Distributing Enterprise Apps for iOS Devices

Distributing Enterprise Apps for iOS Devices

Introduction

This document describes how to purchase apps from the App Store in bulk, and how to distribute enterprise apps that you develop for in-house use.

In addition to the methods described here, some Mobile Device Management (MDM) servers let you instruct a device to install an enterprise app or App Store app. You an also use MDM to remove these “managed apps” and mark them so that they aren’t backed up by iTunes or iCloud. For information see the documentation that came with your MDM server.

Distributing Enterprise Apps for iOS Devices ► Volume Purchase Program for business

About the Volume Purchase Program for business

The App Store features thousands of great commercial apps that users can purchase, download, and install from the App Store. With the App Store Volume Purchase Program, your business can purchase iOS apps from the App Store in volume for distribution to employees. All paid apps on the App Store are eligible for purchase under the program, at list price and in any quantity.

The Volume Purchase Program also lets you purchase custom B2B apps developed for you by third-party developers and business partners.

Distributing Enterprise Apps for iOS Devices ► Volume Purchase Program for business

Enrolling in the Volume Purchase Program

To purchase apps in volume for your business, you need to enroll and create a volume purchasing account with Apple. You need to provide information about your business, such as a D&B D-U-N-S number and contact information. You also need to create an Apple ID that’s used only for volume purchasing.

For more information about enrollment and the countries or regions where the Volume Purchase Program is available, go to: www.apple.com/business/vpp

---

Distributing Enterprise Apps for iOS Devices ► Volume Purchase Program for business

Purchasing apps in volume

You use the Volume Purchase Program website to purchase apps for your business. The URL is: vpp.itunes.apple.com/store/us

Use the Apple ID associated with your Volume Purchase Program account to log in to the website. Search for the apps you want to purchase, then indicate the number of copies you’re purchasing. You pay with a corporate credit card. There’s no limit to the number of copies of an app that you can purchase. For each copy you purchase, you receive a unique redemption code that lets your users download and install the app without

purchasing it.

You can purchase only paid apps in volume. Users can download free apps individually from the App Store.

After you make a purchase, you're notified by email when your redemption codes are ready. An XLS spreadsheet containing the redemption codes will be available in the account section of the Volume Purchase Program website. The website lists each purchase by order number, app name, total cost, and number of licences. Download the associated spreadsheet to view the redemption codes for each app, in the quantity purchased. For example, if you purchase seven copies of the Pages app, you receive seven redemption codes for Pages.

The spreadsheet also contains a redemption URL for each redemption code. These URLs let users download and install the apps onto their devices without entering the redemption code.

Distributing redemption codes

You can distribute the redemption URLs by email or SMS, or post them on a website that you make accessible to the appropriate groups and users. You may want to create a website that offers a catalog of the apps you purchased and that issues redemption codes to authorized users. Many third-party Mobile Device Management (MDM) solutions also provide a way to centrally manage and distribute codes.

Users install the apps you purchase for them by going to the redemption URL on their iOS device. This takes them directly to the App Store with the redemption code already entered, so all they have to do is authenticate with their Apple ID. It's the same process as with any other app from the App Store, but because you've provided the prepaid redemption code, users aren't charged for the purchase.

Each redemption code can be used only once. Each time a redemption code is used, an updated version of the purchase spreadsheet becomes available for you at the Volume Purchase Program website. Download the spreadsheet to see how many codes have been used, and to view the remaining redemption codes.

Once a user installs the app, it's backed up and updated just like any other App Store app.

Purchasing custom B2B apps

Custom apps that a vendor creates or customizes for your business (B2B) can also be purchased via the Volume Purchase Program.

Vendors registered in the iOS Developer Program can submit apps for B2B distribution using iTunes Connect, the same process used to submit other apps to the App Store. The developer sets the price per copy and adds your Volume Purchase Program Apple ID to their authorized B2B purchasers list. Only authorized purchasers are able to see or purchase the app.

B2B apps aren't secured by Apple—the security of the data in an app is the responsibility of the developer. Apple recommends using iOS best practices for in-app authentication and encryption.

B2B apps are reviewed by Apple to ensure they meet the App Store guidelines. Work with your vendor to determine the best way to allow Apple reviewers to log in and review your app. You may want to provide a generic account or sanitized test data for this purpose.

After Apple reviews the app, you use the Volume Purchase Program website to purchase copies and get redemption URLs, as described in Purchasing apps in volume. B2B apps aren't listed on the App Store—they must be purchased using the Volume Purchase Program website and installed using redemption URLs.

About in-house apps

If you develop your own iOS apps for use by your company, the iOS Developer Enterprise Program lets you deploy the in-house apps. The process for deploying an in-house app is:

- Register for the iOS Developer Enterprise Program.
- Prepare your app for distribution.
- Create an enterprise distribution provisioning profile that authorizes devices to use apps you've signed.
- Build the app with the provisioning profile.
- Deploy the app to your users.

Registering for app development

To develop and deploy custom apps for iOS, first register for the iOS Developer Enterprise Program at: developer.apple.com/programs/ios/enterprise

Once you register, you can request a developer certificate and developer provisioning profile. You use these during development to build and test your app. The development provisioning profile allows apps signed with your developer certificate to run on registered devices. You create the developer provisioning profile at the iOS Provisioning Portal. The ad-hoc profile expires after 3 months and specifies which devices (by device ID) can run development builds of your app. You distribute your developer signed build, and the development provisioning profile, to your app team and testers.

Preparing apps for distribution

After you finish development and testing and are ready to deploy your app, you sign your app using your distribution certificate and package it with a provisioning profile. The designated Team Agent or the Admin for your program membership creates the certificate and profile at the iOS Provisioning Portal at: http://developer.apple.com/iphone

Generating the distribution certificate involves using the Certificate Assistant (which is part of the Keychain Access application on your Mac OS X development system) to generate a Certificate Signing Request (CSR). You upload the CSR to the iOS Provisioning Portal and receive a distribution certificate in response. When you install this certificate in Keychain, you can set Xcode to use it to sign your app.

The enterprise distribution provisioning profile allows your app to be installed on an unlimited number of iOS devices. You can create an enterprise distribution provisioning profile for a specific app, or for multiple apps.

Once you have both the enterprise distribution certificate and provisioning profile installed on your Mac, you use Xcode to sign and build a release/production version of your app. Your enterprise distribution certificate is valid for three years, after which you'll have to sign and build your app again using a renewed certificate. The provisioning profile for the app is good for one year, so you'll want to release new provisioning profiles annually. See Providing updated apps.

It's very important that you limit access to your distribution certificate and its private key. Use Keychain Access on OS X to export and back up these items in p12 format. If the private key is lost, it cannot be recovered or redownloaded. In addition to keeping the certificate and private key safe, you should restrict access to personnel who are responsible for final acceptance of the app. Signing an app with the distribution certificate gives your company's seal of approval for the app's content, function, and adherence to the Enterprise Developer Agreement licensing terms.

About deploying apps

There are four ways to deploy an app:

- Distribute the app for your users to install using iTunes.
- Have an IT administrator install the app on devices using iPhone Configuration Utility or Apple Configurator.
- Post the app on a secure web server; users access and perform the installation wirelessly.
- Use your MDM server to instruct managed devices to install an in-house or App Store app, if your MDM server supports it.

Installing apps using iTunes

If your users use iTunes to install apps on their devices, securely distribute the app to the users and have them follow these steps.

To install apps on a user's device:

1. In iTunes, choose File > Add to Library, and then select the file (.app, .ipa, or .mobileprovision). The user can also drag the file to the iTunes application icon.
2. Connect a device to the computer, and then select it in the Devices list in iTunes.
3. Click the Apps tab, and then select the app in the list.
4. Click Apply.

If your users' computers are managed, instead of asking them to add the files to iTunes, you can deploy the files to their computers and ask them to sync their device. iTunes automatically installs the files found in iTune's Mobile Applications and Provisioning Profiles folders.

Installing apps using iPhone Configuration Utility or Apple Configurator

You can use iPhone Configuration Utility (Windows) or Apple Configurator (OS X) to install apps and profiles on iOS devices.

iPhone Configuration Utility, available from support.apple.com/kb/DL1466, can be used by an IT administrator to install in-house apps directly onto iOS devices over USB.

Apple Configurator, available from the Mac App Store, can be used by IT administrators to install in-house apps —or apps from the App Store using Volume Purchase Program redemption codes—over USB. Apple Configurator keeps track of Volume Purchase Program licenses you've used, too.

Installing apps wirelessly

iOS supports over-the-air installation of enterprise apps, letting you distribute in-house software to your users without using iTunes.

Requirements

- A secure web server that authenticated users can access
- An iOS app in .ipa format, built for release/production with an enterprise provisioning profile
- An XML manifest file, described later in this document
- A network configuration that allows the devices to access an iTunes server at Apple

Installing the app is simple. Users download the manifest file from your website to their iOS device. The manifest file instructs the device to download and install the apps referenced in the manifest file.

You can distribute the URL for downloading the manifest file by SMS or email, or by embedding it in another enterprise app you create.

It's up to you to design and host the website used to distribute apps. Make sure that users are authenticated, perhaps using basic auth or directory-based authentication, and that the website is accessible via your intranet or the Internet. You can place the app and manifest file in a hidden directory, or in any other location that's readable using HTTP or HTTPS.

Preparing an enterprise app for wireless distribution

To prepare your enterprise app for wireless distribution, you build an archived version (an .ipa file), and a manifest file that enables wireless distribution and installation of the app.

You use Xcode to create an app archive. Sign the app using your distribution certificate and include your enterprise deployment provisioning profile in the archive. For information about the manifest file, see below. For more information about building and archiving apps, visit the iOS Dev Center or refer to the *Xcode User Guide*, available from the Help menu in Xcode.

About the wireless manifest file

The manifest file is an XML plist. It's used by an iOS device to find, download, and install apps from your web server. The manifest file is created by Xcode, using information you provide when you share an archived app for enterprise distribution. See the previous section, Preparing apps for distribution.

The following fields are required:

Item	Description
URL	The fully qualified HTTP or HTTPS URL of the app (.ipa) file.
display-image	A 57 x 57-pixel PNG image that's displayed during download and installation. Specify the image's fully qualified URL.

full-size-image	A 512 x 512-pixel PNG image that represents the app in iTunes.
bundle-identifier	Your app's bundle identifier, exactly as specified in your Xcode project.
bundle-version	Your app's bundle version, as specified in your Xcode project.
title	The name of the app, which is displayed during download and installation.

For Newsstand apps only, the following fields are required:

Item	Description
newsstand-image	A full-sized PNG image for display on the Newsstand shelf.
UINewsstandBindingEdge UINewsstandBindingType	These keys must match those in your Newsstand app's info.plist.
UINewsstandApp	Indicates that the app is a Newsstand app.

Optional keys you can use are described in the sample manifest file. For example, you can use the MD5 keys if your app file is large and you want to ensure download integrity beyond the error checking normally done for TCP communications.

You can install more than one app with a single manifest file by specifying additional members of the `items` array.

A sample manifest file is included at the end of this document.

Constructing your website

Upload these items to an area of your website that your authenticated users can access:

- The app (.ipa) file
- The manifest (.plist) file

Your website design can be as simple as a single page that links to the manifest file. When a user taps a web link, the manifest file is downloaded, which triggers the downloading and installation of the apps it describes.

Here's a sample link:

```
<a href="itms-services://?action=download-manifest&url=http://example.com/
manifest.plist">Install App</a>
```

Don't add a web link to the archived app (.ipa). The .ipa is downloaded by the device when the manifest file is loaded. Although the protocol portion of the URL is itms-services, the iTunes Store isn't involved in this process.

Setting server MIME types

You may need to configure your web server so the manifest file and app file are transmitted correctly.

For OS X Server, add the following MIME types to the Web service's MIME Types settings:

```
application/octet-stream ipa
text/xml plist
```

For IIS, use IIS Manager to add the MIME type in the Properties page of the server:

```
.ipa application/octet-stream
.plist text/xml
```

Distributing Enterprise Apps for iOS Devices ► In-house apps ► Deploying apps

Troubleshooting wireless app distribution

If wireless app distribution fails with an “unable to download” message, check the following:

- Make sure the app is signed correctly. Test it by installing it on a device using iPhone Configuration Utility or Apple Configurator, and see if any errors occur.
- Make sure the link to the manifest file is correct and the manifest file is accessible to web users.
- Make sure the URL to the .ipa file (in the manifest file) is correct and the .ipa file is accessible to web users.

Distributing Enterprise Apps for iOS Devices ► In-house apps ► Deploying apps

Network configuration requirements

If the devices are connected to a closed internal network, you should let iOS devices access the following:

URL	Reason
ax.init.itunes.apple.com	The device obtains the current file-size limit for downloading apps over the cellular network. If this site isn’t reachable, installation may fail.
ocsp.apple.com	The device contacts this site to check the status of the distribution certificate used to sign the provisioning profile. See Certificate validation.

Distributing Enterprise Apps for iOS Devices ► In-house apps

Providing updated apps

Apps you distribute yourself aren’t automatically updated. When you have a new version for users to install, notify them of the update and instruct them to install the app. Consider having the app check for updates and notify the user when it opens. If you’re using wireless app distribution, the notification can provide a link to the manifest file of the updated app.

If you want users to keep the app’s data stored on their device, make sure the new version uses the same bundle-identifier as the one it’s replacing, and tell users not to delete their old version before installing the new one. The new version will replace the old one and keep data stored on the device, if the bundle-identifiers match.

Distribution provisioning profiles expire 12 months after they’re issued. Two months before expiration, the iOS device begins displaying notifications about the impending expiration. After the expiration date, the app won’t

launch.

Before to a provisioning profile expires, use the iOS Development Portal to create a new profile for the app. Create a new app archive (.ipa) with the new provisioning profile, for users who are installing the app for the first time.

For users who already have the app, you may want to time your next released version so that it includes the new provisioning profile. If not, you can distribute just the new .mobileprovision file so users won't have to install the app again. The new provisioning profile will override the one that's already in the app archive.

Provisioning profiles can be installed and managed using MDM, downloaded and installed by users from a secure website that you provide, or distributed to users as an email attachment to open and install.

When your distribution certificate expires, the app won't launch. Your distribution certificate is valid for three years from when it was issued, or until your Enterprise Developer Program membership expires, whichever comes first. To prevent the premature expiration of your certificate, be sure to renew your membership before it expires. For information about how the distribution certificate is checked, see Certificate validation.

You can have two distribution certificates active at the same time; each is independent from the other. The second certificate is intended to provide an overlapping period during which you can update your apps before the first certificate expires. When requesting your second distribution certificate from the iOS Dev Center, be sure you don't revoke your first certificate.

Distributing Enterprise Apps for iOS Devices ► In-house apps

Certificate validation

The first time a user opens an app, the distribution certificate is validated by contacting Apple's OCSP server. Unless the certificate has been revoked, the app is allowed to run. Inability to contact or get a response from the OCSP server isn't interpreted as a revocation. To verify the status, the device must be able to reach ocsp.apple.com. See Network configuration requirements.

The OCSP response is cached on the device for the period of time specified by the OCSP server—currently, between three and seven days. The validity of the certificate isn't checked again until the device has restarted and the cached response has expired. If a revocation is received at that time, the app is prevented from running.

Revoking a distribution certificate invalidates all of the apps you've signed with it. You should revoke a certificate only as a last resort, if you're sure the private key is lost or the certificate is believed to be compromised.

Distributing Enterprise Apps for iOS Devices

Sample app manifest file

```
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
   <!-- array of downloads. -->
   <key>items</key>
   <array>
       <dict>
           <!-- an array of assets to download -->
           <key>assets</key>
```

```
            <array>
                <!-- software-package: the ipa to install. -->
                <dict>
                    <!-- required. the asset kind. -->
                    <key>kind</key>
                    <string>software-package</string>
                    <!-- optional. md5 every n bytes. will restart a chunk if md5 fails.
    -->
                    <key>md5-size</key>
                    <integer>10485760</integer>
                    <!-- optional. array of md5 hashes for each "md5-size" sized chunk.
    -->
                    <key>md5s</key>
                    <array>
                        <string>41fa64bb7a7cae5a46bfb45821ac8bba</string>
                        <string>51fa64bb7a7cae5a46bfb45821ac8bba</string>
                    </array>
                    <!-- required. the URL of the file to download. -->
                    <key>url</key>
                    <string>http://www.example.com/apps/foo.ipa</string>
                </dict>
                <!-- display-image: the icon to display during download .-->
                <dict>
                    <key>kind</key>
                    <string>display-image</string>
                    <!-- optional. indicates if icon needs shine effect applied. -->
                    <key>needs-shine</key>
                    <true/>
                    <key>url</key>
                    <string>http://www.example.com/image.57x57.png</string>
                </dict>
                <!-- full-size-image: the large 512x512 icon used by iTunes. -->
                <dict>
                    <key>kind</key>
                    <string>full-size-image</string>
                    <!-- optional. one md5 hash for the entire file. -->
                    <key>md5</key>
                    <string>61fa64bb7a7cae5a46bfb45821ac8bba</string>
                    <key>needs-shine</key>
                    <true/>
                    <key>url</key>
    <string>http://www.example.com/image.512x512.jpg</string>
                </dict>
            </array><key>metadata</key>
            <dict>
                <!-- required -->
                <key>bundle-identifier</key>
```

```
                <string>com.example.fooapp</string>
                <!-- optional (software only) -->
                <key>bundle-version</key>
                <string>1.0</string>
                <!-- required. the download kind. -->
                <key>kind</key>
                <string>software</string>
                <!-- optional. displayed during download; typically company name -->
                <key>subtitle</key>
                <string>Apple</string>
                <!-- required. the title to display during the download. -->
                <key>title</key>
                <string>Example Corporate App</string>
            </dict>
        </dict>
    </array>
</dict>
</plist>
```
