Note:** Do not edit, delete, clone, or create packages or package versions. Do not delete, clone, or create licenses. These records are automatically created and contain important information for tracking the licenses and packages in the License Management App. They cannot be repopulated.

Here are the attributes of a package version.

Field	Description
Beta	An early version of a managed package that is uploaded to AppExchange for a sampling of your intended audience to test it.
Created By	Defaults to the License Manager user.

Field	Description
Last Modified By	The name of the user that last modified this record, as well as the date and time the record was updated.
Package	The package for which this is a package version.
Package Version Name	The name of the package version, as specified by the developer during upload.
Release Date	The date this package version was uploaded to AppExchange by the developer.
Version	The version, as specified by the developer during upload to Force.com AppExchange.
Version ID	The 18-character ID of this package version.

About Licenses

When an app is installed, Salesforce creates a license entry in the LMA. This license is directly related to the version of the package that the installer has installed. Click the Licenses tab to display all licenses that a user has recently viewed. To view all licenses for the packages that the LMO manages, select **All** from the drop-down list and click **Go!**



Note: Do not edit, delete, clone, or create packages or package versions. Do not delete, clone, or create licenses. These records are automatically created and contain important information for tracking the licenses and packages in the License Management App. They cannot be repopulated.

License Details


From the License tab, you can click a license name to view details including status, package version, owner, and install date. Here are the attributes of a license.

Field	Description
Account	The account for a converted lead.
Contact	The contact for a converted lead.
Created By	Defaults to the License Manager user.
Expiration Date	Displays the expiration date or Does not expire if the license does not expire. The default value is Does not expire .
Information Current As Of	The last time Salesforce retrieved information about the installer's organization.
Install Date	The date the package version was initially installed by the installer.
Instance	The Salesforce instance where the installer's organization resides.
Last Modified By	The name of the user that last modified this record, as well as the date and time the record was updated.

Field	Description
Lead	<p>The lead that the LMA automatically created when the app was installed. This lead represents the user who owns the license.</p> <p>If the lead is converted into an opportunity, the lead name is retained but the lead record no longer exists. Therefore, if you click this link, a page displays indicating that the lead has been converted.</p>
License Name	Represents an instance of a license. The license name is an auto-number that increments by one for each new license.
Licensed Seats	A formula field that displays the number of licenses or <code>Site License</code> . The default value is <code>Site License</code> .
License Status	Indicates the type of license. Available values are <code>Trial</code> , <code>Active</code> , <code>Suspended</code> , and <code>Uninstalled</code> .
License Type	Indicates whether the license is editable.
Org Edition	The edition of the organization where the package is installed.
Org Expiration Date	If the installer is using a trial organization, the date when the trial expires.
Org Status	The status of the installer's organization. Possible values include <code>Trial</code> or <code>Active</code> .
Owner	Defaults to the license manager user. You can transfer ownership at any time to any user in the organization by clicking Change next to the current owner's name.
Package Version	Links to the detail page for the package version that is the parent of this license.
Package Version Number	The version number of the installed package.
Sandbox	If the license is for a package installed in a sandbox organization.
Subscriber Org ID	A globally unique 15-character ID representing the installer's organization.
Used Licenses	<p>Displays the number of users in an organization who have a license to a package. (Read only)</p> <p>This field is blank if:</p> <ul style="list-style-type: none"> • A customer uninstalled the package • <code>Licensed Seats</code> is set to <code>Site License</code>. This is because all users in an organization implicitly have a license to the package when <code>Licensed Seats</code> is set to <code>Site License</code>.

Editing License Owners

The license owner automatically defaults to the [License Manager](#) user. You can change the owner, provided you have the appropriate permissions. You can also transfer ownership for multiple licenses to a single owner by navigating to a specific license list view.

 **Note:** You can assign ownership to any user in your organization. Therefore, it is very important to ensure that the user to which you are transferring ownership has access to the LMA and the custom app, and has the appropriate user permissions.

To change the owner of a license:

1. Click the Licenses tab.
2. Select a license.
3. Click the **Change** link next to the owner.
4. Click the lookup icon and choose a new owner.
5. Optionally, check the **Send Notification Email** checkbox to email the current and new owners.
6. Click **Save**.


To change the owner of more than one license:

1. Click the Licenses tab.
2. Select a license list view.
3. Select the licenses you want to change.
4. Click **Change Owner**.
5. Click the lookup icon and choose a new owner.
6. Optionally, check the **Send Notification Email** checkbox to email the current and new owners.
7. Click **Save**.

Editing Licenses

You can modify licenses for each installer. To modify a license:

1. Click the Licenses tab.
2. Select the appropriate license.
3. Click **Modify License**.

 **Note:** If **Modify License** is not visible, check your page layout or contact your administrator.

4. Make the necessary changes to [editable fields](#).
5. Click **Save**.

Editable License Fields

You can modify the following License fields.

USER PERMISSIONS


To edit licenses and packages:

- "Read"
- AND
- "Edit"

USER PERMISSIONS

To edit licenses and packages:

- "Read"
- AND
- "Edit"

Field	Description
Expiration	Enter the expiration date to identify the last day the installer can access the package under the license agreement, or check Does not expire if the license does not expire.
Seats	Enter a positive number to set the number of licenses or check Site License to make the package available to all users in the installer's organization. The value defaults to Site License.
Status	<p>Use the drop-down list to set the license status. Available values are:</p> <ul style="list-style-type: none"> • Trial: Set to Trial to allow the installer to try out the package for up to 90 days. Once a trial license is converted to an active license, it cannot return to a trial license. • Active: Set to Active to allow the installer to use your package according to your license agreement. • Suspended: Set to Suspended to prohibit the installer from accessing the package. You may want to set a package to Suspended if a user fails to pay for the license. <p> Note: When the installer uninstalls the package, the status is set to Uninstalled. The license manager cannot set or change this status. Once the package is uninstalled the license becomes read only and is no longer editable. The allowed status changes are:</p> <ul style="list-style-type: none"> • Trial to Active • Active to Suspended • Suspended to Active

Integrating with Sales and Marketing

You can use the information in the LMA to send notifications to your customers.



Warning: Do not create workflow rules, triggers, or validation rules that required custom fields on the license or lead objects. Do not impose any conditions on updating or creating license or lead records. Doing so will block creation of the license or lead records by the LMA, resulting in the loss of data concerning the packages installed by your customers.

Send upgrade notifications

Create an email template for notifying customers that an upgrade is available.

Send license expiration notifications

Create a workflow rule that automatically emails the sales representative or account manager before the license expires. To do this, create an email template for the notification. Then, create a workflow rule with a filter that specifies a time period (month) before the

`Expiration` date. Finally, associate the workflow rule with a workflow alert that sends an email to the appropriate team member or sales representative. You should configure the rule so that your sales representative and the customer have sufficient time to explore contract renewal options before the license expires. It is also a good idea to send emails to customers warning them of license expiration.

Retain customers

Create a workflow rule that automatically emails a customer service representative or customer retention specialist when the package is no longer installed in that organization. To do this, create an email template for the notification. Then, create a workflow rule with a filter that specifies that the `License Status` equals "Uninstalled". Finally, associate the workflow rule with a workflow alert that sends an email to the appropriate team member or customer retention specialist.

Use the API to find licensed users

You can use the `isCurrentUserLicensed` method to determine if a user has a license to a managed package. For more information, see the *Apex Developer's Guide*.

Best Practices

Avoid mandatory fields

Do not create mandatory custom fields on lead, license, package and package version objects.

Avoid `before-create` triggers and validation rules

Don't define `before-create` triggers or validation rules on lead, license, package, or package version objects.

Use Custom Domains

When using Package Support Access to troubleshoot customer setup issues, you may be automatically logged out of your LMO. To avoid being logged out, use a custom domain for your LMO. For more information, see "My Domain Overview" in the Salesforce online help.

Make sure the lead user is a valid, active user

This is necessary for lead and license creation.

Track licenses

Create a lead list view filter for leads created by installed packages.

Set up history tracking for license fields.

Send license expiration notifications

Create a workflow rule that automatically emails the sales representative or account manager before the license expires. To do this, create an email template for the notification. Then, create a workflow rule with a filter that specifies a time period (for example, month) before the `Expiration` date. Finally, associate the workflow rule with a workflow alert that sends an email to the appropriate team member or sales representative. You should configure the rule so that your sales representative and the customer have sufficient time to explore contract renewal options before the license expires. It is also a good idea to send emails to customers warning them of license expiration.

Send upgrade notifications

Create an email template for notifying customers that an upgrade is available. The Leads tab in the LMA has a list of all customers.

Use the API to find licensed users

You can use the `isCurrentUserLicensed` method to determine if a user has a license to a managed package. For more information, see the [Force.com Apex Code Developer's Guide](#).

Retain customers

If a customer uninstalls a package, the status of the license in the LMO changes to Uninstalled. Create a workflow rule that automatically emails a customer service representative when the package is no longer installed in that organization. To do this, create an email template for the notification. Then, create a workflow rule with a filter that specifies that the `License Status` equals "Uninstalled". Finally, associate the workflow rule with a workflow alert that sends an email to the appropriate team member or customer retention specialist.



Warning: Do not create workflow rules, triggers, or validation rules that require custom fields on the license or lead objects. Do not impose any conditions on updating or creating license or lead records. Doing so will block creation of the license or lead records by the LMA, resulting in the loss of data concerning the packages installed by your customers.

License Management App FAQ

How do I change my License Management Organization?

Contact Salesforce support. License records in the current LMO don't move to the new LMO. They stay with the original LMO.

Why aren't Lead and License records being created in my LMO?

- Your LMO hasn't been associated with your package.
- Your package version hasn't been registered.
- There are mandatory custom fields in lead, license, package, and package version custom objects.
- The LMO Lead Manager (the lead records owner) is not a valid active user.
- The `before_` triggers are preventing lead creation.

For more possibilities, see [Troubleshooting](#).

Why can't I see the Modify License button?

Customize the page layout by adding the **Modify License** button to your page.

A customer installed my package before I associated it with my LMO. How can I manage the license record?

Ask the customer to install your package again without uninstalling it.

What happens when I decrease the number of licenses below the current number of licensed users?

All users still have access until a system administrator in the installer's organization revokes the extra licenses.

Why should I install the LMA into my production Salesforce organization?

The LMA is an important part of a partner's sales, billing, and marketing of a package on the Salesforce Platform. Installing it in a separate organization means they can't manage package licenses as a part of their sales and marketing process.

Installing the LMA into the customer's production organization means they can manage the lead, trial, licensing, and billing as part of their overall Salesforce dataset.

Can I automate the assignment of package licenses to users in the subscriber organization?

Yes, you can use the API to assign or revoke licenses for managed packages installed in your organization. For more information, see the [PackageLicense](#) and [UserPackageLicense](#) objects in the *SOAP API Developer's Guide*.

Troubleshooting

The most frequent troubles arise from leads and licenses not being created or a proxy user being deactivated.

Leads and licenses are not being created

When a subscriber installs my package, leads or licenses are not being created in the LMO. Usually this is caused by a misconfiguration of the organization where the LMA is installed. Here is a list of things to check:

Did the subscriber really install the package?

AppExchange counts the installation when the user clicks "Get it now", but the installation may fail or be canceled and the license won't be delivered unless the installation was completed.

Is State and Country picklist validation enabled?

If so, try disabling it. There is a known issue that prevents leads from being created in the LMA if this feature is enabled. The issue occurs if subscribers haven't provided State and Country values in their user profiles, or those values are incorrect.

Is there a trigger on the lead or license object in the LMO?

There should never be a `before_create` or `before_update` trigger on a lead or license in the LMO. Try using `after_` triggers instead. Or try removing the trigger; if the trigger fails for some reason, it can block license creation.

Is there a required custom field on the lead or license record?

If so remove the requirement; the LMA doesn't populate that required field, so it can block license or lead creation.

Is the LMO Lead Manager user a valid, active user?

The Lead Manager (the owner of lead records) must be a valid, active user, if not, it will block license or lead creation.

Is there a validation rule on the lead or license record?

Often these will block creation of the LMA lead or license records because that required field is not there.

Is there a workflow rule on lead or licenses?

These might be preventing creation of the license. Try removing the workflow rule.

Was the lead converted to an account?

When leads are converted to accounts, they are no longer leads.



Note: Licenses that were not delivered to an LMO will usually be re-delivered after a few days if the issue is solved in the above steps.

A proxy user has deactivated

If a "proxy user has deactivated" message appears when editing a license in the LMO, a subscriber organization may be locked, deleted, or otherwise disabled. Check the following:

Verify the organization is active

Check with the organization administrator to see if the organization is active or deleted. If the organization is deleted, delete the corresponding license record.

Verify the package is installed

Check with the organization administrator to see if the package is still installed. If the organization is locked or the package no longer installed, the license may not be updated. Have the subscriber reinstall the package.

CHAPTER 9 Provide a Free Trial

In this chapter ...

- [Why Use Trialforce?](#)
- [Trialforce](#)
- [Set Up Trialforce](#)
- [Provide a Free Trial on the AppExchange](#)
- [Provide a Free Trial on Your Website](#)
- [Modify the Trial for an Upgrade](#)
- [Trialforce Best Practices](#)
- [Creating Signups using the API](#)
- [Trialforce FAQ](#)

Free trials help you to reach a wider range of customers and maximize the adoption of your offering. It's not uncommon for partners to more than double their leads after enabling free trials. You have several options for providing a free trial.

- From the AppExchange, by letting prospects install your app or component
- From the AppExchange, by configuring a test drive
- From the AppExchange, using Trialforce
- From your website, using Trialforce

Trialforce is a provisioning technology for Salesforce organizations that lets partners set up and manage free trials of apps and components. Use Trialforce to configure a trial to your specifications, include relevant sample data, and even customize the look and feel to reflect your company's branding.



Note: This feature is available to eligible partners. For more information on the Partner Program, including eligibility requirements, visit www.salesforce.com/partners.

Why Use Trialforce?

Trialforce lets you provision a free trial of your offering quickly and easily. Each time a trial is provisioned, Trialforce creates a lead in the License Management App, which helps you track usage and convert prospects into paying customers. With Trialforce, you can:

- Run your own marketing campaign to maximize customer reach and adoption.
- Customize your offering, including branding, functionality, design, data, and trial experience.
- Manage trials for multiple offerings, versions, and editions from one convenient place.
- Let customers, including non-admin users, try your app or component without logging in to their production environment.

Trialforce

A Trialforce setup consists of several parts. It's important to understand these parts and the relationship between them before you start using Trialforce.

Trialforce Management Organization (TMO)

The TMO is the starting point for setting up Trialforce and the central location for managing Trialforce after it's set up. You must file a case in the [Partner Community](#) to receive your TMO. The two tasks you perform in the TMO are create Trialforce source organizations and define templates for custom branding.

Trialforce Source Organization (TSO)

You use the TSO to create a template for the trial orgs received by your customers. You create the TSO from your TMO. The tasks you perform in a TSO are: install your offering, along with any sample data; specify branding by choosing from the templates you created previously in the TMO; configure the TSO to be exactly as you want your customers to experience it; and generate a Trialforce template, which becomes the basis for all trial organizations.

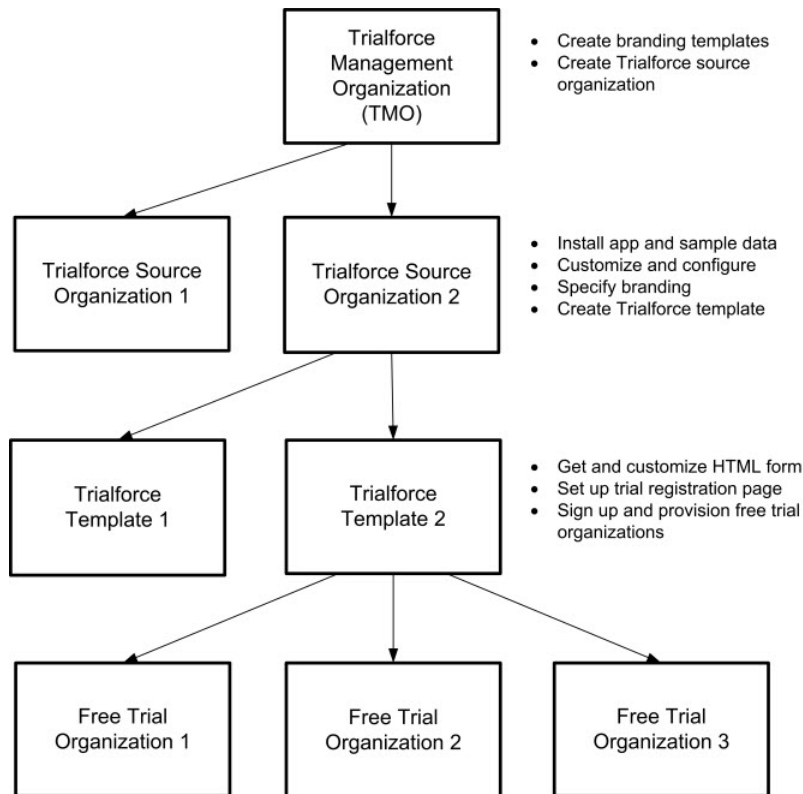
Trialforce Template

The template is a snapshot or exact copy of your TSO at a specific instance in time. You create it from a TSO after you've installed your offering and made configuration changes. The Trialforce template is specified in the HTML page from which customers sign up for trials. It defines the trial organization that is provisioned each time a customer signs up.

HTML Signup Form

This HTML form serves as the registration page on your website from which customers sign up for trials. You must file a case in the Partner Community to get this form and then customize it with your company details. It is associated with the Trialforce template you plan to use for trials. This ensures that each time a customer signs up for a trial on your website, they receive an organization that is an exact copy of your chosen Trialforce template.

Relationship between organizations used to set up Trialforce



The TMO, TSOs, and Trialforce templates have a hierarchical relationship, as illustrated above.

- You can create multiple TSOs from a given TMO. For example, if you want to offer trials for two different apps, you would generate two different TSOs from the same TMO, one for each app. This enables you to use the TMO as a central hub to manage the trials for all Force.com apps or components produced by your company.
- You can create multiple Trialforce templates from the same TSO. For example, if you release a new version of your component after you've started using Trialforce, you can install the upgraded version into the previous TSO and generate a new Trialforce template from it. If you then update your HTML signup form to point to the new Trialforce template, all trial organizations created subsequently have the new version of the package.

As a best practice, we recommend that you have one unique TMO for your company, one TSO for each app or component, and one Trialforce template for each version or edition. Splitting up the configuration process across these different levels makes it easier to maintain and update your trials. Then each time you change something, such as the version, its branding, or a configuration detail of the trial organization, you only need to make the change at one level in the hierarchy. This minimizes the configuration steps involved and makes it easy to concurrently manage trials for multiple products, versions, and editions.

After you've configured a TMO, TSO, and Trialforce template, choose how to provide trials to prospective customers:

- Using the AppExchange**— Customers begin a trial of your offering directly from an AppExchange listing. This approach is ideal if you're looking for the quickest, easiest way to make a trial available because it requires only a few steps to configure.
- Using an HTML signup form**— Customers begin a trial of your offering after filling in a customizable HTML signup form. Because you can modify the form's look and feel to match your own website, this approach is ideal for integrating the signup process into your company's web presence.
- Using the API**— You provision a trial of your app or component programmatically using the SignupRequest API. This approach is ideal if you're looking to have full control of the signup process because it allows for advanced customization.

Set Up Trialforce

After you've built your offering and passed the AppExchange security review, follow these steps to set up Trialforce.

 **Note:** To enable Trialforce, you must first sign the ISVforce/OEM agreement.

1. Create your managed package.
2. Configure a License Management Organization (LMO) to manage customers' access to apps and components. If you're an existing Salesforce user, install the License Management Application (LMA) in your CRM organization (Enterprise Edition is required). If you're new to the Partner Program, the LMA is preinstalled in your partner business org.
3. [Link a version with the LMO and set the license defaults](#). This step ensures that each time a prospect creates a trial, the LMO receives a new lead and license record.
4. [Request a Trialforce Management Organization \(TMO\)](#).
5. Optionally, create a customized [branded login page](#) and [branded emails](#) in your TMO.
6. [Create a Trialforce Source Organization \(TSO\) from your TMO](#).
7. Install your managed package in the TSO, and customize it as you want your prospects to experience it. You can apply custom branding, load sample data, create custom profiles, and so on.
8. [Create a new Trialforce template from the TSO](#).
9. [Link the Trialforce template to the AppExchange](#).
10. [Submit the Trialforce template for security review](#) and get it approved.

You can now use this template to create free trials. For more information, see:

- [Providing a free trial on AppExchange](#)
- [Providing a free trial on your website](#)
- [Providing a free trial using the API](#)

EDITIONS

Available in:

- Developer

USER PERMISSIONS

To manage Trialforce:

- "Customize Application"

Link a Package with Your License Management Organization

To receive lead and license records from customer installs, link a managed package to your License Management Organization (LMO), the organization where the License Management App (LMA) is installed. You also specify default license settings for your offering during this process. Default license values are used to set the Status, Expiration Date, and Seats fields on the license record in the LMA and in the installer's organization.

 **Note:** When you link a package with an LMO, that package's leads and licenses must be permanently managed out of the LMO. You can't migrate licenses to another organization.

1. Log in to the Partner Community.
2. On the Publishing page, click the **Packages** tab.
3. Find the package that you want to link, and click **Manage Licenses**.
4. Click **Register**.
5. Enter the login credentials for your LMO, and click **Submit**.
6. Select whether your default license is a free trial or active.
7. Enter the license length in number of days. If your license is free or doesn't expire, select **License does not expire**.

8. Enter the number of seats associated with your default license, or select **License is site-wide** to offer the license to all users in the installer's organization.
9. Click **Save**.

To verify that you linked the package successfully, log in to the LMO and click the **Package Versions** tab. After you link a package to your LMO, all versions of that package are associated.

Request a Trialforce Management Organization

Trials are managed in a Trialforce Management Organization (TMO). The TMO enables you to create a Trialforce Source Organization (TSO) and specify custom branding for your login page and emails. The TMO is a separate organization from your business organization or the Developer Edition organization in which you built your app or component. To receive a TMO, you must be a qualified ISV partner, and your offering must have passed the AppExchange security review.

1. Log in to the Partner Community.
2. On the Support tab of the Partner Community, click **New Case**.
3. For category, choose **AppExchange and Feature Requests > Trialforce**.
4. Complete the form to create a ticket requesting a TMO. In the body of the ticket, provide the requested information. Include details about your package and the organization ID of the Developer Edition org that you want to use as your TMO.



Note: To find your organization ID, go to **Company Profile > Company Information** in the Setup menu, and see the Organization Details section.

5. Click **Save**.

A ticket is generated and sent to the partner operations team. The team reviews the case and configures your Developer Edition org to be the TMO or contacts you if they have questions.

EDITIONS

Available in:

- Developer

USER PERMISSIONS

To manage Trialforce:

- "Customize Application"

Setting up Custom Branding for Trialforce

App developers using Trialforce to create new trials of their product can optionally set up a branded login site and system emails. By branding these areas with your company's look and feel, users of your application will be immersed in your brand from sign-up to login. Custom branding should only be used for non-CRM apps, not for apps that extend Salesforce CRM and require Salesforce standard objects such as Leads, Opportunities, and Cases.

A branded login page enables you to specify your login domain and login site.

- Login domains end with `.cloudforce.com`, so that if your company name is "mycompany," then your login domain will be `mycompany.cloudforce.com`.
- Your custom login site includes your text and company logo, and mobile-friendly versions of your login site as well.

Branded emails allow you to specify fields in system-generated emails so that your company name, address, and other pertinent details are used in email correspondence. You can create multiple branded email sets for different campaigns or customer segments.



Note: To configure branding, you must be logged in to a Trialforce Management Organization (TMO). To get your TMO, log a case in the [Partner Community](#).

EDITIONS

Available in:

- Developer

USER PERMISSIONS

To manage Trialforce:

- "Customize Application"

Creating Branded Emails

You can customize the branding of the emails sent to subscribers of new trial organizations.

To create a branded email set:

1. Log in to your Trialforce Management Organization.
2. From Setup, click **App Setup > Trialforce > Branding > Email Sets**.
3. Click **New Email Set** or **Edit** next to an existing email set.
4. Enter a name for the email set and your company information.
5. In the Preview Emails area, click through the different types of generated emails and make sure they read correctly.



Note: The login URL displayed in the preview will always be `http://login.salesforce.com` even if you use a branded login page. These two processes are distinct.

6. Click **Save**.
7. If you're ready to make these emails available to your Trialforce Source Organization (TSO), click **Publish**. Otherwise your changes are saved and you can publish later.

To assign a branded email set to your TSO:

1. From Setup, click **App Setup > Trialforce > Source Organizations**.
2. Click **Edit** next to your TSO.
3. Select the email set.
4. Click **Save**.
5. Click **Login** if you want to see your branded login page in action.

Creating a Branded Login Page

Customers typically log in to your app using the traditional `login.salesforce.com` site. A branded login page enables you to customize this domain and parts of this login page so you can provide a branded experience for your customers. Your custom login site includes your text and company logo, and mobile-friendly versions of your login site as well.

To create a branded login page:

1. Log in to your Trialforce Management Organization.
2. From Setup, click **Trialforce > Branding > Login Site**.
3. Click **Set Up Login Site**.
4. Select a subdomain for your login site by providing a name in the field provided. Usually this is the name of your company.



Note: Login domains end with `.cloudforce.com`, so that if your company name is "mycompany," then your login domain will be `mycompany.cloudforce.com`.

5. Check the availability of the domain and then accept the terms of use.
6. Click **Save and Launch Editor**.
7. Use the Login Brand Editor to change how your login page looks. For additional help using the editor, click **Help for this Page**.
8. Click **Save and Close**.

EDITIONS

Available in:

- Developer

USER PERMISSIONS

To manage Trialforce:

- "Customize Application"

EDITIONS

Available in:

- Developer

USER PERMISSIONS

To manage Trialforce:

- "Customize Application"

9. If you're ready to make these changes available to your TSO, click **Publish**. Otherwise your changes are saved and you can publish later.

Create a Trialforce Source Organization

A Trialforce Source Organization (TSO) acts as the basis for a new trial organization. After you create a TSO, you install your package there. You then add data to give your prospects something to explore when they first log in to the trial organization.


You have two options for creating a TSO: You can use a Trialforce Management Organization (TMO) or the Environment Hub. If you plan to brand your emails or login page, use a TMO. Otherwise, use the Environment Hub.

1. Log in to your TMO.
2. From Setup, click **Trialforce > Source Organizations**.
3. Click **New**.
4. Enter a new username and email address for the administrator account.
5. Enter a name for the TSO. Optionally, specify the custom branding by choosing a branded email set or login site.
6. Click **Create**.

You receive an email with the login details for your TSO. You can then log in to the TSO and install your package, along with sample data and configurations. Optionally, you can also create:

- Custom profiles
- New users
- Sample records

The goal is to configure the TSO exactly as you want your customers to experience it. You can then create a Trialforce template, which is a snapshot or exact copy of your TSO at a specific point in time.

 **Note:** Here are some considerations when working with a TSO.

- Always associate a managed package with the License Management Organization (LMO) before installing the offering in your TSO. If you don't follow that order, trial organizations provisioned from the TSO don't generate leads or licenses in the LMO.
- Before creating a Trialforce template, ensure that the TSO admin has a license for the offering installed in the TSO.
- You can create multiple TSOs from your TMO. This allows you to set up trials for different products, each with its own configuration and branding.
- All new TSOs expire after one year. Log a case to request an extension if you plan to use the TSO for a longer period.

EDITIONS

Available in:

- Developer

USER PERMISSIONS

To manage Trialforce:

- "Customize Application"

Create a Trialforce Template

A Trialforce template is a snapshot or exact copy of your Trialforce Source Organization (TSO) at a given instance in time. Before you create the template, make sure that you've installed your package into the TSO. Then, configure it exactly as you want your customers to experience it, with the appropriate sample data, profiles, users, and records.

 **Note:** You can create a Trialforce template only if your TSO is less than 256 MB.

1. Log in to your TSO.
2. From Setup, click **App Setup > Trialforce**.
3. Click **New Trialforce Template**.
4. Describe the template and whether to include data. The default option is fine for most cases.
5. Click **Save**.

You receive an email with the organization ID of the new template after it's generated. You must submit the template for review before you can use it to sign up trial organizations. Remember to generate a new template each time you make updates to your TSO so that your trials always reflect the most recent state.

Each Trialforce template has a status with one of the following values.

In Progress

When a Trialforce template is first created, it always has this status. It then moves to either Success or Error status.

Success

The Trialforce template can be used to create trial organizations.

Error

The Trialforce template cannot be used because something has gone wrong and debugging is required.

Deleted

The Trialforce template is no longer available for use. Deleted templates are removed during system updates.

EDITIONS

Available in:

- Developer

USER PERMISSIONS

To manage Trialforce:

- "Customize Application"

Link a Trialforce Template to the AppExchange

To offer a free trial with your app or component listing, link a Trialforce template to the AppExchange.

1. Log in to the Partner Community.
2. On the Publishing page, click the **Organizations** tab.
3. Click **Connect Organization**.
4. Enter the login credentials for the organization that contains the trial template. If you developed multiple trial templates in this organization, they are all linked to the AppExchange.
5. Click **Submit**.
6. Optionally, click the **Trial Templates** tab to view the linked template and create a listing.

Submit a Trialforce Template for Security Review

To offer a trial on the AppExchange using Trialforce, your template must pass a security review. Before requesting a review, link the organization containing your Trialforce template to the AppExchange.

 **Note:** You can only request a review on a Trialforce template that has at least one package installed. You must own or license the installed packages in the template and it should have already passed the security review.

1. Log in to the Partner Community.
2. On the Publishing page, click the **Trial Templates** tab.
3. Next to the template that you want reviewed, click **Start Review**.

You receive an email confirmation after you initiate the review and another email when the review is completed. The review is free for partners and typically takes 2–3 days.

Provide a Free Trial on the AppExchange

To create trials on the AppExchange, your app or component must:


- Be a managed package
- Be managed via the License Management Application
- Autoprovision—that is, the user must not need to interact with you at any point to get the app or component up and running
- Have passed the security review
- Have passed the Trialforce template review

You can provide a free trial on the AppExchange in three ways.

- [Using Trialforce](#)
- [By configuring a test drive](#)
- [By installing your app or component into an existing organization](#)

Provide a Free Trial on the AppExchange Using Trialforce

Providing a free trial lets potential customers experience your offering before purchasing or subscribing.

 **Note:** You must be an eligible partner to provide free trials. For more information on the Partner Program, including eligibility requirements, visit www.salesforce.com/partners.

1. Create a Trialforce template with your offering installed and configured as you want your prospects to experience it. For details, see [Setting up Trialforce](#).
2. Submit the Trialforce template for security review. This review is free and takes less time than the initial review of your app or component.
3. Link the Trialforce template to your AppExchange listing.
 - a. Log in to the Partner Community.
 - b. On the Publishing page, click the **Listings** tab.
 - c. Find the listing where you want to offer a trial, and click it to open the AppExchange publishing console.
 - d. Click the **Trials** tab, and select **Offer a free trial organization**.
 - e. Follow the on-screen prompts to add a trial template to the listing.
4. Click **Save**.

Now, when customers visit your listing, they can start a free trial with your offering preinstalled, even if they don't have a Salesforce account. If they decide to start a trial, we collect their contact information and ask them to agree to your terms and conditions and our MSA. After they provide this information, prospects receive an email prompting them to log in to a trial organization.

Provide a Test Drive on the AppExchange

A test drive is a preconfigured Developer Edition (DE) org that lets prospects try your offering with sample data that you provide. The test drive has two users: an administrator and a read-only evaluation user. The administrator role is used to configure the organization for the test drive. The evaluation role is used when prospects log in to the organization to experience your app or component.

1. Enable test drives on your AppExchange listing.
 - a. Log in to the Partner Community.
 - b. On the Publishing page, click the **Listings** tab.
 - c. Find the listing where you want to offer a test drive, and click it to open the AppExchange publishing console.
 - d. Click the **Trials** tab, and select **Offer a Test Drive**.
2. Create the test drive organization.
 - a. Click **Create Test Drive**.
 - b. Give the test drive a customer-friendly name, and associate the package containing your app or component.
 - c. Click **Submit**. Salesforce creates a DE org and emails you login credentials for administrator and evaluation users.
3. Configure the test drive organization.
 - a. Log in as the administrator to add sample data and perform other setup tasks.
 - b. Log in as the evaluation user to define a password.
4. Connect the test drive organization to your AppExchange listing.
 - a. Return to the **Trials** tab in the publishing console.
 - b. Click **Connect Organization**.
 - c. Enter the login credentials for the evaluation user.
 - d. Click **Submit**.
5. Click **Save**.

Now, when prospects take a test drive from your listing, they are logged in as read-only evaluation users.

Provide a Free Trial on the AppExchange When Your Offering Is Installed

You can provide a free trial of your offering by setting the default license settings on your package. When a customer installs the app or component in an existing Salesforce organization, they can use it for the specified trial period.

Provide a Free Trial on Your Website

You can use an HTML form to drive traffic to your business and show prospective customers the products and services that you offer.

Before providing a free trial on your website, be sure you've followed the steps outlined in [Setting Up Trialforce](#).

1. [Set up Trialforce](#).
2. [Request an HTML registration form](#).
3. [Link your Trialforce template to the HTML form](#).
4. [Customize the HTML form](#).

5. Provision new trial organizations.

After you've completed these tasks, you're ready to go live. Now, each time prospective customers enter their information and submit your form, Salesforce provisions a trial based on your Trialforce template.



Note: As an alternative to using a web form, you can [create Trialforce sign-ups using the API](#). The API gives you more control over the sign-up process and enhanced visibility into your prospective customers.

Requesting a Signup Form for Trialforce

Partners wishing to use Trialforce need to link their website to an HTML form that collects important information about the individual requesting a trial. Salesforce provides an easy way to request a sample HTML form whose branding you can customize to match your site, while retaining the information required to ensure that submissions of the form will successfully provision the right trial.

To request a signup form:

1. Log in to the [Partner Community](#).
2. Under the Support tab, click **New Case**.
3. Log a case in the **AppExchange and Feature Requests > Trialforce** category.

You'll receive an email with the signup form, as well as instructions on how to modify it to ensure proper trial provisioning. Follow the instructions in the email provided to make the necessary changes to the default HTML form.



Note: You can request the signup form before the Trialforce template is ready. Since the form won't be linked to any particular template, the resulting organization will be a simple trial organization that expires in two days. But it provides you a good way to test the form without requiring a copy of the Trialforce template.

Link a Trialforce Template to the Sign-Up Form

Link a Trialforce template to your HTML sign-up form so that customers who request a trial receive an organization with your offering installed, along with the data that you added to the template.

If you skip this step, customers who fill out the form receive a generic trial organization that expires in two days.

1. Log in to your Trialforce Source Organization.
2. From Setup, click **App Setup > Trialforce**.
3. Note the template ID of the Trialforce template that you want to use. The ID has a value similar to 0TTi0000000Sxd8.
4. Note the form name value on your sign-up form.
5. Log a case to associate your Trialforce template with the sign-up form.
 - a. Log in to the [Partner Community](#).
 - b. On the Support tab, click **New Case**.
 - c. Select the **AppExchange and Feature Requests > Trialforce** category.
 - d. In the description, provide the ID of the Trialforce Source Organization, the ID of your Trialforce template, and the form name.

You'll receive an email confirming that your request has been processed. Test that the template has been properly associated to the registration form by filling out and submitting the form. You receive an email when the new trial has been provisioned.

Customizing the HTML Registration Form

The sample registration HTML form you received by logging a case on the Partner Community is just a template, and you'll need to modify it. There are some mandatory changes that must be made to ensure that the proper trial is provisioned. You can, optionally, modify the file to reflect the look and feel of your website and brand.

You'll need to assign the `formName` and `Lead.Partner_Account` values in the HTML form to those provided by the partner support team. These values will be given to you in the email that contains the sample HTML registration form.

1. Open your registration form HTML file in a text or HTML editing tool.
2. Change the following lines of HTML in the registration form to reflect this information by updating the value attribute. Search for the TODO comment, which will help you find the lines to change.

```
<!-- TODO: Add Signup Config Item Name of Config record as formName -->
<!-- TODO: Add Partner Account Record Id for Partner Lookup on Lead
(Provided by Salesforce.com-->
<input type="hidden" name="formName" value=""/>
<input type="hidden" name="Lead.Partner_Account__c" value=""/>
```

3. Search for and modify other sections in the form labeled "TODO." This will allow you to do things like specify company logos, override success or failure pages, and so on.
 - a. Open your registration form .HTML file in a text or html editing tool.
 - b. Search for the term "TODO".
 - c. Follow the instructions in the comments to change the appropriate portion of the form. Examples of items that can be changed in this way:
 - provide a custom logo
 - modify the display name of the application (do not use spaces)
 - specify your company name as the referring entity for any leads generated by the form
 - specify custom URLs for success and failure redirects
 - update the language/locale of the form
4. Modify the HTML and CSS to match the look and layout of your website



Note: Don't make changes to the JavaScript except in specifically identified sections to ensure that the form will provision a new trial properly.

Once the changes have been made, you can test filling out and submitting the form to verify that new trial organizations are provisioned correctly.

If you haven't already done so, log another case on the Partner Community to associate your trial template snapshot ID with your `FormName`. Otherwise, the trials provisioned by filling out the form will be generic Force.com two day trials.

Provisioning New Trial Organizations

Once you've configured Trialforce, you can provision new trial organizations in one of two ways.

- Push—You provision a trial on behalf of a customer by filling out the registration form with your prospect's information.
 - Pull—Prospects request a trial on their own by filling out a registration form on your public website.
1. Upload the HTML registration form to your public web servers.

2. Edit and publish the appropriate HTML pages on your company website where you want to include a link to the Trialforce registration form.
3. Navigate to the registration page from your company website.
4. Fill in the required fields and submit the form.

Anyone with access to the registration form can manually create a trial on behalf of a prospect without the need to expose the registration form on the company website. Simply launch the registration form HTML file in a browser and fill in the fields on behalf of the customer, then submit the form. Your prospect will receive an email, optionally branded with your company information, indicating the new trial is available.

Modify the Trial for an Upgrade

You can update your trials to reflect changes to your offering or its custom branding. To do so, you must:

- Create and publish a new version of your managed package (or an extension package).
 - Have a Trialforce Source Organization (TSO) where you can upload the new package version. You can reuse the TSO that you used to create your original Trialforce template, or you can create a new one. If you create a new TSO, be sure to link it to the AppExchange.
1. Install your updated managed package (or extension package) into your TSO.
 2. Make any other desired changes in the TSO, such as loading sample data or updating custom branding.
 3. Create a Trialforce template for your trial.
 4. [Submit the template for review.](#)
 5. For trials created using an HTML sign-up form or using the API, complete the following steps.

Trial Method	Steps
Via an HTML form on your company website	<p>Log a case to associate the new Trialforce template with the sign-up form.</p> <ol style="list-style-type: none"> a. Log in to the Partner Community. b. Under the Support tab, click New Case. c. Select the AppExchange and Feature Requests > Trialforce category. d. For the description, provide the TSO ID, the ID of the new Trialforce template, and the name of your sign-up form.
Using the API	<p>Log a case to get the template approved for SignupRequest API use.</p> <ol style="list-style-type: none"> a. Log in to the Partner Community. b. Under the Support tab, click New Case. c. Select the AppExchange and Feature Requests > Trialforce category. d. For the description, provide the TSO ID, the ID of the new Trialforce template, and the organization to use for creating sign-ups.

Trialforce Best Practices

Here are some best practices for using Trialforce.

- Create several Trialforce Source Organizations (TSOs) for customized trial experiences, for example, one for each managed package, industry vertical solution, country.
- Load sample data into the TSO.
- Apply custom branding to your trial signup form, login page, and emails.
- Update your Trialforce template each time you release a new version of your app.
- Once you've set up Trialforce, go through the signup flow to confirm everything is working as you expect it to. This can also help you identify areas the signup process can be improved.

Although Trialforce was primarily designed for enabling free trials, it's also useful in other contexts. For example, you can use it to:

- Create trial organizations for sales demos.
- Create test organizations with sample data for internal QA.

Creating Signups using the API

You can use API calls to the SignupRequest object to create trial organizations for prospective customers. When creating trial organizations (or signups) using a web form, there's no way to customize the signup process or track its status. Using the API, you can collect and analyze detailed information on all signups from your business organization. This gives you more control over the signup process, and enhanced visibility into your prospective customers. For example, you can:

- Run reports and collect metrics, such as the number of signups per day or the number of signups in different countries.
- Customize the SignupRequest object to add fields of special interest to your company.
- Create triggers to initiate specific actions, such as sending an email notification, whenever a new signup request is made.
- Enable signups from a wide range of client applications and devices, so you have additional channels for customer acquisition.

To start creating new signups using the API:

1. Create a Trialforce Source Organization (TSO) from your Trialforce Management Organization.
2. Install your app in the TSO, along with any sample data that might be useful for the trial.
3. Configure the TSO as you want your customers to experience it, including specifying any custom branding.
4. Create a Trialforce template from the TSO.
5. File a case to activate this feature.
 - a. Log in to the [Partner Community](#).
 - b. Under the Support tab, click **New Case**.
 - c. Select the **AppExchange and Feature Requests > Trialforce** category.
 - d. In the description, provide the following details.
 - the organization ID of your TSO
 - the template ID of the Trialforce template you want to use
 - the organization you plan to use for creating signups (so the appropriate user permission can be enabled)

USER PERMISSIONS

To create or view signup requests:

- "Signup Request API"



Note: Although you can create new signups from any organization with the appropriate permissions, we recommend doing so from your business organization. You can then easily integrate signup data with your existing business processes. For example, you can create a workflow rule to convert each signup request into a lead or run reports to track the number of signups in a given period.

You'll be notified by email once the template is approved. It can then be used to create new signups by making API calls to the SignupRequest object. See below for details of the SignupRequest object and a code sample demonstrating its use. For more information on working with objects, see the [Object Reference for Salesforce and Force.com](#)

SignupRequest

Represents a request for a new Trialforce signup. This object is available in API version 27.0 and later.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`

Fields

Field Name	Details
AuthCode	<p>Type string</p> <p>Properties Create, Filter, Group, Sort</p> <p>Description A one-time authorization code that can be exchanged for an OAuth access token and refresh token using standard Salesforce APIs. It's used in conjunction with <code>ConnectedAppCallbackUrl</code> and <code>ConnectedAppConsumerKey</code>, when the specified connected app hasn't been configured with an X.509 certificate. This is a read-only field provided by the system once the signup request has been processed. This field is available in API version 29.0 and later.</p>
Company	<p>Type string</p> <p>Properties Create, Filter, Group, Sort</p> <p>Description The name of the company requesting the trial signup.</p>
ConnectedAppCallbackUrl	<p>Type string</p> <p>Properties Create, Filter, Group, Sort</p>

Field Name	Details
	<p>Description</p> <p>When used in conjunction with <code>ConnectedAppConsumerKey</code>, specifies a connected app that should be approved automatically during the signup creation. This field is available in API version 28.0 and later.</p>
<code>ConnectedAppConsumerKey</code>	<p>Type</p> <p>string</p> <p>Properties</p> <p>Create, Filter, Group, Sort</p> <p>Description</p> <p>When used in conjunction with <code>ConnectedAppCallbackUrl</code>, specifies a connected app that should be approved automatically during the signup creation. This field is available in API version 28.0 and later.</p>
<code>Country</code>	<p>Type</p> <p>string</p> <p>Properties</p> <p>Create, Filter, Group, Sort</p> <p>Description</p> <p>The two-character, upper-case ISO-3166 country code. You can find a full list of these codes at a number of sites, such as: www.iso.ch/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html. The language of the trial organization is auto-determined based on the value of this field.</p>
<code>CreatedOrgId</code>	<p>Type</p> <p>string</p> <p>Properties</p> <p>Filter, Group, Nillable, Sort</p> <p>Description</p> <p>The 15-character organization ID of the trial organization created. This is a read-only field provided by the system once the signup request has been processed.</p>
<code>CreatedOrgInstance</code>	<p>Type</p> <p>string</p> <p>Properties</p> <p>Filter, Group, Nillable, Sort</p> <p>Description</p> <p>The server instance of the new trial organization, for example, "na8." This field is available in API version 29.0 and later.</p>
<code>ErrorCode</code>	<p>Type</p> <p>string</p>

Field Name	Details
	<p>Properties Filter, Group, Nillable, Sort</p> <p>Description The error code if the signup request isn't successful. This is a read-only field provided by the system to be used for support purposes.</p>
FirstName	<p>Type string</p> <p>Properties Create, Filter, Nillable, Sort</p> <p>Description The first name of the admin user for the trial signup.</p>
LastName	<p>Type string</p> <p>Properties Create, Filter, Group, Sort</p> <p>Description The last name of the admin user for the trial signup.</p>
SignupEmail	<p>Type email</p> <p>Properties Create, Filter, Group, Sort</p> <p>Description The email address of the admin user for the trial signup.</p>
Status	<p>Type picklist</p> <p>Properties Filter, Group, Sort, Update</p> <p>Description The status of the request. Possible values are New, In Progress, Error, or Success. The default value is New.</p>
Subdomain	<p>Type string</p> <p>Properties Create, Filter, Group, Sort</p>

Field Name	Details
	<p>Description</p> <p>The subdomain for the new trial organization when it uses a custom My Domain. The maximum length is 33 characters for Developer Edition (DE) and 40 characters for all other editions (because a suffix is appended to all DE organizations).</p>
SuppressSignupEmails	<p>Type</p> <p>boolean</p> <p>Properties</p> <p>Filter, Group, Nillable, Sort</p> <p>Description</p> <p>When set to <code>true</code>, no signup emails are sent when the trial organization is created. This field is used for the Proxy Signup feature, and is available in API version 29.0 and later.</p>
TemplateId	<p>Type</p> <p>string</p> <p>Properties</p> <p>Create, Filter, Group, Sort</p> <p>Description</p> <p>The 15-character ID of the approved Trialforce template that is the basis for the trial signup. The template is required and must be approved by Salesforce.</p>
TrialDays	<p>Type</p> <p>anyType</p> <p>Properties</p> <p>Create, Defaulted on create, Filter, Group, Sort</p> <p>Description</p> <p>The duration of the trial signup in days. Must be equal to or less than the trial days for the approved Trialforce template. If not provided, it defaults to the trial duration specified for the Trialforce template.</p>
TrialSourceOrgId	<p>Type</p> <p>string</p> <p>Properties</p> <p>Filter, Group, Nillable, Sort</p> <p>Description</p> <p>The 15-character organization ID of the Trialforce Source Organization from which the Trialforce template was created.</p>
Username	<p>Type</p> <p>string</p> <p>Properties</p> <p>Create, Filter, Group, Sort</p>

Field Name

Details

Description

The username of the admin user for the trial signup. It must follow the address convention specified in RFC822: www.w3.org/Protocols/rfc822/#z10

Usage

The Java class below uses the REST API to create a SignupRequest object. It authenticates to the Trialforce Management Organization and then posts a request to the SignupRequest object.

Here are the variables you need to specify in this example.

- SERVER — The name of the host server for the Trialforce Management Organization (TMO), for example, "na1.salesforce.com."
- USERNAME — The admin username for the TMO.
- PASSWORD — The concatenation of the admin password and the security token for the TMO. To get an email with the security token, from your personal settings in Salesforce select **Reset My Security Token** and click **Reset Security Token**.
- CLIENT_ID — From Setup in Salesforce, click **Create > Apps** and click **New** under Connected Apps. Enter values for the required fields (the Callback URL is required but can initially be set to any valid URL as it's not used), grant full access for the OAuth scopes in the "Selected OAuth Scopes" selector, and click **Save**. Then copy the value of "Consumer Key" and use it for this variable.
- CLIENT_SECRET — On the same page, click **Click to reveal**. Then copy the value of "Consumer Secret" and use it for this variable.

```
public class IsvSignupDriver {
    private static final String SERVER = server_name:port;
    private static final String USERNAME = tmo_username;
    private static final String PASSWORD = tmo_passwordsecurity_token;
    private static final String CLIENT_ID = consumer_key;
    private static final String CLIENT_SECRET = consumer_secret;

    private static SignupRequestInfo signupRequest = null;

    public static String createSignupRequest (SignupRequestInfo sr)
        throws JSONException, IOException {
        JSONObject createResponse = null;
        signupRequest = sr;
        JSONObject loginResponse = login(SERVER, USERNAME, PASSWORD);
        String instanceUrl = loginResponse.getString("instance_url");
        String accessToken = loginResponse.getString("access_token");
        createResponse = create(instanceUrl, accessToken);
        System.out.println("Created SignupRequest object: " + createResponse + "\n");
        return createResponse.toString();
    }

    /* Authenticates to the TMO using the required credentials */

    private static JSONObject login(String server, String username, String password)
        throws ClientProtocolException, IOException, JSONException {
        String authEndPoint = server + "/services/oauth2/token";
        HttpClient httpClient = new DefaultHttpClient();
        try {
            HttpPost post = new HttpPost(authEndPoint);
```

```

        List<NameValuePair> params = new ArrayList<NameValuePair>();
        params.add(new BasicNameValuePair("grant_type", "password"));
        params.add(new BasicNameValuePair("client_id", CLIENT_ID));
        params.add(new BasicNameValuePair("client_secret", CLIENT_SECRET));
        params.add(new BasicNameValuePair("username", username));
        params.add(new BasicNameValuePair("password", password));
        post.setEntity(new UrlEncodedFormEntity(params, Consts.UTF_8));

        BasicResponseHandler handler = new BasicResponseHandler();
        String response = httpClient.execute(post, handler);
        return new JSONObject(response);
    } finally {
        httpClient.getConnectionManager().shutdown();
    }
}

/* Posts a request to the SignupRequest object */

private static JSONObject create(String instanceUrl, String accessToken)
    throws ClientProtocolException, IOException, JSONException {
    HttpClient httpClient = new DefaultHttpClient();
    try {
        HttpPost post = new HttpPost(instanceUrl +
            "/services/data/v27.0/objects/SignupRequest/");
        post.setHeader("Authorization", "Bearer " + accessToken);
        post.setHeader("Content-Type", "application/json");

        JSONObject requestBody = new JSONObject();
        requestBody.put("TemplateId", signupRequest.getTemplateID());
        requestBody.put("SignupEmail", signupRequest.getEmail());
        requestBody.put("username", signupRequest.getUsername());
        requestBody.put("Country", "US");
        requestBody.put("Company", signupRequest.getCompanyName());
        requestBody.put("lastName", signupRequest.getLastName());

        StringEntity entity = new StringEntity(requestBody.toString());
        post.setEntity(entity);
        BasicResponseHandler handler = new BasicResponseHandler();
        String response = httpClient.execute(post, handler);
        return new JSONObject(response);
    } finally {
        httpClient.getConnectionManager().shutdown();
    }
}
}

```

Error Codes

If the signup fails, the system generates an error code that can help you identify the cause. This table shows the most important error codes.

Error Code	Description
C-1007	Duplicate username.
C-1015	Error while establishing the new organization's My Domain settings. Contact Salesforce support for assistance.
C-1016	Error while configuring the OAuth connected app for Proxy Signup. Verify that your connected app has a valid consumer key, callback URL, and unexpired certificate (if applicable).
C-1018	Invalid subdomain value provided during signup.
C-1019	Subdomain in use. Please choose a new subdomain value.
C-9999	Generic "fatal error." Contact Salesforce support for assistance.
S-1006	Invalid email address (not in a proper email address format).
S-2006	Invalid country code.
T-0001	Template ID not valid (not in the format OTTxxxxxxxxxxx).
T-0002	Template not found. Either the template doesn't exist (it may have been deleted), or it doesn't exist at the appropriate version.
T-0003	Template not approved for use by Salesforce.

Signup Request Home

Clicking the Signup Requests tab displays the signup requests home page. In the **Recent Signup Requests** section:

- Click **New** to create a new signup.
- Click the number of any signup request to see details about it, including its history and approval status.
- Select an item from the drop-down list to display a list of the signup requests matching that criterion. From the list, you can click any signup request name to go directly to the signup request detail.
- To show a filtered list of items, select a predefined list from the **View** drop-down list, or click **Create New View** to define your own custom views. To edit or delete any view you created, select it from the **View** drop-down list and click **Edit**.



Note: You must get your Trailforce template approved, by filing a case in the Partner Community, before you can create new signups from this page.

USER PERMISSIONS

To create or view signup requests:

- "Signup Request API"

Creating a Signup Request

1. Select **Signup Request** from the Create New drop-down list in the sidebar, or click **New** next to **Recent Signup Requests** on the signup requests home page.
2. Enter the information for the signup request.
3. Click **Save** when you're finished, or click **Save & New** to save the current signup request and add another.

USER PERMISSIONS

To create or view signup requests:

- "Signup Request API"

Viewing Signup Request Details

From the Signup Request detail page:

- Click **Delete** to delete the signup request
- Click **Clone** to create a new signup request with the same attributes as this one

The detail page has the following sections.

- [Signup Request Detail](#)
- [Signup Request History](#)

USER PERMISSIONS

To create or view signup requests:

- "Signup Request API"

Signup Request Detail

This section displays the following attributes (in alphabetical order).

Attribute	Description
Company	The name of the company requesting the trial signup.
Country	The two-character, upper-case ISO-3166 country code. You can find a full list of these codes at a number of sites, such as: www.iso.ch/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html
Created Org	The 15-character Organization ID of the trial organization created. This is a read-only field provided by the system once the signup request has been processed.
Email	The email address of the admin user for the trial signup.
Error Code	The error code if the signup request isn't successful. This is a read-only field provided by the system to be used for support purposes.
First Name	The first name of the admin user for the trial signup.
Last Name	The last name of the admin user for the trial signup.
Source Org	The 15-character Organization ID of the Trialforce Source Organization from which the Trialforce template was created.
Status	The status of the request. Possible values are <code>New</code> , <code>In Progress</code> , <code>Error</code> , or <code>Success</code> . The default value is <code>New</code> .
Template	The 15-character ID of the approved Trialforce template that is the basis for the trial signup. The template is required and must be approved by salesforce.com.
Template Description	The description of the approved Trialforce template that is the basis for the trial signup.
Trial Days	The duration of the trial signup in days. Must be equal to or less than the trial days for the approved Trialforce template. If not provided, it defaults to the trial duration specified for the Trialforce template.
Username	The username of the admin user for the trial signup. It must follow the address convention specified in RFC822: www.w3.org/Protocols/rfc822/#z10

Signup Request History

This section shows the date the signup request was created, the user who created it, and the actions that have been performed on it.

Adding Custom Fields to Signup Requests

You can add custom fields to the SignupRequest object, as for any other standard object.

1. In your Salesforce Management Organization, from Setup, click **Customize > Signup Requests > Fields**.
2. Click **New**.
3. Specify the details of the custom field and click **Save**.

To see the custom field in a list of existing SignupRequest records, create a custom view containing that field on the Signup Requests tab.

USER PERMISSIONS

To create or view signup requests:

- "Signup Request API"

Running Reports on Signup Requests

Once a few SignupRequest records have been created, you can run custom reports on them.

1. In your Salesforce Management Organization, from Setup, click **Create > Report Types** and click **Custom Report Types**.
2. Select **Signup Requests** as the Primary Object.
3. Enter a label, name, description, and store in a category such as Administrative Reports.
4. Finish the wizard, and save the Report Type.
5. Configure the report with the fields you're interested in and click **Save**.
6. Select the Reports tab and click **New Report** to create a Report from your new Report Type.
7. Select your report type name and click **Create**.

Once you've created the report, you can run it periodically to see trends in the data.

USER PERMISSIONS

To create or view signup requests:

- "Signup Request API"

Using Triggers with Signup Requests

You can set up triggers to initiate specific actions, each time a signup request is submitted.

1. In your Salesforce Management Organization, from Setup, click **Customize > Signup Requests > Triggers**.
2. Click **New**.
3. Add the code for the trigger, and click **Save**.

For example, this trigger inserts a new lead based on information in the signup request.

```
trigger SignupRequestTrigger on SignupRequest (after insert) {
    private SignupRequest[] sr = Trigger.new;
    Lead l = new Lead(
        LastName = sr[0].LastName,
        FirstName = sr[0].FirstName,
        Company = sr[0].Company,
        Email = sr[0].SignupEmail,
        LeadSource = 'Trial Signup'
    );
    insert l;}
```

You can verify that a Lead record is created, each time you create a SignupRequest. To easily find a specific lead, you can sort leads by email.

USER PERMISSIONS

To create or view signup requests:

- "Signup Request API"

Creating Proxy Signups for OAuth and API Access

Using the SignupRequest object, you can programmatically create a new organization without any system-generated emails being sent to the user. You can then obtain an OAuth access token to log in to the organization and make API requests from it, without any action by the user. This is called proxy signup because it enables you to create and operate the organization on the user's behalf, without their knowledge that you're using Salesforce behind the scenes..

In the traditional signup process, when you create a new organization, the user receives a system-generated email containing the login URL and initial password for logging in to the organization. The user then has to log in and explicitly grant you API access to make calls into the organization on his behalf. With proxy signup, no user emails are generated and no action is required by the user to provide you API access.

The ability to create and manage organizations by proxy expands your options for integrating Salesforce with external applications on other platforms. It enables you to incorporate any feature of the Force.com platform into your own application, without exposing the Salesforce user interface (UI). In effect, all features of Salesforce can be decoupled from the UI and are available to integrate into any other application runtime or UI in a seamless and invisible way.

For example, suppose an ISV has a web application, built on the .NET platform, that helps companies manage travel expense reporting and reimbursement for employees. The ISV might want to integrate Chatter into its application, so all employees of a company can share feedback and tips about their travel experiences with each other. The ISV can do this by using the appropriate Salesforce APIs to implement the following solution.

1. Use proxy signup to create a Salesforce organization for each of its customers.
2. Create users in each customer organization for all employees of that company.
3. Set up and maintain a Chatter group for sharing travel information.
4. Monitor each user's Chatter feed and extract information from individual posts.
5. Insert the information into its application, and display it in the existing UI.

This enables the ISV to provide its customers access to Chatter functionality, without having to develop it from scratch. The ISV's customers experience Chatter as a natural extension of the existing application, in an interface they're familiar with, and without needing to know about or log in to Salesforce. The same approach can be extended to any other feature of Salesforce, including standard and custom objects, Apex, and Visualforce. In effect, proxy signup gives ISVs the ability to consume Salesforce as a service, integrating its features into applications on any platform, without exposing the Salesforce UI. The potential applications are limited only by the ISV's imagination.

Here are the steps for creating a proxy signup.

1. Log in to a Developer Edition organization (which has the Connected Apps user permission enabled by default).
2. In the Setup menu, click **Create > Apps** and click **New** under Connected Apps.
3. Enter values for the required fields. You must specify an X.509 certificate and grant full and refresh token access for the OAuth scopes in the "Selected OAuth Scopes" selector. The callback URL is required but can initially be set to any valid URL as it's not used. Click **Save** when you're done.
4. Record the value of `Consumer Key` on the same page. Also, click **Click to reveal** and record the value of `Consumer Secret`.
5. Package the Connected App by adding it as a component to a new package. Record the Installation URL value for the package.
6. Log in to your Salesforce Management Organization and create a new Salesforce Source Organization from it.
7. Log in to your Salesforce Source Organization and install the package containing the Connected App, using the installation URL from step 5.
8. Once the Connected App is installed in the Salesforce Source Organization, you can customize it by going to **Setup > Manage Applications**. You can see the Connected App and can edit its attributes. Specify the appropriate profiles and permission sets, and

USER PERMISSIONS

To create or view signup requests:

- "Signup Request API"

choose the option “Admin approved users are pre-authorized” in the OAuth policies section. This ensures you can authenticate into the organization on behalf of users with these criteria.

9. Once you’ve configured the Trialforce Source Organization to your requirements, create a Trialforce template from it. Select the **All Setup and Data** radio button when creating the Trialforce template.
10. File a case in the [Partner Community](#) to get approval for creating new signups using the template.
11. Once the template is approved, you can sign up a new organization using the SignupRequest object. You will need to specify the OAuth values necessary to connect to the newly-created organization, that is: Consumer Key and Callback URL.

```
POST https://mycompany-tmo.salesforce.com/services/data/v27.0/subjects/SignupRequest/
Authorization Bearer
  00Dxx0000001gR6!ARoAQAS3Uc6brlY8q8TWrrI_u1THuUGmSAp
  XrksSniyjom9kXfDac4UP.m9FAPjTw9ukJfKqWuD8pA9meeLaltRmNFvPqUn7
Content-Type application/json Body:
{
  "TemplateId": "0TT0000000000001",
  "SignupEmail": "john.smith@mycompany.com",
  "Username": "gm@trial1212.org",
  "Country": "US",
  "Company": "salesforce.com",
  "LastName": "Smith",
  "ConnectedAppConsumerKey":
    "3MVG9AOp4kbriZOLfSVjG2Pxa3cJ_nOkwhxL1J1AuV22u8bm82FtDtWfVv__
    Vs6mvqoVbAnwsChp9YT4bfrYu",
  "ConnectedAppCallbackUrl":
    "https%3A%2F%2Fwww.mysite.com%2Fcode_callback.jsp" }
```

When the ConnectedAppConsumerKey and ConnectedAppCallbackUrl fields are specified in the SignupRequest object, a proxy signup flow is triggered to automatically approve an existing Connected App for use in this new organization. In that flow, no signup-related emails are sent to the user. With knowledge of the admin username, consumer key and consumer secret, you now have all the information required to:

- make API requests to the newly-created organization as an admin user of that organization.
- request an updated access token at any time in the future.

Trialforce FAQ

This section contains a list of frequently asked questions about Trialforce.

- [How do I upgrade my trial with a new version of my offering?](#)
- [Can I distribute my app or component using both Trialforce and the AppExchange?](#)
- [How are trials different from Trialforce?](#)
- [Is it possible to install another app in a trial organization?](#)

How do I upgrade my trial with a new version of my offering?

Install the new version of the package into your Trialforce source organization. After upgrading, create a new Trialforce template and use the template as the basis for your trial.

Can I distribute my app or component using both Trialforce and the AppExchange?

Of course! The most effective way to distribute your offering is by using Trialforce and the AppExchange together. You can even advertise your Trialforce page on your AppExchange listing and vice versa. Generally, the AppExchange is best for engaging existing Salesforce customers, while Trialforce works great with new customers.

How are trials different from Trialforce?

Trials are administered from the AppExchange whereas Trialforce is administered from your own website.

Is it possible to install another app in a trial organization?

Yes. The Trialforce master organization is a fully functioning Salesforce organization. Your customer will have your app installed and can subsequently install additional apps into the same organization as they see fit. It's just like any other free trial of Salesforce.

CHAPTER 10 Supporting Your Customers

In this chapter ...

- [Subscriber Support Console](#)
- [Usage Metrics](#)

App publishers are responsible for end user support of all their listings. When customers contact Salesforce Customer Support with a question about your listing, we direct the user to the support information on the About and Support tabs of your listing. Make sure your AppExchange listings include support information.

If you have installed the License Management App (LMA), you can log in to a customer's organization and provide administrative support for your customers. This feature is only available for managed packages that have passed the security review. For more information, see [Logging in to Subscriber Organizations](#).

Subscriber Support Console

Using the Subscriber Support Console, you can easily access information about all your subscribers, such as which Salesforce Edition they are using and if they are over their limits. Subscribers can also grant you login access to troubleshoot issues directly within the app, in the familiar manner that they grant login access to administrators. Once granted access, you can log in to the subscriber's organization and directly view their configuration and data to help troubleshoot problems.



Note: This feature is available to eligible Salesforce partners. For more information on the Partner Program, including eligibility requirements, please visit us at www.salesforce.com/partners.

Viewing Subscriber Details

The Subscriber Overview page, accessed by clicking the organization's name from the **Subscribers** tab of the LMA, provides detailed information about each subscriber organization. This can give you insight into how a customer is using your app and help you in troubleshooting problems.

Under Organization Details:

- The name and contact information is in Setup, on the **Company Profile > Company Information** page in the subscriber's organization. This may differ from the information shown in your LMA lead, account, or contact records.
- Organization ID is a unique ID that identifies this customer's Salesforce organization.
- Instance determines which Salesforce data center this customer's organization resides in. It also determines when the customer will get upgraded with a new version of Salesforce. See trust.salesforce.com during the release period to understand which version of Salesforce the customer is using.

The page also includes these related lists.

Limits

Information on the file space, data space, and number of API requests associated with this customer, as a percentage.

Login Access Granted

A list of users who have granted login access and the date when access will expire.

Packages and Licensing

A list of all packages installed in this organization and associated with this LMA. For each package, it shows the version of the app a customer is currently using, the total number of licenses provisioned to the subscriber and the number they've used. This information should match the license record for the subscriber in your LMA.

Requesting Login Access

Ask the user to go to their personal settings and click **Grant Account Login Access** or **Grant Login Access**, whichever appears, to grant access. If the publisher isn't listed on this page, it's for one of the following reasons:


- A system administrator disabled the ability for non-administrators to grant access.
- The user doesn't have a license for the package.
- The package is licensed to the entire organization. Only administrators with the "Manage Users" permission enabled on their profile can grant access.
- The organization preference **Administrators Can Log in as Any User** is enabled.



Note: Unless the organization preference **Administrators Can Log in as Any User** is enabled, access is granted for a limited amount of time, and the subscriber can revoke access at any time. Any changes you make while logged in as a subscriber are logged in the audit trail.


Logging in to Subscriber Organizations

Available in: **Enterprise, Performance, Unlimited**, and **Developer** Editions

 **Note:** This feature is only available in organizations with a full Salesforce license.

To log in, once a user has granted you access:

1. In the License Management App (LMA), click the **Subscribers** tab.
2. To find a subscriber organization quickly, enter a subscriber name or organization ID in the search box and click **Search**.
3. Click the name of the subscriber organization.
4. On the Organization Details page, click **Login** next to a user's name. Note that you have the same permissions as the user you logged in as.
5. When you're finished troubleshooting, from Setup, click **Return to Subscriber Overview** to return to your organization.

 **Note:** Only subscribers who have installed at least one managed package that is linked to your LMA will appear in this list.

USER PERMISSIONS

To log in to subscriber organizations:

- "Log in to Subscriber Organization"

Best Practices

- When you access a subscriber organization, you're logged out of your LMO (License Management Organization). You can set up a my domain so that you aren't automatically logged out of your LMO when you log in to a subscriber organization. To set up a my domain, from Setup, click **Domain Management > My Domain**.
- Be careful to allow only trusted support and engineering personnel to log in to a subscriber's organization. Since this feature may include full read/write access to customer data and configurations, it's vital to your reputation to preserve their security.
- Control who has access by giving the "Log in to Subscriber Organization" user permission to specific support personnel, via a profile or permission set.

Troubleshooting in Subscriber Organizations

When logged in as a user in a subscriber's organization, you can generate Apex debug logs that contain the output from your managed packages. This includes logging that would normally not be exposed to the subscriber. Using this log information, you can troubleshoot issues that are specific to that subscriber organization.

1. Launch the Developer Console from *Your Name* > **Developer Console**.
2. Perform the operation and view the debug log with your output. If the user has access, set up a Debug Log; from Setup, click **Monitoring > Debug Logs** or **Logs > Debug Logs**.

Note that subscribers will be unable to see the logs you set up or generate since they contain your Apex code unobfuscated. In addition, you can also view and edit data contained in protected custom settings from your managed packages when logged in as a user.

Usage Metrics

You can collect detailed usage metrics from each organization in which your managed package is installed. By analyzing this information, you can gain valuable insights into the utilization and performance of your app across your entire customer base. For example, you can identify:

- The features most and least used — this can help you prioritize your development efforts when planning the next version of your app.
- The customers using your app most intensively — these are your most valuable customers.
- The customers whose usage of your app is minimal or declining — these are the customers most at risk of attrition.

You can collect the following daily metrics on two types of components in a managed package.


- **Custom objects** — the total number of records existing per organization in each custom object. This enables you to track how the usage of that custom object is growing with time in any subscriber organization, which is a reliable indicator of how much it's being utilized
- **Visualforce pages** — the number of times per organization each Visualforce page was accessed, the number of unique users who accessed it, and the average loading time (in milliseconds). By comparing the metrics for different Visualforce pages, you can determine the relative popularity of different parts of your app in a specific customer organization, as well as trends across all customers.

The custom objects data is a snapshot that reflects the state of the organization at the time the database was sampled, while the Visualforce data covers usage over a 24-hour period.

The usage metrics data for all production organizations in a given instance is merged and written into a text file, in a specified format, once a day. Currently, no data is collected on packages installed in sandbox organizations or on managed beta packages.

This feature is intended for API access only. You must write a custom process to collect the metrics data from the reporting organization, and export it to a system of your choice for analysis. This gives you the maximum flexibility to monitor and analyze the usage trends most relevant for your app.

Your customers' consent is not required for usage data to be collected, and there's no way for them to opt out. This ensures you receive complete data for your entire customer base. Allowing some users to be excluded would skew the results, making the data less useful.

 **Note:** If any of your customers have concerns about privacy, reassure them any data collected is limited to usage statistics. No customer data is ever exposed to the ISV under any circumstances. This is consistent with salesforce.com's emphasis on trust as a core value.

EDITIONS

Available in:


- Professional
- Enterprise
- Performance
- Unlimited
- Developer

Setting up Usage Metrics

To set up Usage Metrics for any package, two organizations have special importance.

- **Release organization** — the Development Edition organization used to upload the package.
- **Reporting organization** — the organization to which the usage data is delivered, on a daily basis.

The release organization and reporting organization must be members of the same Environment Hub. This is a security feature, to ensure usage data is only delivered to an organization controlled by the developer of the package. We recommend using the Environment Hub as your reporting organization.

 **Note:** For information on setting up and connecting organizations to the Environment Hub, see [Environment Hub](#) on page 83.

To set up Usage Metrics for a package:

1. Set up Environment Hub, if you haven't already done so.

2. Connect the release organization to the Environment Hub.
3. Connect the reporting organization to the Environment Hub (if they're different).
4. Log a case in the [Partner Community](#) to activate Usage Metrics. You'll need to provide the package ID for your app.

Once the feature is activated, you'll receive a confirmation email. From that point on, usage data will automatically be collected from all organizations in which your package is installed, and delivered to the reporting organization on a daily basis. There is no way to get usage data retroactively, that is, for any period prior to the activation of Usage Metrics.

Accessing Usage Metrics Data

The usage data for a package is stored in `MetricsDataFile` records in your reporting organization. Once you activate the Usage Metrics feature, one new record is created for all custom objects and one for all Visualforce pages, per Salesforce instance per day.



Note: To see the number of Salesforce instances currently in use, visit trust.salesforce.com.

The usage data for each day and instance is stored as a text file, encoded in Base 64, in the `MetricsDataFile` field of the record. Other fields in the record identify these properties.

- Namespace prefix of the package
- Salesforce instance
- Start time and date of data collection
- End time and date of data collection
- Size of the data file in bytes
- Type of data, which is either `CustomObject` or `Visualforce`

The custom objects data is a snapshot that reflects the state of the organization at the time the database was sampled, while the Visualforce data covers usage over a 24-hour period.

The custom object count is a snapshot captured once each day. Here's a section of a sample data file for custom objects. It shows there were 3500 and 1500 records in the `Alpha` and `Beta` custom objects, respectively, in the specified customer organization on the specified day.

```
"00Dxx0000001gbk","org1","Enterprise Edition","TRIAL","Alpha", "3500"
"00Dxx0000001gbk","org1","Enterprise Edition","TRIAL","Beta", "1500"
```

In a record for Visualforce pages, each row of the text file contains usage data in the following order.

- Organization ID
- Organization name
- Organization edition
- Organization status
- Package version number
- Name of the Visualforce page
- Number of times the page was accessed
- Number of unique users who accessed the page
- Average loading time of the page, in milliseconds

The Visualforce counts for each organization measure the number of times the page was viewed in the duration between the start and end times. Here's a section of a sample data file for Visualforce pages.

```
"00Dxx0000001gbk","org1","Enterprise Edition","TRIAL","1.0","/apex/gm12__f1","1","1","66.0"
"00Dxx0000001gbk","org1","Enterprise Edition","TRIAL","1.0","/apex/gm12__f2","1","1","128.0"
"00Dxx0000001gbk","org1","Enterprise Edition","TRIAL","1.0","/apex/gm12__f3","1","1","107.0"
"00Dxx0000001gbf","org1","Enterprise Edition","TRIAL","1.0","/apex/gm12__f1","5","1","73.6"
"00Dxx0000001gbf","org1","Enterprise Edition","TRIAL","1.0","/apex/gm12__f2","1","1","72.0"
"00Dxx0000001gbf","org1","Enterprise Edition","TRIAL","1.0","/apex/gm12__f3","7","1","50.8"
```

You must write a custom process to query the reporting organization to collect the metrics data, and export it to a system of your choice for analysis. This gives you the maximum flexibility to monitor and analyze the usage trends most relevant for your app.

MetricsDataFile

Represents a data file containing usage metrics on all installations of a managed package in a Salesforce instance. This object is available in API version 30.0 and later.

Supported Calls

`query()`, `delete()`

Fields

Field Name	Details
MetricsDataFile	Type base64 Properties Filter, Query, Sort Description A text file containing the usage data encoded in Base 64.
MetricsDataFileContentType	Type string Properties Filter, Query, Sort Description The format of the data file. Currently, the only allowed value is <code>text/csv</code> .
MetricsDataFileLength	Type int Properties Filter, Query, Sort Description The size of the data file in bytes.

Field Name	Details
MetricsRunDate	<p>Type dateTime</p> <p>Properties Filter, Query, Sort</p> <p>Description The date when the usage metrics collection job was run.</p>
MetricsEndDate	<p>Type dateTime</p> <p>Properties Filter, Query, Sort</p> <p>Description The end time and date for the data collection.</p>
MetricsStartDate	<p>Type dateTime</p> <p>Properties Filter, Query, Sort</p> <p>Description The start time and date for the data collection.</p>
MetricsType	<p>Type picklist</p> <p>Properties Filter, Query, Sort</p> <p>Description The type of data being collected. The possible values are <code>CustomObject</code> and <code>Visualforce</code>.</p>
NamespacePrefix	<p>Type string</p> <p>Properties Filter, Query, Sort</p> <p>Description The namespace prefix of the package for which data is being collected.</p>
SendingInstance	<p>Type string</p> <p>Properties Filter, Query, Sort</p>

Field Name	Details
	Description The server instance from which this data was collected, for example, "na8."

Usage

Use this object to access customer usage metrics for a managed package. Each record contains one day's data, on either custom objects or Visualforce pages, for all organizations in a Salesforce instance that have the package installed. The following data is collected each day.

- **Custom objects** — the number of records stored in each custom object.
- **Visualforce pages** — the number of times each Visualforce page was accessed, the number of unique users who accessed it, and the average loading time (in milliseconds).

Usage Metrics Visualization

The Usage Metrics Visualization app, available from Salesforce Labs on the AppExchange, enables you to visualize trends in usage metrics data for your app. You can use the Usage Metrics Visualization app to generate charts showing changes in various app metrics, over a specified duration, for one or more customer organizations.

The app must be installed in your Usage Metrics reporting organization and requires Usage Metrics to be enabled in advance, so some data is available for analysis. You can analyze data going back a maximum of 30 days. If Usage Metrics wasn't enabled for the entire time period that you specify, only partial data is plotted.

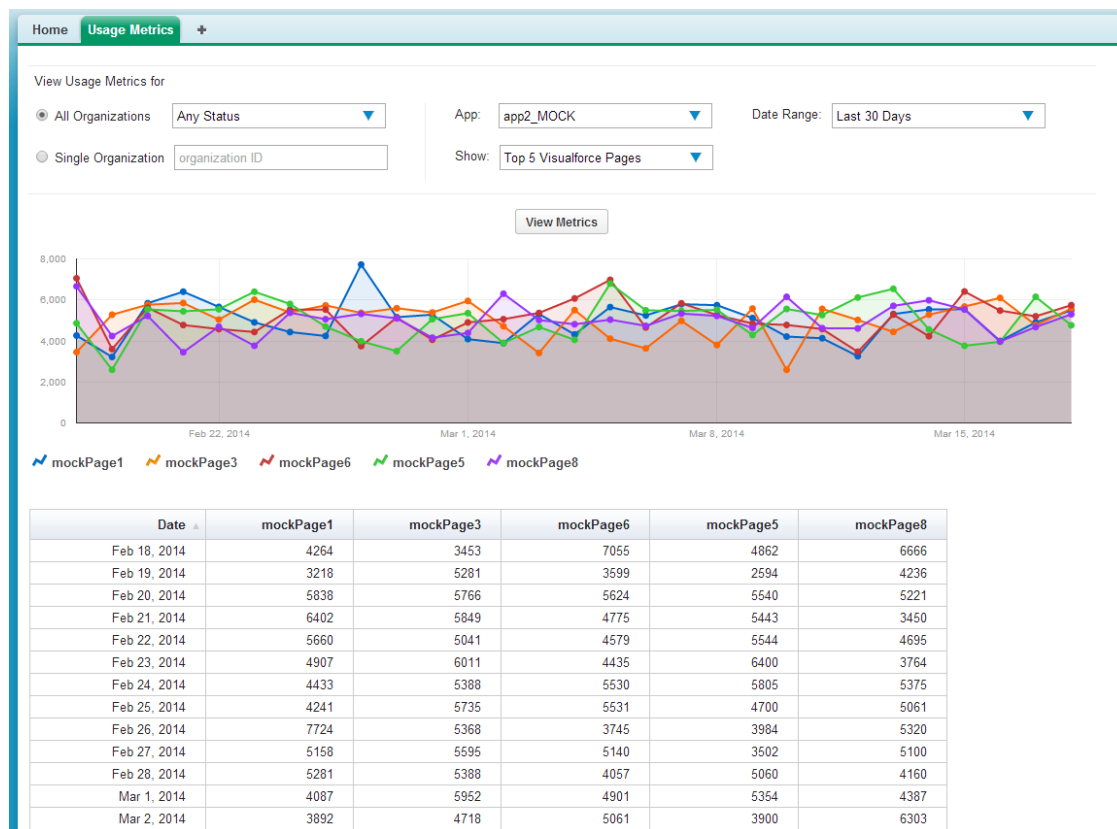
The app is intended as a reference implementation, for illustration purposes only. It's distributed as an unmanaged package, so you can review its components and extend or customize it to meet your requirements. If your visualization needs are more complex, you can export the raw metrics data from the reporting organization and analyze it by using custom code or a third-party tool.

To install the Usage Metrics Visualization app:

1. Go to the AppExchange and search for the Usage Metrics Visualization app.
2. Click **Get It Now**.
3. Enter the credentials for your reporting organization, and then click the login button.
4. Click **Install**.

You'll see a message describing the progress and a confirmation message after the installation is complete.

The Usage Metrics Visualization app showing data for the top five Visualforce pages.



To visualize the usage metrics data:

1. Specify the app whose metrics you want to view by selecting it from the App menu.



Note: You should have enabled Usage Metrics for your app at least a few days before, so some usage data is available to analyze.

2. Specify the organization(s) that you want to view metrics for by choosing one of these options.

- For a single organization, enter its Organization ID in the Single Organization field.
- For a group of organizations, select one of the following from the All Organizations menu.
 - Any Status
 - All Active: These are organizations used by paying customers.
 - All Free: These are Developer Edition (DE) organizations.
 - All Trial: These are trial organizations, which expire after a specified period.

3. Specify the type of metric that you want to visualize by selecting one of these values from the Show menu.

- Total Visualforce Page Views
- Top 5 Visualforce Pages
- Total Record Count
- Top 5 Objects by Record Count

4. Specify the time period to cover by selecting one of these values from the Date Range menu.

- Last 30 Days
- Last 7 Days
- Last 2 Days



Note: If the volume of usage data is too large, you might get an error message. In that case, choose a smaller date range and try again.

5. Click **View Metrics**.

The data you specified is displayed on the page as a chart and as a table. To visualize a different data set, change the parameters, and then click **View Metrics** again.

CHAPTER 11 Upgrading Your App

In this chapter ...


- [About Package Versions](#)
- [Creating and Uploading Patches](#)
- [Working with Patch Versions](#)
- [Publishing Upgrades to Managed Packages](#)
- [Pushing an Upgrade](#)

After you upload a packaged application, you can update it to fix bugs or introduce new functionality. When you update a package, you create a new package version.

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. Unmanaged packages are not upgradeable, so each package version is simply a set of components for distribution. A package version has more significance for managed packages. Packages can exhibit different behavior for different versions. Publishers can use package versions to evolve the components in their managed packages gracefully by releasing subsequent package versions without breaking existing customer integrations using the package.

Version numbers depend on the package release type, which identifies the way packages are distributed. There are two kinds:

Major Release

A major release denotes a  Managed - Released package. During these releases, the major and minor numbers of a package version increase to a chosen value.

Patch Release

A patch release is only for patch versions of a package. During these releases, the patch number of a package version increments.

The following table shows a sequence of version numbers for a series of uploads:

Upload Sequence	Type	Version Number	Notes
First upload	Managed - Beta	1.0	The first Managed - Beta upload.
Second upload	Managed - Released	1.0	A Managed - Released upload. Note that the version number does not change.
Third upload	Managed - Released	1.1	Note the change of the minor release number for this Managed - Released upload. If you are uploading a new patch version, you can't change the patch number.
Fourth upload	Managed - Beta	2.0	The first Managed - Beta upload for version number 2.0. Note the major version number update.
Fifth upload	Managed - Released	2.0	A Managed - Released upload. Note that the version number does not change.

When an existing subscriber installs a new package version, there is still only one instance of each component in the package, but the components can emulate older versions. For example, a subscriber may be using a managed package that contains an Apex class. If the publisher decides to deprecate a method in the Apex class and release a new package version, the subscriber still sees only one instance of the Apex class after installing the new version. However, this Apex class can still emulate the previous version for any code that references the deprecated method in the older version.

SEE ALSO:

[Working with Patch Versions](#)


[About Push Upgrades](#)

About Package Versions

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. Unmanaged packages are not upgradeable, so each package version is simply a set of components for distribution. A package version has more significance for managed packages. Packages can exhibit different behavior for different versions. Publishers can use package versions to evolve the components in their managed packages gracefully by releasing subsequent package versions without breaking existing customer integrations using the package.

Version numbers depend on the package release type, which identifies the way packages are distributed. There are two kinds:

Major Release

A major release denotes a  Managed - Released package. During these releases, the major and minor numbers of a package version increase to a chosen value.

Patch Release

A patch release is only for patch versions of a package. During these releases, the patch number of a package version increments.

When an existing subscriber installs a new package version, there is still only one instance of each component in the package, but the components can emulate older versions. For example, a subscriber may be using a managed package that contains an Apex class. If the publisher decides to deprecate a method in the Apex class and release a new package version, the subscriber still sees only one instance of the Apex class after installing the new version. However, this Apex class can still emulate the previous version for any code that references the deprecated method in the older version.

Package developers can use conditional logic in Apex classes and triggers to exhibit different behavior for different versions. This allows the package developer to continue to support existing behavior in classes and triggers in previous package versions while continuing to evolve the code.


When you are developing client applications using the API, you can specify the version of each package that you use in your integrations.

EDITIONS

Available in: **Developer** Edition


Package uploads and installs are available in **Group, Professional, Enterprise, Performance, Unlimited, and Developer** Editions

Creating and Uploading Patches

 **Note:** Patch versions and push upgrades are only available to [Salesforce ISV partners](#).

To create a patch version:

1. From Setup, click **Create > Packages**.
2. Click the name of your managed package.
3. Click the Patch Organization tab and then click **New**.
4. Select the package version that you want to create a patch for in the Patching Major Release drop-down list. The release type must be Managed - Released.
5. Enter a `Username` for a login to your patch organization.
6. Enter an `Email Address` associated with your login.
7. Click **Save**.


 **Note:** If you ever lose your login information, click **Reset** on the package detail page under Patch Development Organizations to reset the login to your patch development organization.

After you receive an email indicating Salesforce has created your patch development organization, you can click **Login** to begin developing your patch version.

Development in a patch development organization is restricted. The following is a list of caveats:

- New package components can't be added.
- Existing package components can't be deleted.
- API and dynamic Apex access controls can't change for the package.
- No deprecation of any Apex code.
- No new Apex class relationships, such as `extends`, can be added.
- No new Apex access modifiers, such as `virtual` or `global`, can be added.
- No new Web services can be added.
- No new feature dependencies can be added.


When you finish developing your patch, in your patch development organization:

1. Click **Create > Packages** and click the name of the package.
 2. On the Upload Package page, click **Upload**.
 3. Enter a `Version Name`. As a best practice, it's useful to have a short description and the date.
 4. Notice that the `Version Number` has had its `patchNumber` incremented.
 5. For managed packages, select a `Release Type`:
 - Choose **Managed - Released** to upload an upgradeable version. After upload, some attributes of Salesforce components are locked.
 - Choose **Managed - Beta** if you want to upload a version of your package to a small sampling of your audience for testing purposes. You'll still be able to change the components and upload additional beta versions.
-  **Note:** Beta packages can only be installed in Developer Edition or sandbox organizations, and thus can't be pushed to customer organizations.
6. Change the `Description`, if necessary.
 7. Optionally, enter and confirm a password to share the package privately with anyone who has the password. Don't enter a password if you want to make the package available to anyone on AppExchange and share your package publicly.
 8. Salesforce automatically selects the requirements it finds. In addition, select any other required components from the `Package Requirements` and `Object Requirements` sections to notify installers of any requirements for this package.
 9. Click **Upload**.

To distribute your patch, you can either share the upload link or [schedule a push upgrade](#).

Working with Patch Versions

 **Note:** Patch versions and push upgrades are only available to [Salesforce ISV partners](#).

A *patch version* enables a developer to change the functionality of existing components in a managed package, while ensuring that subscribers experience no visible changes to the package. Patches should be considered as minor upgrades to a  **Managed - Released** package and only used for fixing bugs or other errors.

Patch versions can only be created for Major Releases. Subscribers can receive patch upgrades just like any other package version. However, you can also distribute a patch by using [push upgrades](#).

When you create a patch, the *patchNumber* on a package's *Version Number* increments by one. For example, suppose you release a package with the version number 2.0. When you release a patch, the number changes to 2.0.1. This value can't be changed manually.

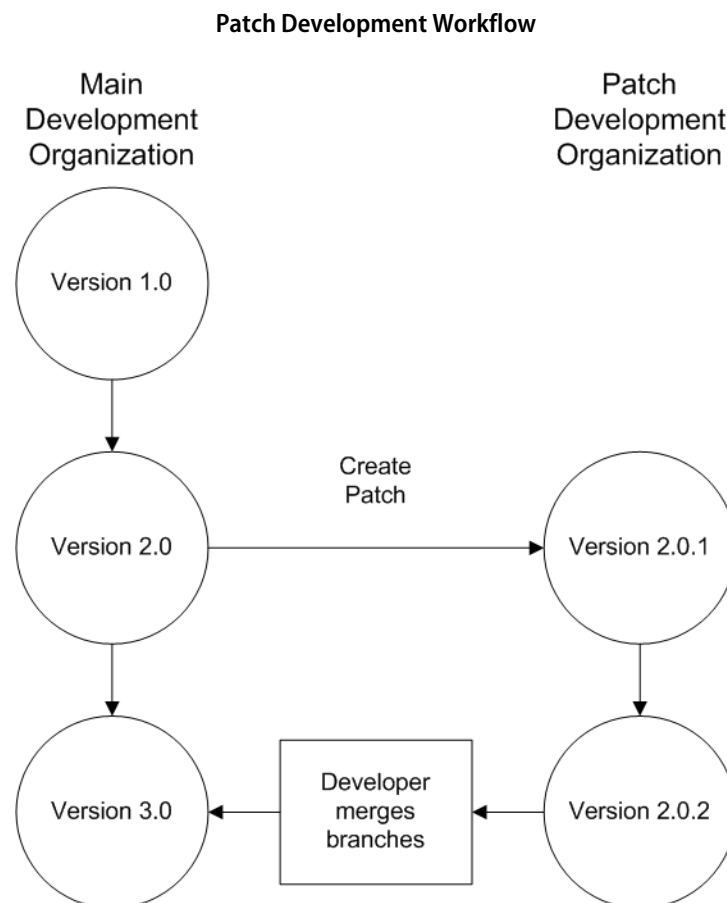
Patch Development Organizations

Every patch is developed in a *patch development organization*, which is the organization where patch versions are developed, maintained, and uploaded. To start developing a patch, you need to create a patch development organization. To do this, see [Creating and Uploading Patches](#). Patch development organizations are necessary to permit developers to make changes to existing components without causing incompatibilities between existing subscriber installations.

A patch development organization can upload an unlimited number of patches. Only one patch development organization can exist per major release of your package. Thus, a patch development organization created for a package with a version number of 4.2 can only work on patches such as 4.2.1, 4.2.2, 4.2.3, and so on, but not on version 4.1 or 4.3.

Integrating Patch Development

The following diagram illustrates the workflow of creating a patch and integrating any work into future versions:



In the diagram above, after version 2.0 is released, the developer creates a patch. The package version number in the patch development organization starts at 2.0.1. As the main development organization moves towards a released version of 3.0, a second patch is created

for 2.0.2. Finally, the developer merges the changes between the main development organization, and the patch development organization, and releases the package as version 3.0.

If you are developing your packages using the Force.com IDE, you can take advantage of revision control systems in Eclipse to compare and merge different project branches.

Salesforce recommends using the Subversion plug-in. To install Subversion for the Force.com IDE:

1. Go to <http://subclipse.tigris.org> to get the latest Eclipse Update Site URL compatible with your version of Eclipse.
2. In the Force.com IDE, navigate to **Help > Software Updates**, and select the Available Software tab. Click **Add Site**, and enter the URL from the previous step.
3. Select the new site and click **Finish** to fetch the latest version of the Subclipse plug-in. Select the required Subclipse plug-in from the list returned from the site.
4. Click **Next**, accept the terms, and click **Next** again.
5. Click **Finish** to begin the installation, and then **Install All** when prompted. You will be required to restart Eclipse once the installation completes.

You have now linked your Force.com IDE environment to Subclipse. The next step is to connect your repository to the environment:

1. Open the **SVN Repository Exploring** perspective in the IDE, which will open the SVN Repositories view.
2. Use the **Add SVN Repository** icon on the far right to configure Subclipse to access the local repository. The URL to access your repository locally is `file:///svn_repos`.

The subversion repository tracks changes made to stored projects. Because working with patches involves two different branches—a main development organization and a patch development organization—you need to combine your changes for a future release. To view the different versions of your package:

1. Open the **Project Explorer** perspective.
2. Navigate to a file in your main development project that you want to compare, and use the context menu to select **Compare With... > Branch/Tag...**
3. In the **Compare to** field, select the patch version of the file.
4. Click **Graphical**, then click **OK**.

The changes between the main development organization's file and the file stored in the patch development organization are highlighted. You can use this view to merge any differences between the two projects.

For more information on using the Force.com IDE, see the *Platform Developer's Guide*.

Versioning Apex Code

Package developers can use conditional logic in Apex classes and triggers to exhibit different behavior for different versions. This allows the package developer to continue to support existing behavior in classes and triggers in previous package versions while continuing to evolve the code.

When subscribers install multiple versions of your package and write code that references Apex classes or triggers in your package, they must specify the version that they are referencing. Within the Apex code that is being referenced in your package, you can conditionally execute different code paths based on the version setting of the calling Apex code that is making the reference. The package version setting of the calling code can be determined within the package code by calling the `System.requestVersion` method. In this way, package developers can determine the request context and specify different behavior for different versions of the package.

The following sample shows different behavior in a trigger for different package versions:

```
trigger oppValidation on Opportunity (before insert, before update) {
```

```
for (Opportunity o : Trigger.new){

    // Add a new validation to the package
    // Applies to versions of the managed package greater than 1.0
    if (System.requestVersion().compareTo(new Version(1,0)) > 0) {
        if (o.Probability >= 50 && o.Description == null) {
            o.addError('All deals over 50% require a description');
        }
    }

    // Validation applies to all versions of the managed package.
    if (o.IsWon == true && o.LeadSource == null) {
        o.addError('A lead source must be provided for all Closed Won deals');
    }
}
}
```

To compare different versions of your Apex classes, click the **Class Definition** tab when viewing the class details.

For more information about the `System.requestVersion` method, see the [Force.com Apex Code Developer's Guide](#).

Apex Deprecation Effects for Subscribers

This section demonstrates how deprecation of an Apex method affects subscribers that install the managed package. The table shows a typical sequence of actions by a package developer in the first column and actions by a subscriber in the second column. Each row in the table denotes either a package developer or subscriber action.

Package Developer Action	Subscriber Action	Notes
Create a global Apex class, <code>PackageDevClass</code> , containing a global method <code>m1</code> .		
Upload as Managed - Released version 1.0 of a package that contains <code>PackageDevClass</code> .		
	Install version 1.0 of the package.	The Version Number for the package is 1.0. The First Installed Version Number is 1.0.
	Create an Apex class, <code>SubscriberClass</code> , that references <code>m1</code> in <code>PackageDevClass</code> .	
Deprecate <code>m1</code> and create a new method, <code>m2</code> .		
Upload as Managed - Released version 2.0 of the package.		
	Install version 2.0 of the package.	The Version Number for the package is 2.0. The First Installed Version Number is still 1.0. <code>SubscriberClass</code> still


Package Developer Action	Subscriber Action	Notes
		references version 1.0 of the package and continues to function, as before.
	Edit the version settings for <code>SubscriberClass</code> to reference version 2.0 of the package. Save the class. Note an error message indicating that <code>m1</code> cannot be referenced in version 2.0 of the package.	
	Change <code>SubscriberClass</code> to reference <code>m2</code> instead of <code>m1</code> . Successfully save the class.	

Publishing Upgrades to Managed Packages

As a publisher, first ensure that your app is upgradeable by converting it to a managed package. Any changes you make to the components in a managed package are automatically included in subsequent uploads of that package, with one exception. When you upgrade a package, changes to the API access are ignored even if the developer specified them. This ensures that the administrator installing the upgrade has full control. Installers should carefully examine the changes in package access in each upgrade during installation and note all acceptable changes. Then, because those changes are ignored, the administrator should manually apply any acceptable changes after installing an upgrade. For more information, see [About API and Dynamic Apex Access in Packages](#) on page 51.

To publish upgrades to a managed package:

1. From Setup, click **Create > Packages**, and select the package from the list of packages available.
2. View the list of package components. Changes you have made to components in this package are automatically included in this list. If the changes reference additional components, those components are automatically included as well. To add new components, click **Add** to add them to the package manually.
3. Click **Upload** and upload it as usual.

 **Note:** After you upload a new version of your Managed - Released package, you can click **Deprecate** so installers cannot install an older version. Deprecation prevents new installations of older versions without affecting existing installations. For more information, see [Managing Versions](#) on page 226.

You cannot deprecate the most recent version of a managed package upload.

4. When you receive an email with the link to the upload on Force.com AppExchange, notify your installed users that the new version is ready. Use the list of installed users from the License Management Application (LMA) to distribute this information. The License Management Application (LMA) automatically stores the version number that your installers have in their organizations.

EDITIONS

Available in: **Developer Edition**

Package uploads and installs are available in **Group, Professional, Enterprise, Performance, Unlimited, and Developer Editions**

USER PERMISSIONS

To configure developer settings:

- “Customize Application”


To create packages:

- “Create AppExchange Packages”

To upload packages:

- “Upload AppExchange Packages”

Deleting Components in Managed Packages

After you've uploaded a  Managed - Released package, you may find that a component needs to be deleted from your organization. One of the following situations may occur:

- The component, once added to a package, can't be deleted.
- The component can be deleted, but can only be undeleted from the Deleted Package Components page.
- The component can be deleted, but can be undeleted from either the Deleted Package Components page or through the Recycle Bin

USER PERMISSIONS

To delete components from a package:

- "Create AppExchange Packages"

Note:

- Log a case in the [Partner Community](#) to enable Component Deletion in your packaging organization.
- Deleting Visualforce pages and global Visualforce components from a managed package requires a two-stage process, because their behavior differs from the behavior of public Apex classes and public Visualforce components. Upon package upgrade in a subscriber organization, Visualforce pages and global Visualforce components that you've deleted aren't removed. A "Delete" button or link is made available to the organization's administrators, but many organizations continue using obsolete pages and components. However, public Apex classes and public Visualforce components are deleted as part of the upgrade process. If you delete pages and components without performing this two-stage procedure, Salesforce can't warn you when your later deletion of public classes and components would break your subscribers' obsolete pages and components.

If you're deleting a Visualforce page or global Visualforce component that refers to or uses public Apex classes or public Visualforce components, perform the deletion steps in this order.

1. Stage one: Remove references.
 - i. Edit your Visualforce page or global Visualforce component to remove all references to public Apex classes or public Visualforce components.
 - ii. Upload your new package version.
 - iii. Push the stage-one upgrade to your subscribers.
2. Stage two: Delete your obsolete pages or components.
 - i. Delete your Visualforce page or global Visualforce component.
 - ii. Optionally, delete other related components and classes.
 - iii. Upload your new package version.
 - iv. Push the stage-two upgrade to your subscribers.

Here are some key types of components you can delete when updating a previously released managed package.

- Custom buttons or links
- Custom fields
- Custom objects
- Custom settings
- Custom tabs
- Field sets
- Permission sets
- Record types
- Static resources

- Validation rules
- Visualforce components
- Visualforce pages

For a complete list, see [Available Components](#) on page 21.

Deleting any component will permanently delete any data that exists in that component, delete tracked history data, and change any integrations that rely on the component, such as assignment or escalation rules. Also, once you delete a component in a managed package, you can't restore it or create another component with the same name.

No data or metadata is ever deleted in a subscriber organization without specific action by the customer. Subscribers who upgrade to the new package version will still have the deleted components available in their organization. They're displayed in the Unused Components section of the Package Details page. This ensures subscribers have the opportunity to export data and modify custom integrations involving those components, before explicitly deleting them. For example, before deleting custom objects or fields, customers can preserve a record of their data by going to Setup and clicking **Data Management > Data Export**.



Note: It's your responsibility to educate your customers about the potential impact from any components you delete. You should list all custom components you've deleted and notify customers of any actions they need to take, in the Release Notes for your upgraded package.

The following restrictions apply when deleting managed components.

- A component of any type is not deletable if it's referenced by any other metadata, such as workflow rules, validation rules, or Apex classes.
- A custom object is not deletable if it includes any of the following: Apex Sharing Reason, Apex Sharing Recalculation, Related Lookup Filter, Compact Layout, or Action.
- Deleting a custom field that is referenced by a custom report type in the same package is not recommended, as that will lead to an error when installing the upgraded package.

You can delete managed components, both declaratively, from the user interface, and programmatically, using the Metadata API. In the latter case, specify the components you want to delete in a `destructiveChanges.xml` manifest file and then use the standard `deploy()` call. The process is identical to that for deleting components that aren't managed. For more information, see the [Metadata API Developer's Guide](#).

Viewing Deleted Components

To access the Deleted Package Components page, from Setup, click **Create > Packages**, select the package the component was uploaded to, then click **View Deleted Components**. You can retrieve components from the Recycle Bin and Deleted Package Components page any time *before* uploading a new version of the package. To do this, click **Undelete** next to the component.

After a package is uploaded with a component marked for deletion, it is deleted forever.



Warning: Although a component is deleted, its **Name** remains within Salesforce. You can never create another component with the same name. The Deleted Package Components page lists which names can no longer be used.

To access the Deleted Package Components page, from Setup, click **Create > Packages**, select the package the component was uploaded to, then click **View Deleted Components**. If a component can be retrieved through the Recycle Bin, it can also be retrieved through this page. You can retrieve the following types of components from here:

- Apex classes and triggers that don't have `global` access.
- Custom tabs.
- Visualforce components with `public` access.
- Protected components, including:
 - Custom labels


- Custom links (for Home page only)
- Workflow alerts
- Workflow field updates
- Workflow outbound messages
- Workflow tasks
- Workflow flow triggers

The Process Builder has superseded flow trigger workflow actions, formerly available in a pilot program. Organizations that are using flow trigger workflow actions can continue to create and edit them, but flow trigger workflow actions aren't available for new organizations. For information on enabling the Process Builder in your organization, contact Salesforce.

- Data components, such as Documents, Dashboards, and Reports. These are the only types of components that can also be undeleted from the Recycle Bin.

You can retrieve components from the Recycle Bin and Deleted Package Components page any time *before* uploading a new version of the package. To do this, click **Undelete** next to the component.

The Deleted Components displays the following information (in alphabetical order):

Attribute	Description
Action	If the  Managed - Released package hasn't been uploaded with the component deleted, this contains an Undelete link that allows you to retrieve the component.
Available in Versions	Displays the version number of the package in which a component exists.
Name	Displays the name of the component.
Parent Object	Displays the name of the parent object a component is associated with. For example, a custom object is the parent of a custom field.
Type	Displays the type of the component.

Modifying Custom Fields after a Package is Released

The following changes are allowed to custom fields in a package, after it is released.

- The length of a text field can be increased or decreased.
- The number of digits to the left or right of the decimal point in a number field can be increased or decreased.
- A required field can be made non-required and vice-versa. If a default value was required for a field, that restriction can be removed and vice-versa.

EDITIONS


Available in:

- Developer

Managing Versions


After you upload a package to the AppExchange, you can still manage it from Salesforce. To manage your versions:

1. From Setup, click **Create > Packages**.
2. Select the package that contains the app or components you uploaded.
3. Select the version number listed in the Versions tab.
 - Click **Change Password** link to change the password option.
 - Click **Deprecate** to prevent new installations of this package while allowing existing installations to continue operating.

 **Note:** You cannot deprecate the most recent version of a managed package.

When you deprecate a package, remember to remove it from AppExchange as well. See “Removing Apps from AppExchange” in the AppExchange online help.

- Click **Undeprecate** to make a deprecated version available for installation again.

 **Note:** To create a test drive or choose a [License Management Organization \(LMO\)](#) for what you have uploaded, click **Proceed to AppExchange** from the package upload detail page.

EDITIONS

Available in:

- Group
- Professional
- Enterprise
- Performance
- Unlimited
- Developer

USER PERMISSIONS

To upload packages:


- “Upload AppExchange Packages”

Pushing an Upgrade

A *push upgrade* is a method of automatically upgrading your customers to a newer version of your package. This feature works with managed packages only and can be used to ensure that all of your customers are on the same or latest version of your package. You can push an upgrade to any number of organizations that have installed your managed package.

A package subscriber doesn't need to do anything to receive the push upgrade. The only indication a subscriber receives after a successful push upgrade is that the package's `Version Number` on the Package Detail page has a higher value. Any upgrades that fail must be resolved by the developer initiating the push.

Push upgrades minimize the potential risks and support costs of having multiple subscribers running different versions of your app. You can also automate many post-upgrade configuration steps, further simplifying the upgrade process for your customers.

 **Note:** This feature is available to eligible Salesforce partners. For more information on the Partner Program, including eligibility requirements, please visit us at www.salesforce.com/partners.


About Push Upgrades

 **Note:** Registered ISV partners can request Push Major Upgrade functionality by logging a case in the [Partner Community](#).

You can push either a patch or a major upgrade. A patch only contains bug fixes and minor enhancements. In contrast, a major upgrade can include major enhancements and new features that add new components. At a high level, pushing an upgrade involves the following steps:

- Upgrade your managed package installed in a customer organization from version X to version Y
- Select one, many, or all customer organizations to upgrade and select a particular version to upgrade to
- Schedule the upgrade to start at a particular date and time
- View progress of upgrades, abort upgrades in progress, or view the result of a push upgrade

- In conjunction with push, you may use a post install Apex Script to automate many post-upgrade configurations that your customers might have previously performed manually

 **Warning:** When you push an upgrade, you're making changes to a subscriber's organization without their explicit consent. Hence, it's important to plan ahead and exercise due caution.

Pushing a major upgrade entails a higher degree of risk as it can impact existing functionality in a subscriber's organization. This is because new components in the upgraded package might not be available to existing users of the package, or could overwrite users' customizations. As the app developer, it's your responsibility to protect users from any adverse impact due to upgrading. We strongly recommend you consider all potential consequences of the upgrade and take appropriate steps to prevent any problems.

When pushing a major upgrade, we recommend that you divide changes in your package into two categories:

1. Enhancements to existing features that users already have access to—Use a post install Apex script to automatically assign the relevant components to existing users. This ensures all current users of the package can continue using it without explicit action by administrators.
2. New features you're introducing for the first time—Don't use a post install Apex script to auto-assign components. This ensures your subscribers have the opportunity to decide if and when to use the new features.

Here are some additional guidelines to keep in mind when planning a push upgrade.

- Avoid changes to validation rules, formula fields, and errors thrown from Apex triggers, as they may negatively impact subscribers' integrations.
- Don't make visible changes to a package in a patch. This is because other than a change in the package version number, subscribers aren't notified of push upgrades.
- Test your upgraded package in multiple environments, replicating all relevant features of your customers' organizations, including editions, customizations, other installed packages, and permission sets.
- Schedule push upgrades at your customers' off-peak hours and outside of Salesforce's major release windows, to minimize potential subscriber impact.
- Notify your subscribers in advance about the timing of the upgrade, its potential consequences, and any steps they need to take.

Push Upgrade Best Practices

Push Upgrade is one of the most powerful features we provide to our partners. You have the power to upgrade your customers, but it's imperative that you use that power carefully. Pushing an upgrade without proper planning and preparation can result in significant customer satisfaction issues. Hence, we strongly recommend that you adhere to the best practices documented here.

Plan, Test, and Communicate

- Communicate, communicate, and communicate! Your customers might not even know about the Push Upgrade feature. Some might have strong reservations about changes being pushed to their organizations. Reach out to them and explain how the cloud-computing model works, how they can benefit from seamless upgrades, how you are using best practices to ensure a smooth upgrade, and what your process and commitment is to them regarding the timing and content of an upgrade. Timely and thorough communication is critical for the success of this program.
- Share an upgrade timeline plan with your customers so they know when you will upgrade, and how often.
- Plan when you want to push upgrades to your customers' organizations. Keep in mind that most customers don't want changes around their month-end, quarter-end, and year-end or audit cycles. Do your customers have other critical time periods when they don't want any changes to their organization? For example, there might be certain times when they don't have staff available to verify changes or perform any required post-installation steps.
- Schedule push upgrades during your customers' off-peak hours, such as late evening and night. Have you considered time zone issues? Do you have customers outside the United States who have different off-peak hours? You can schedule push upgrades to

any number of customer organizations at a time. Consider grouping organizations by time zone, if business hours vary widely across your customer base.

- Don't schedule push upgrades close to Salesforce-planned maintenance windows. In most cases, it might be better to wait 3-4 weeks after a major Salesforce release before you push major upgrades.
- Test, test, and test! Since you're pushing changes to the organization instead of the customer pulling in changes, there is a higher bar to ensure the new version of your app works well in all customer configurations.

Stagger the Push

- Don't push changes to all customers at once. It's important to ensure that you have sufficient resources to handle support cases if there are issues. Also, it's important that you discover possible issues before your entire customer base is affected.
- Push to your own test organizations first to confirm that the push happens seamlessly. Log in to your test organization after the push upgrade and test to see that everything works as expected.
- When applicable, push to the sandbox organizations of your customers first before pushing to their production organizations. Give them a week or more to test, validate, and fix in the sandbox environment before you push to their production organizations.
- Push upgrades to small batches of customer production organizations initially. For example, if you have 1,000 customers, push upgrades to 50 or 100 customers at a time, at least the first few times. Once you have confidence in the results, you can upgrade customers in larger batches.

Focus on Customer Trust

- You're responsible for ensuring that your customers' organizations are not adversely affected by your upgrade. Avoid making changes to the package, such as changes to validation rules or formula fields, that might break external integrations made by the customer. If for some reason you do, test and communicate well in advance. Please keep in mind that you can impact customer data, not just metadata, by pushing an upgrade that has bugs.
- Write an Apex test on install to do basic sanity testing to confirm that the upgraded app works as expected.
- If you're enhancing an existing feature, use a post-install script to automatically assign new components to existing users using permission sets.
- If you're adding a new feature, don't auto-assign the feature to existing users. Communicate and work with the administrators of the customer organization so they can determine who should have access to the new feature, and the timing of the roll-out.

Assigning Access to New Components and Fields

If the new version of your package includes new components or new fields in existing components, existing users of the package won't automatically have access to the new components and fields after the upgrade. This can limit them from using the new features you've added or prevent older features from working properly. By default, any new components in your package are assigned only to administrators. You have two options for ensuring that all users of the package have access to the new components and fields.

Notify administrators to assign the appropriate permissions to all users of the package

We recommend this for any new features you're introducing. This ensures administrators have the option of deciding if and when to make the new features available.

Assign the new components to existing users automatically, using a post install Apex script

We recommend this for enhancements to existing features. This ensures all current users of the package can continue using it without explicit action by administrators.

To assign access to new components automatically, you can use the following strategy.

1. Create new permission sets that define the default access settings for all new components and fields.

2. Include the new permission sets in the new package version.
3. Write a post install Apex script to run automatically in the subscriber organization after the package is upgraded. The script must perform these tasks.
 - a. For each new permission set, choose an existing component whose user assignment needs to be copied.
 - b. Find all profiles that can access that component.
 - c. Assign the new permission sets to every user with those profiles.



Note: The default permission sets for all standard profiles aren't editable. Hence, the post install script will trigger an exception if it tries to update one of these permission sets. It's important that you create a new permission set to assign access to the new components in your package.

Sample Post Install Script for a Push Upgrade

This section shows a sample post install script that automates the assignment of new components to existing users of a package. For more information on writing a post install Apex script, see [Running Apex on Package Install/Upgrade](#) on page 108.

The sample script covers a scenario in which an ISV is upgrading subscribers to a new package version that contains new Visualforce pages and a new permission set that grants access to those pages. After upgrading, existing users of the package will not have access to the new pages by default. The post install script resolves this problem by identifying which users have access to the Visualforce pages in the old version of the package and granting them access to the new pages. The script performs the following actions.

- Get the Id of the Visualforce pages in the old version of the package
- Get the permission sets that have access to those pages
- Get the list of profiles associated with those permission sets
- Get the list of users who have those profiles assigned
- Assign the permission set in the new package to those users

```
global class PostInstallClass implements InstallHandler {
    global void onInstall(InstallContext context) {

        //Get the Id of the Visualforce pages
        List<ApexPage> pagesList = [SELECT Id FROM ApexPage WHERE NamespacePrefix =
        'TestPackage' AND Name = 'vfpagel'];

        //Get the permission sets that have access to those pages
        List<SetupEntityAccess> setupEntityAccessList = [SELECT Id,
        ParentId, SetupEntityId, SetupEntityType FROM SetupEntityAccess
        where SetupEntityId IN:pagesList];
        Set<ID> PermissionSetList = new Set<ID> ();

        for(SetupEntityAccess sea : setupEntityAccessList){
            PermissionSetList.add(sea.ParentId);
        }
        List<PermissionSet> PermissionSetWithProfileIdList =
        [SELECT id,Name,IsOwnedByProfile,Profile.Name,
        ProfileId FROM PermissionSet where IsOwnedByProfile = true
        AND Id IN :PermissionSetList ];

        //Get the list of profiles associated with those permission sets
        Set<ID> ProfileList = new Set<ID> ();
        for(PermissionSet per : PermissionSetWithProfileIdList){
            ProfileList.add(per.ProfileId);
        }
    }
}
```

```

    }
    //Get the list of users who have those profiles assigned
    List<User> UserList  =[SELECT id FROM User where ProfileId  IN :ProfileList ];

    //Assign the permission set in the new package to those users
    List<PermissionSet> PermissionSetToAssignList = [SELECT id,Name
        FROM PermissionSet where Name='TestPermSet' AND
        NamespacePrefix = 'TestPackage'];
    PermissionSet PermissionSetToAssign = PermissionSetToAssignList[0];

    Set<ID> UsersSet = new Set<ID> ();
    for(User us : UserList){
        PermissionSetAssignment psa= new PermissionSetAssignment();
        psa.PermissionSetId = PermissionSetToAssign.id;
        psa.AssigneeId = us.id;
        UsersSet.add(us.id);
    }
}

```

```

// Test for the post install class
@Test
private class PostInstallClassTest {
    @Test
    public static void test() {
        PostInstallClass myClass = new PostInstallClass();
        Test.testInstall(myClass, null);
    }
}

```

Known Limitations for Push Upgrade

The following are known limitations when pushing an upgrade.

- Settings for tab visibility are not carried over by permission sets. Hence, if your upgrade requires changes to tab settings, you can't implement the changes automatically using a post install script. Instead, you'll need to notify administrators to make the change manually.
- Push upgrade fails if the package contains Visualforce markup that attempts to access the `OpenActivity` entity. For example, the following code in the package will cause the push upgrade to fail.

```

<apex:outputLabel for="taskDate"
value="{!$ObjectType.OpenActivity.fields.ActivityDate.label}" />

```

As a workaround, avoid using the global `ObjectType` reference to obtain the label for `OpenActivity.ActivityDate` by specifying the label as follows:

```


<apex:outputLabel for="taskDate" value="Date" />

```

Scheduling Push Upgrades

 **Note:** Patch versions and push upgrades are only available to [Salesforce ISV partners](#).

After you've created a "patch version" in the Salesforce Help of your package, you can automatically deploy it to customers using a push upgrade.

 **Tip:** Salesforce strongly recommends following this sequence for pushing package upgrades.

1. Push the upgrade to your own organizations so you can run tests and fix any bugs before upgrading subscribers.
2. When you're ready and have coordinated with your customers on their change management process, push to a small number of customer organizations. Try sandbox organizations first, if possible.
3. Once you're comfortable with the initial results, push to your wider customer base, based on your agreements with each customer.
4. Deprecate the previous version of your package in your main development organization. Replace the version on AppExchange if necessary, and update your [Trialforce](#) setup.
5. If your upgrade was a patch, after you've successfully distributed the upgrade to subscriber organizations, reintegrate those changes into your main development organization. For more information about combining patches in the main development organization, see [Working with Patch Versions](#) on page 218.

Read "Best Practices for Push Upgrades and Patch Versions" in the Salesforce Help for more information.

To schedule a push upgrade:

1. Log in to your main development organization (not the patch organization you used to upload the new version).
2. From Setup, click **Create > Packages** and click the name of the managed package whose upgrade you want to push. On the package detail page, click the **Versions** tab, and then click **Push Upgrades**.
3. Click **Schedule Push Upgrades**.
4. Select a package version to push from the **Patch Version** drop-down list.

 **Note:** Beta versions aren't eligible for push.

5. Enter a **Scheduled Start Date**, indicating when a push upgrade should begin.
6. In the Select Target Organizations section, select the organizations to receive your push upgrade. If an organization already received a push upgrade for the selected package version, it won't appear in this list. You can select organizations by:
 - Entering a term that filters based on an organization's name or ID. Names can match by partial string, but IDs must match exactly.
 - Choosing between production and sandbox organizations from the **Organizations** drop-down list.
 - Choosing organizations that have already installed a particular version.
 - Clicking on individual organizations, or the **Select All** and **Deselect All** checkboxes.

This section lists the following information about the organization (in alphabetical order):

Field	Description
Current Version	The current package version an organization has installed.

EDITIONS

Available in:

- Developer

USER PERMISSIONS

To push an upgrade:

- "Upload AppExchange Packages"

Field	Description
Organization ID	The ID that uniquely identifies the organization to Salesforce.
Organization Name	The name of the organization. Clicking this name shows the upgrade history for the organization .
Primary Contact	The name of the contact who installed the package.

7. Click **Schedule**.

While a push upgrade is in progress, you can click **Abort** to stop it.

On the previous push upgrades page, a table lists recently scheduled push upgrades for the package.

Viewing Push Upgrade Details



Note: Patch versions and push upgrades are only available to [Salesforce ISV partners](#).

For information about a specific push upgrade that your organization sent, from Setup, click **Create > Packages**, click the name of the package you want to view, then click **Push Upgrades**. Clicking on the name of a **Target** takes you to the Push Upgrade Details page, which has information both for the push job and each organization that it was pushed to.

The Job Details section has the following information about the overall push upgrade (in alphabetical order):

Field	Description
End Date	The date and time the push upgrade finished.
Ignore Apex Test Failures	Whether Apex test failures that may cause the installed application not to function properly were ignored.
Scheduled By	The name of the user who initiated the push upgrade.
Start Date	The scheduled start date and time of the push upgrade.
Status	The status of the push upgrade, whether scheduled, in progress, completed, aborted, or completed with failures.
Version	The package version number that was pushed.

EDITIONS

Available in:

- Developer

USER PERMISSIONS

To view push upgrade details:

- "Upload AppExchange Packages"

In the Organizations section, you can get a list of all the organizations that received a push upgrade. You can filter organizations by using the search box and entering a term that filters based on an organization's name or ID. Names can match by partial string, but IDs must match exactly. From the drop-down list, you can also filter based on the status of the push upgrade.

The list contains the following information specific to each organization (in alphabetical order):

Field	Description
Duration	The amount of time the push upgrade took.
Failure Type	Lists the type of failure that occurred (if any). If the push upgrade did fail, a possible explanation is provided in the collapsible section. If the push upgrade was unsuccessful, click Retry to try it again.
Organization ID	The ID that uniquely identifies the organization to Salesforce.
Organization Name	The name of the organization. Clicking this name shows the upgrade history for the organization .
Start	The scheduled start date and time of the push upgrade.
Status	The status of the push upgrade, whether scheduled, in progress, completed, aborted, or completed with failures.

Viewing an Organization's Upgrade History



Note: Patch versions and push upgrades are only available to [Salesforce ISV partners](#).

For more information about a specific organization that received a push upgrade, from Setup, click **Create > Packages**, click the name of the package you want to view, then click on the name of a Target. Clicking on an organization in the target list provides the following details (in alphabetical order):

Field	Description
Current Version	The current package version an organization has installed.
Organization ID	The ID that uniquely identifies the organization to Salesforce.
Organization Name	The name of the organization.
Primary Contact	The name of the contact who installed the package.
Primary Contact Email	The email address of the package publisher.
Status	The status of the push upgrade, whether scheduled, in progress, completed, aborted, or completed with failures.

The Push Upgrade History lists the following information (in alphabetical order):

Field	Description
Action	Clicking View Details returns you to the job details for that upgrade.

EDITIONS

Available in:

- Developer

USER PERMISSIONS

To view push upgrade history:

- "Upload AppExchange Packages"

Field	Description
Start Date	The scheduled start date and time of the push upgrade.
Status	The status of the push upgrade, whether scheduled, in progress, completed, aborted, or completed with failures.
Version	The package version number that was pushed.

APPENDICES

APPENDIX A ISVforce User License Comparison


Introduction

The following tables compare object access, user permissions and features, and organization limits for these license types.

- Force.com Administrator—A standard Salesforce license with complete customization capabilities. It prohibits Create, Read, Update, and Delete on Leads, Opportunities, Products, Cases, Solutions, and Campaigns.
- Force.com—A standard Salesforce Platform license with access to Accounts, Contacts, and custom objects. Used by non-administrators.

 **Note:** For a complete list of license types, see: https://help.salesforce.com/HTViewHelpDoc?id=users_license_types_available.htm

The following symbols are used in the tables.

-  —Included in license
- \$—Available as an add-on for an additional fee
- C—Create access to the object
- R—Read access to the object
- U—Update access to the object
- D—Delete access to the object

Object Accessed

Object Accessed	Force.com Administrator		Force.com	
	EE	UE/PXE	EE	UE/PXE
Accounts	CRUD	CRUD	CRUD	CRUD
Activities, Tasks	CRUD	CRUD	CRUD	CRUD
Assets				
Calendar, Events	CRUD	CRUD	CRUD	CRUD
Campaigns				
Cases				
Contacts	CRUD	CRUD	CRUD	CRUD
Content	CRUD	CRUD	CRUD	CRUD
Contracts				

Object Accessed	Force.com Administrator		Force.com	
	EE	UE/PXE	EE	UE/PXE
Documents	CRUD	CRUD	CRUD	CRUD
Entitlements				
Ideas	CRUD	CRUD	CR	CR
Knowledge	R	R		
Leads				
Opportunities				
Products & Price Books				
Questions and Answers	CRUD	CRUD		
Quotes				
Service Contracts				
Solutions				

User Features

User Features	Force.com Administrator		Force.com	
	EE	UE/PXE	EE	UE/PXE
Company Community	\$	\$	\$	\$
Content	✓	✓	✓	✓
Jigsaw Exports	\$	\$	\$	\$
Knowledge	\$	\$	\$	\$
Send Mass Email	✓	✓	✓	✓
Mobile (Full)	\$	✓	\$	✓
Offline	✓	✓	✓	✓
Siteforce Contributor	\$	\$	\$	\$
Siteforce Publisher	\$	\$	\$	\$
Visual Workflow	✓	✓	✓	✓

User Permissions

User Permissions	Force.com Administrator		Force.com	
	EE	UE/PXE	EE	UE/PXE
Customize Reports	✓	✓	✓	✓
Customize Dashboards	✓	✓	✓	✓
View Dashboards*	✓	✓	✓	✓
Chatter (Groups, Files, Profiles)	✓	✓	✓	✓
Create Workflow and Approval Process	✓	✓		
Manage Users and Profiles	✓	✓		
Identity	✓	✓	✓	✓
Identity Connect	\$	\$	\$	\$
Write Apex Code	✓	✓		
Work.com	\$	\$	\$	\$
Custom Apps Limit	10	Unlimited	10	Unlimited
Custom Tabs Limit	25	Unlimited	25	Unlimited
Custom Objects Limit**	200	2,000	200	2,000

* The running user of a dashboard must be a Force.com or a Force.com One App user to view the dashboard. Dashboards using the Force.com administrator as the running user are not viewable by other Force.com license types.

** Restricted limit for Force.com One App and Chatter Plus.

Additional Organization Limits

Additional Organization Limits (Added Per User)	Force.com Administrator		Force.com	
	EE	UE/PXE	EE	UE/PXE
Data Storage	20 MB	120 MB	20 MB	120 MB
File Storage	2 GB	2 GB	2 GB	2 GB
API Calls (Per Day Per User)	1,000	5,000	1,000	5,000

ISVforce User License Comparison

For data storage, Enterprise, Performance, and Unlimited Editions are allocated either 1 GB or a per-user limit, whichever is greater. For example, an Enterprise Edition organization with 10 users receives 1 GB because 10 users multiplied by 20 MB per user is 200 MB, which is less than the 1 GB minimum. An Enterprise Edition organization with 100 users receives more than the 1 GB minimum because 100 users multiplied by 20 MB per user is 2,000 MB.

For file storage, Enterprise, Performance, and Unlimited Editions are allocated a per-user limit multiplied by the number of users in the organization plus an additional per-organization allocation of 11 GB. For example, an Enterprise Edition organization with 600 users receives 1,211 GB of file storage, or 2 GB per user multiplied by 600 users plus an additional 11 GB.



Note: For a complete list of storage limits for each edition, see:

https://help.salesforce.com/HTViewHelpDoc?id=limits_storage_allocation.htm

APPENDIX B OEM User License Comparison

Introduction

The following tables compare object access, user permissions and features, and organization limits for these license types.


- OEM Embedded—A full Force.com license with contractual restrictions. It prohibits Create, Read, Update, and Delete on Leads, Opportunities, Cases, Solutions, and Campaigns.
- Customer Community—The Customer Community license is similar to a High Volume Customer Portal license and is well suited for business-to-consumer communities with a large number of external users.*
- Partner Community—The Partner Community license is similar to a Gold Partner license and is well suited for business-to-business communities, such as a partner community.*
- ISV Portal—An Authenticated Website license with basic data sharing options (manual sharing to user and participation in sharing groups is not permitted). Users can only log in via Force.com Sites. This is best used when projected user volumes will exceed 100,000.*
- ISV Portal with Sharing—A Customer Portal Manage Custom license with full sharing capabilities. User can only log in via Force.com Sites. This is best used when projected user volumes are under 100,000 and granular security access is required.*

* Licenses can be assigned to external users only.

End users can't develop applications or extend applications by creating custom objects, but they can have access to additional applications as long as those applications are sold with an embedded license.

Community and Portal licenses sold by a partner can only be used to access the partner's application. ISV Portal and ISV Portal with Sharing licenses are only available to partners who already sell Portals.

The following symbols are used in the tables:

-  —Included in license
- \$—Available as an add-on for a fee
- C—Create access to the object
- R—Read access to the object
- U—Update access to the object
- D—Delete access to the object

Objects

Object Accessed	OEM Embedded	ISV Portal	ISV Portal with Sharing	Customer Community	Partner Community
Accounts	CRUD		CRU	R	CRUD

OEM User License Comparison

Object Accessed	OEM Embedded	ISV Portal	ISV Portal with Sharing	Customer Community	Partner Community
Activities, Tasks	CRUD			R	CRUD
Calendar, Events	CRUD				CRUD
Contacts	CRUD		CRU	R	CRUD
Content	CRUD		R		CRUD
Contracts*	CRUD	CRU	CRU	CRUD	CRUD
Documents	CRUD	R	R	R	R
Ideas	CR	CR	CR	CR	CR
Orders*	CRUD	CRU	CRU	CRUD	CRUD
Products & Price Books*	CRUD	CRU	CRU	R	R
ISV Custom Object	CRUD	CRUD	CRUD	CRUD	CRUD

* With the Orders Platform permission set license (PSL), available to OEM partners only, administrators can give users with Force.com user licenses access to Contracts, Products, Price Books, and Orders. Orders functionality is automatically available to all licenses except the Force.com licenses, which explicitly require the new PSL to grant access.

User Features

User Feature	OEM Embedded	ISV Portal	ISV Portal with Sharing	Customer Community	Partner Community
Knowledge	\$		R	R	R
Send Mass Email	✓				
Salesforce1 Mobile App	✓	✓	✓	✓	✓
Identity	✓	✓	✓	✓	✓
Visual Workflow	✓	✓	✓	✓	✓
Territory Management	✓		✓		✓

User Permissions

User Permission	OEM Embedded	ISV Portal	ISV Portal with Sharing	Customer Community	Partner Community
Create and Customize Reports	✓				
View Reports	✓		✓		✓
Create and Customize Dashboards	✓				
View Dashboards*	✓				✓
Enhanced User/Role Based Sharing	✓		✓		✓
Identity	✓	✓	✓	✓	✓
Identity Connect					
Chatter (Groups, Files, Profiles)	✓			✓	✓
Submit Workflow Approvals	✓		✓	✓	✓
Work.com	\$	\$	\$	\$	\$
Custom Apps Limit	1				
Custom Tabs Limit	25	25	25	25	25
Custom Objects Limit	400**	200	200	200	200

* The running user of a dashboard must be a Force.com user to view the dashboard. Dashboards using the Force.com administrator as the running user are not viewable by other Force.com license types.

** The limit of 400 custom objects applies to the primary app offering. Subscribers cannot create their own custom objects.

Storage Limits

Additional Organization Limits (Added Per User)	OEM Embedded	ISV Portal	ISV Portal with Sharing	Customer Community	Partner Community
Data Storage	20 MB	0	2 MB	0	5 MB
File Storage	2 GB	0	0	0	0

For data storage, each OEM Embedded organization is allocated either 1 GB or a per-user limit, whichever is greater. For example, an OEM Embedded organization with 20 users receives 1 GB because 20 users multiplied by 20 MB per user is 400 MB, which is less than

OEM User License Comparison

the 1 GB minimum. An OEM Embedded organization with 100 users receives more than the 1 GB minimum because 100 users multiplied by 20 MB per user is 2 GB.

For file storage, each OEM Embedded organization is allocated a per-user limit multiplied by the number of users in the organization plus a per organization allocation of 11 GB. For example, an OEM Embedded organization with 600 users receives 1,211 GB of file storage, or 2 GB per user multiplied by 600 users plus 11 GB.

Salesforce Edition	Data Storage Minimum per Organization	File Storage Minimum per Organization	Storage Allocation Per User License
OEM Embedded	1 GB, plus 5 MB for each Gold Partner license	11 GB	20 MB of data storage and 2 GB of file storage

API Limits

The following table lists the limits for the total API requests (calls) per 24-hour period for an OEM Embedded organization.

Salesforce Edition	API Calls Per License Type	Minimum	Maximum
OEM Embedded	1000	5,000	1,000,000

Limits are enforced against the aggregate of all API calls made by the organization in a 24 hour period; limits are not on a per-user basis. When an organization exceeds a limit, all users in the organization may be temporarily blocked from making additional calls. Calls will be blocked until usage for the preceding 24 hours drops below the limit.

GLOSSARY

The following terms and definitions describe key application and packaging concepts and capabilities:

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 Call Center. 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 Force.com platform.

Beta, Managed Package

In the context of managed packages, a beta managed package is an early version of a managed package distributed to a sampling of your intended audience to test it.

Deploy

To move functionality from an inactive state to active. For example, when developing new features in the Salesforce user interface, you must select the “Deployed” option to make the functionality visible to other users.

The process by which an application or other functionality is moved from development to production.

To move metadata components from a local file system to a Salesforce organization.

For installed apps, deployment makes any custom objects in the app available to users in your organization. Before a custom object is deployed, it is only available to administrators and any users with the “Customize Application” permission.

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>.

Major Release

A significant release of a package. During these releases, the major and minor numbers of a package version increase to any chosen value.

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.

EDITIONS

Available in:

- Group
- Professional
- Enterprise
- Performance
- Unlimited
- Developer

Managed Package Extension

Any package, component, or set of components that adds to the functionality of a managed package. You cannot install an extension before installing its managed package.

Namespace Prefix

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 namespace prefix must be globally unique across all Salesforce organizations. It keeps your managed package under your control exclusively.

Package

A group of Force.com 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.

Package Dependency

This is created when one component references another component, permission, or preference that is required for the component to be valid. Components can include but are not limited to:

- Standard or custom fields
- Standard or custom objects
- Visualforce pages
- Apex code

Permissions and preferences can include but are not limited to:

- Divisions
- Multicurrency
- Record types

Package Installation

Installation incorporates the contents of a package into your Salesforce organization. A package on the AppExchange can include an app, a component, or a combination of the two. After you install a package, you may need to deploy components in the package to make it generally available to the users in your organization.

Package Version

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.

Unmanaged packages are not upgradeable, so each package version is simply a set of components for distribution. A package version has more significance for managed packages. Packages can exhibit different behavior for different versions. Publishers can use package versions to evolve the components in their managed packages gracefully by releasing subsequent package versions without breaking existing customer integrations using the package. See also Patch and Patch Development Organization.

Patch

A patch enables a developer to change the functionality of existing components in a managed package, while ensuring subscribing organizations that there are no visible behavior changes to the package. For example, you can add new variables or change the body of an Apex class, but you may not add, deprecate, or remove any of its methods. Patches are tracked by a *patchNumber* appended to every package version. See also Patch Development Organization and Package Version.

Patch Development Organization

The organization where patch versions are developed, maintained, and uploaded. Patch development organizations are created automatically for a developer organization when they request to create a patch. See also Patch and Package Version.

Patch Release

A minor upgrade to a managed package. During these releases, the patch number of a package version increments.

Publisher

The publisher of an AppExchange listing is the Salesforce user or organization that published the listing.

Push Upgrade

A method of delivering updates that sends upgrades of an installed managed package to all organizations that have installed the package.

Subscriber

The subscriber of a package is a Salesforce user with an installed package in their Salesforce organization.

Test Drive

A test drive is a fully functional Salesforce organization that contains an app and any sample records added by the publisher for a particular package. It allows users on AppExchange to experience an app as a read-only user using a familiar Salesforce interface.

Unmanaged Package

A package that cannot be upgraded or controlled by its developer.

Upgrading

Upgrading a package is the process of installing a newer version. Salesforce supports upgrades for managed packages that are not beta.

Uploading

Uploading a package in Salesforce provides an installation URL so other users can install it. Uploading also makes your packaged available to be published on AppExchange.

INDEX

A

- access control in Connected App [75, 78, 80](#)
- Apex
 - access from packages [51](#)
 - behavior in packages [220](#)
 - deprecation effects [221](#)
 - editing access from AppExchange packages [54](#)
- API
 - access from packages [51](#)
 - default package versions [55](#)
 - downloading enterprise WSDL [57](#)
 - editing access from AppExchange packages [54](#)
- API token
 - requesting [139](#)
- APO
 - associate with partner account [125](#)
 - create provider profile [125](#)
- app
 - installation options [126](#)
 - patch (version) updates [138](#)
 - publish [124](#)
 - search optimization [138](#)
 - sell [124](#)
 - submit for security approval [127](#)
- AppExchange
 - beta packages [96](#)
 - branding [183–184](#)
 - creating packages [20, 108, 110–113, 225, 228, 230](#)
 - custom help [51](#)
 - deleting components [224](#)
 - designing for [48](#)
 - developing app documentation [51](#)
 - editing package API access [54](#)
 - email branding [185–186](#)
 - managed package release types [217](#)
 - managed package versions [217](#)
 - managed packages [94](#)
 - managing uploads [226](#)
 - package API access [51](#)
 - package details [100](#)
 - providing a free trial [187](#)
- AppExchange Checkout [129–133](#)
- AppExchange trials
 - difference from Trialforce [204](#)

- Apps
 - uploading [21, 98](#)
- Attributes [18](#)

B

- Best practices [176](#)
- Beta packages
 - uninstalling [105](#)
 - uploading [96](#)
- Branding [126](#)

C

- Channel Order App [141–142](#)
- Chatter in packages [49](#)
- Checkout [129–133](#)
- company name [137](#)
- Components [18](#)
- Configuration [142, 145, 147–149, 167](#)
- Connected App
 - access control in [75, 78, 80](#)
 - create [64](#)
 - creating [66](#)
 - deleting [72](#)
 - details [75](#)
 - editing [72, 75, 78, 80](#)
 - installing [74](#)
 - IP restrictions for [75, 78, 80](#)
 - managing [76](#)
 - monitoring usage [81](#)
 - packaging [72](#)
 - start URL [75, 78](#)
 - uninstalling [82](#)
- Contract Terms [147, 151–152](#)
- creating a Connected App [64, 66](#)
- Creating packages [21, 98](#)
- Creating patches [217](#)
- creating signups for OAuth and API access [202](#)
- creating signups using the API [192](#)
- Credentials [147](#)
- Custom Fields [148](#)
- Custom help
 - AppExchange apps [51](#)
 - style guide [51](#)
- Custom Profiles
 - creating [48](#)

D

- data file [209](#)
- Defaults
 - setting [168](#)
- deleting a Connected App [72](#)
- Dependencies [46](#)
- Deployment [105](#)
- Designing
 - matching Salesforce look-and-feel [50](#)
- Developer settings
 - configuring [94](#)
 - license manager [96](#)
 - namespace prefix [95](#)
- Developing
 - partner WSDL [56](#)
 - unmanaged packages [20](#)
- Development lifecycle [2](#)
- Domain name
 - setup overview [85](#)
- Dynamic Apex
 - supporting multiple editions [63](#)

E

- editing a Connected App [72](#), [75](#), [78](#), [80](#)
- Editions [3](#), [166](#)
- Editions supported [164](#)
- Email Service [145](#)
- Entity Relationships [164](#)
- Environment Hub
 - adding linked user [90](#), [92](#)
 - connect organization [87](#)
 - disabling single sign-on [92](#)
 - edit organization details [89](#)
 - enabling single sign-on [91](#)
 - view organization details [87](#)
- Extending packages [62](#)
- External services
 - provisioning [57](#)
 - working with [57](#)

F

- FAQ
 - LMA [177](#)
- feedback [140](#)
- free trial
 - create [119](#), [140](#)
 - vs test drive [140](#)

G

- Group edition
 - access control [60](#)
 - accessing REST API [62](#)
 - limits [60](#)
 - packages [58–59](#), [61–62](#)
 - using Apex [60](#)

I

- ideas [140](#)
- industries [137](#)
- Installation [142](#), [166](#)
- installing a Connected App [74](#)
- Installing packages [103](#)
- Integration
 - default package versions [55](#)
 - downloading enterprise WSDL [57](#)
 - managed packages [55](#), [57](#)
- Intellectual property
 - protecting [49](#)
- introduction [208](#)
- IP ranges with Connected App [72](#)
- IP restrictions for Connected App [75](#), [78](#), [80](#)

L

- leads
 - source codes [133–135](#)
 - vs license records [134](#)
- Leads
 - editing lead manager [171](#)
- License Management
 - overview [164](#)
- License Management App
 - terminology [165](#)
 - use [169](#)
- License Management Organization
 - associating a managed package [168](#)
- License Owner
 - editing [174](#)
- licenses
 - choose settings [127](#)
 - vs lead settings [134](#)
- Licenses
 - details [172](#)
 - editable fields [174](#)
 - editing [174](#)
 - editing owner [174](#)
- listing
 - add a test drive [132](#)

listing (*continued*)

add categories [128](#)

analytics [135](#)

delisted by Salesforce [140](#)

submit for security approval [127](#)

Login [206–207](#)

Logo [126](#)

M

Manage Users [149](#)

managed package

change [138](#)

change a listing [139](#)

register [127](#)

Managed packages

about [94](#)

beta [96](#)

component availability [105](#)

component behavior [27, 36](#)

default package versions [55](#)

downloading enterprise WSDL [57](#)

extensions [113](#)

group edition [58–62](#)

limits for group edition [60](#)

limits for professional edition [60](#)

packageable components [21, 25, 34](#)

patch version [218](#)

planning [19](#)

professional edition [58–62](#)

protected components [45](#)

publishing upgrades [222](#)

push upgrades [226–227, 229, 231](#)

release types [217](#)

status [18](#)

supporting multiple editions [62–63](#)

upgrading [215](#)

versions [217](#)

managing a Connected App [76](#)

managing license agreements [163](#)

Marketing [175](#)

MetricsDataFile object [210](#)

monitoring usage of a Connected App [81](#)

O

Objects

MetricsDataFile [210](#)

SignupRequest [193](#)

Operational scope [46](#)

Order detail page [148](#)

Orders [154–155](#)

Overview [164](#)

P

Package

LMO association [168](#)

Package API access [51](#)

Package installation [103](#)

Package versions

behavior versioning Apex [220](#)

deprecating Apex [221](#)

Package Versions

details [171](#)

Packages

about [1–2, 18](#)

branding [183–186](#)

Chatter [49](#)

component availability [105](#)

component behavior [27, 36](#)

creating [20–21, 98, 108, 110–113, 225, 228, 230](#)

deleting components [224](#)

dependencies [46](#)

designing [17](#)

details [100, 169](#)

developing [18](#)

distributing [123](#)

editing lead manager [171](#)

editing owner [170](#)

Editions [18](#)

installing packages [64](#)

installing using the API [106](#)

managed [18](#)

packageable components [21, 25, 34](#)

post install script [108, 110, 228, 230](#)

protected components [45](#)

protecting intellectual property [49](#)

status [18](#)

terminology [18](#)

test failures, resolving [107](#)

understanding [18](#)

uninstall script [111–112](#)

uninstalling using the API [106](#)

unmanaged [18](#)

uploading [21, 98](#)

user support [205](#)

versions [171](#)

Packaging

lifecycle [2](#)

push upgrades, scheduling [231](#)

- packaging a Connected App [72](#)
- partner account [137](#)
- Partner API Key [151](#)
- Partner logo [126](#)
- Partner Order [154–155](#), [158](#)
- Partner WSDL [56](#)
- Patch versions
 - creating [217](#)
 - creating patches [218](#)
 - uploading [217](#)
 - uploading patches [218](#)
- patches [138](#)
- Permission sets [47](#)
- Permissions needed [164](#)
- popularity [139](#)
- Product Catalogs [147](#), [151–152](#)
- Professional edition
 - access control [60](#)
 - accessing REST API [62](#)
 - limits [60](#)
 - packages [58–59](#), [61–62](#)
 - using Apex [60](#)
- Profile settings [47](#)
- Profiles [149](#)
- provider profile
 - and partner accounts [137](#)
 - create or edit [125](#)
- proxy signup [202](#)
- publish
 - create a test drive [132](#)
 - making your listing public [124](#)
- Publishing console [125](#)
- Push upgrades
 - job details [232](#)
 - organization details [233](#)
 - scheduling [231](#)

R

- relevance [139](#)
- resources [118](#)
- REST API
 - accessing in group edition [62](#)
 - accessing professional edition [62](#)
- review
 - edit [138](#)
 - write [140](#)

S

- Sales [175](#)

- sandbox [138](#)
- search optimization [138](#)
- security review
 - requirements [127](#)
 - submit apps and trial templates [127](#)
- Security review
 - extension package [118](#)
 - mobile app [118](#)
 - questionnaire [117](#)
- Service Order Credentials [147](#)
- services
 - publish [124](#)
 - search optimization [138](#)
 - sell [124](#)
- setting up [208](#)
- Signup Request
 - create [199](#)
- Signup Requests
 - home page [199](#)
 - viewing details [200](#)
- SignupRequest object [193](#)
- start URL in Connected App [78](#)
- Subscriber support [206–207](#)
- Support for end users [205](#)
- Supporting multiple editions [63](#)

T

- Terminology
 - License Management App [165](#)
- test drive
 - create or edit [132](#)
 - vs free trial [140](#)
- Testing [3](#), [93](#)
- trial template
 - description [119](#), [140](#)
 - submit for security approval [127](#)
- Trialforce
 - best practices [192](#)
 - create a trial organization [192](#), [201](#)
 - customizing HTML form [190](#)
 - installing another app [204](#)
 - link template to HTML form [189](#)
 - modifying a trial [191](#)
 - providing a free trial on AppExchange [187](#)
 - provisioning trial organizations [190](#)
 - signup [192](#), [201](#)
- Troubleshooting [177](#)
- Tutorials [4](#)

U

- uninstalling a Connected App [82](#)
- Uninstalling packages [105](#)
- Unmanaged packages
 - component behavior [36](#)
 - developing [20](#)
 - packageable components [25](#), [34](#)
 - protected components [45](#)
- update [203](#)
- upgrade [140](#)
- Upgrading packages [215](#)

- Uploading beta packages [96](#)
- Uploading packages [21](#), [98](#)
- Uploading patches [217](#)
- Usage Metrics [208–209](#)
- User support [205](#)
- using an extension package [62](#)
- using dynamic Apex [62–63](#)

W

- whitelisting IP ranges in Connected App [72](#)
- WSDLs
 - downloading [57](#)