Dictionary
App Builder

Installing and Building Apps on a Mac

SIL

Dictionary App Builder:
Installing and Building Apps on a Mac

Last updated: 25 March 2025

The latest version is available at
http://software.sil.org/dictionaryappbuilder/resources/

and on the Help menu of Dictionary App Builder.

Contents

1. Introduction

This document provides information on how to install Dictionary App Builder and build apps on an Apple macOS system. It is possible to build an Android app using DAB on Windows, Linux or Mac, but if you want to build an iOS app for the iPhone or iPad, you will need to build it using a Mac computer.

App Builder Platform	**Build Android Apps**	**Build iOS Apps**
Windows	✓	
Linux	✓	
macOS	✓	✓

Creating an Android app on a Mac is essentially the same process as it is for Windows or Linux. To create a corresponding iOS app, you will need to enter a few more configuration items.

The apps generated by DAB for iOS will run on iPhones and iPads with iOS 12.2 or higher.

2. Installing Dictionary App Builder

To install the Dictionary App Builder program files:

1. Download the current Mac installer file (dmg) from the DAB website:

   http://software.sil.org/dictionaryappbuilder/download/

2. Double click on the file within **Finder** to open the disk image that contains the Dictionary App Builder application.

3. Copy the **Dictionary App Builder** application to your **Application** folder. This can be done by dragging the Dictionary App Builder icon from the disk image window to the shortcut of the Applications folder in the same window.

3. Installing Prerequisites for Android

If you want to build Android apps, you need to install the following components on your computer:

1. Java Development Kit (JDK)
2. Android Software Development Kit (SDK)

3.1. Java Development Kit (JDK)

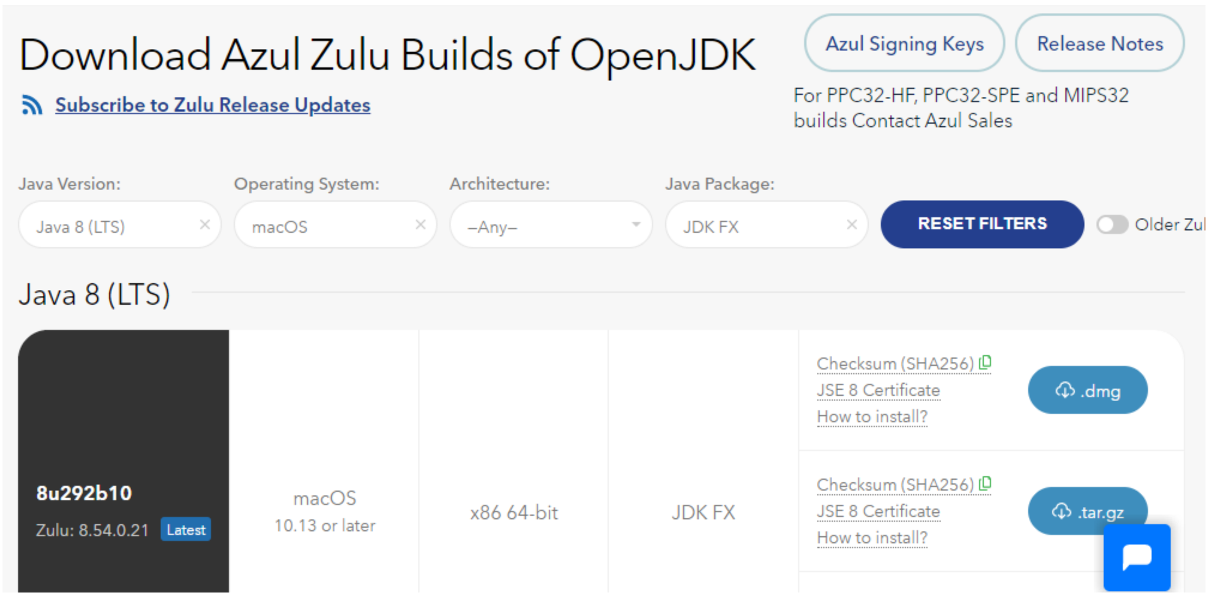
You will need version 8 of the Java Development Kit (JDK) to build apps. We recommend you use Zulu, which is a free distribution of OpenJDK from Azul.

1. Go to the **Download Zulu Builds of OpenJDK** website:

   https://www.azul.com/downloads/?version=java-8-lts&os=macos&package=jdk-fx

   There are many downloads on this page, but the above link will filter the ones you see (Java Version: Java 8 LTS; Operating System: MacOS; Java Package: Java FX).

2. Scroll down the page until you see the downloads under the heading **Download Azul Zulu Builds of OpenJDK**:



Download Azul Zulu Builds of OpenJDK

Subscribe to Zulu Release Updates

For PPC32-HF, PPC32-SPE and MIPS32 builds Contact Azul Sales

Java Version: Java 8 (LTS)
Operating System: macOS
Architecture: –Any–
Java Package: JDK FX

Java 8 (LTS)

8u292b10 Zulu: 8.54.0.21 Latest	macOS 10.13 or later	x86 64-bit	JDK FX	Checksum (SHA256) / JSE 8 Certificate / How to install? — .dmg
				Checksum (SHA256) / JSE 8 Certificate / How to install? — .tar.gz

3. You have a choice between two different architectures: **x86 64-bit** and **ARM 64-bit**. You can find the processor type of your machine by clicking on the Apple symbol at the top left of your screen and selecting **About This Mac**. If you have one of the newer Macs with an M1 chip, choose ARM, otherwise you will need x86 (Intel).

   Once you have identified your device architecture, you have a choice between a **dmg** file, a **tar.gz** file and a **zip** file. **Download the .dmg file** since it comes with its own installer program.

   The file you download will have a filename something like this:

   zulu8.54.0.21-ca-fx-jdk8.0.292-macosx_x64.msi

4. **Double-click the file in Finder** and follow the instructions in the installation wizard to install it. By default, the installer will install the JDK to the following folder:

   /Library/Java/JavaVirtualMachines/zulu-8.jdk/Contents/Home

> **Important**: If you change the JDK install folder to something other than the default folder, you will need to remember the location of the folder so you can tell Dictionary App Builder where to find the JDK.

3.2. Installing Android Software Development Kit (SDK)

The Android Software Development Kit (SDK) is needed for building Android apps. There are two ways of installing the Android SDK:

1. **Online: Download the Android SDK packages from the internet:**
   Use the Android SDK Installation wizard to download and install the command line tools and three additional packages. This method will require an internet connection.

   See 3.2.1 for more details.

2. **Offline: Copy the Android SDK files from someone else:**
   If you know someone who has already downloaded and installed the Android SDK, you can copy all the files from them.

   This method is especially useful in a training workshop where several people need to install the SDK but have limited internet bandwidth.

   See 3.2.2 for more details.

3.2.1. Downloading the Android SDK packages from the internet

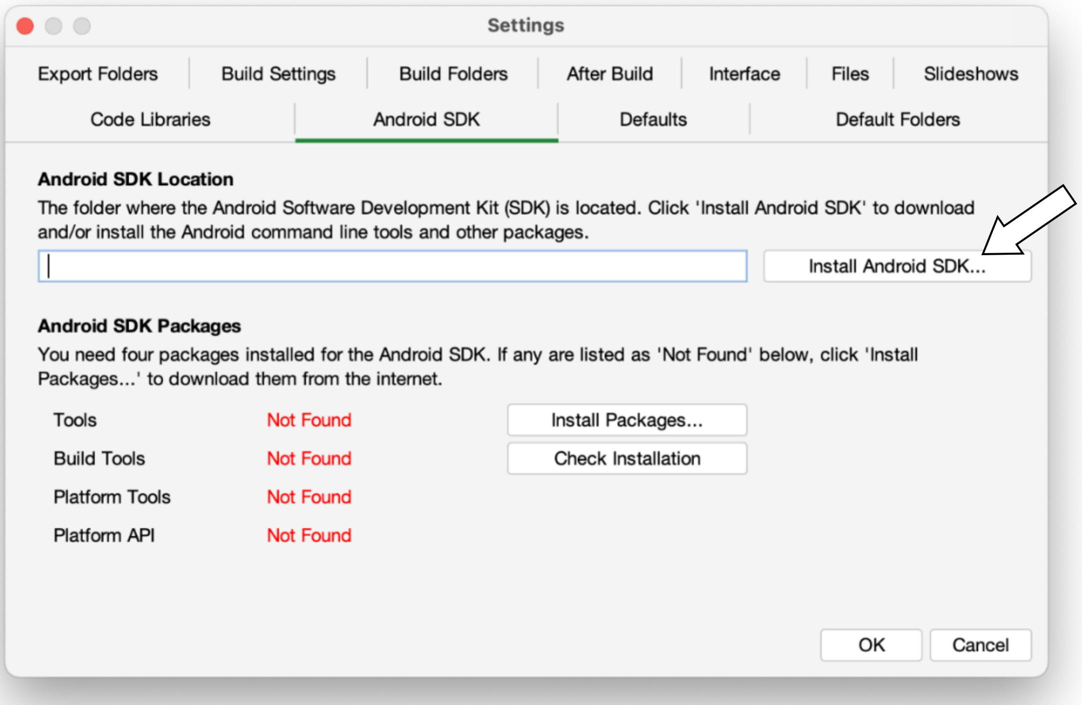
To install the Android SDK from the internet:

1. Launch **Dictionary App Builder**.

2. Select **Dictionary App Builder** ➢ **Preferences** from the main menu.

3. Go to the **Android SDK** tab, which is the second tab.

4. Click the **Install Android SDK** button.



Android SDK Location

The folder where the Android Software Development Kit (SDK) is located. Click 'Install Android SDK' to download and/or install the Android command line tools and other packages.

Android SDK Packages

You need four packages installed for the Android SDK. If any are listed as 'Not Found' below, click 'Install Packages...' to download them from the internet.

Tools	Not Found
Build Tools	Not Found
Platform Tools	Not Found
Platform API	Not Found

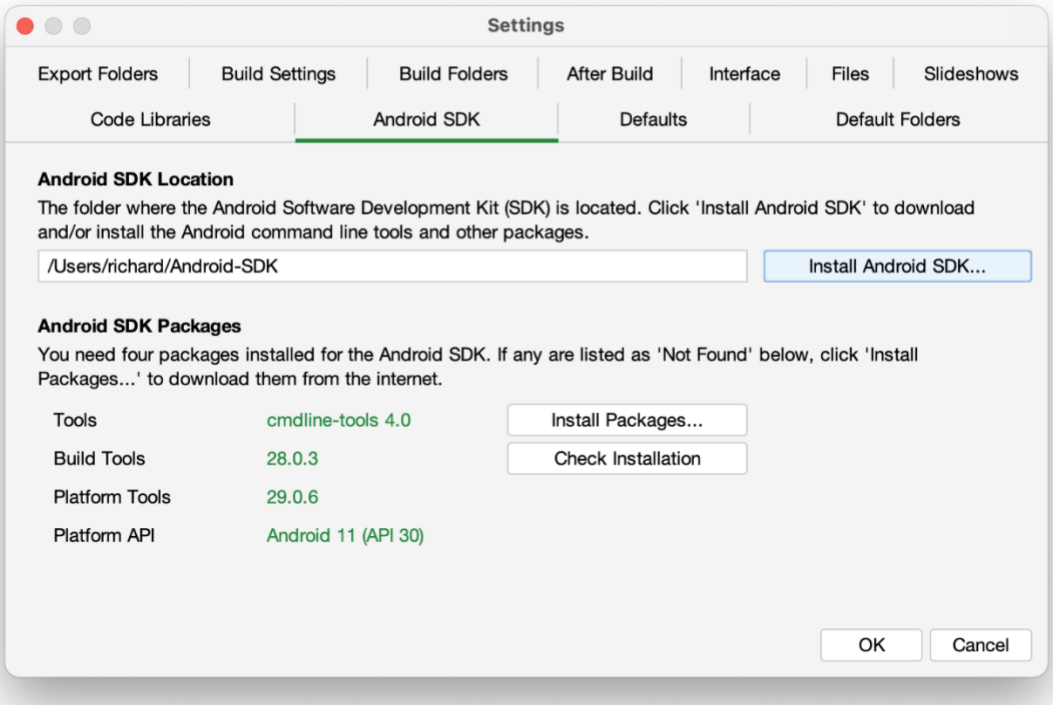
5. Follow the instructions on each page of the **Install Android SDK** wizard to download each of the Android SDK packages and install them.

   When you are asked to specify a target folder, a good place is:
   /Users/*your-name*/Android-SDK.

   Four packages will be downloaded and installed:

   - Command line tools,
   - Build Tools,
   - Platform Tools, and
   - Platform API.

   If the installation was successful, you will see the version numbers displayed in green.



Android SDK Location

The folder where the Android Software Development Kit (SDK) is located. Click 'Install Android SDK' to download and/or install the Android command line tools and other packages.

/Users/richard/Android-SDK

Install Android SDK...

Android SDK Packages

You need four packages installed for the Android SDK. If any are listed as 'Not Found' below, click 'Install Packages...' to download them from the internet.

Tools	cmdline-tools 4.0
Build Tools	28.0.3
Platform Tools	29.0.6
Platform API	Android 11 (API 30)

If any of the Build Tools, Platform Tools or Platform API is listed as “Not Found” (displayed in red), click the **Install Packages** button to install them.

Click the **Check Installation** button to confirm that all the packages have been installed correctly.

You can skip section 3.2.2 and go straight to section 4.

3.2.2. Copying the Android SDK files from someone else

If you know someone who has already downloaded and installed the Android SDK and is successfully building apps with it, you can copy all of their Android SDK files to a folder on your computer.

You need to look for the top-level Android SDK folder, such as **/Users/*user-name*/Android-SDK**, and copy the whole folder and its contents to your computer. A location such as **/Users/*your-name*/Android-SDK** is good. If it makes it easier, you can zip the folders and then unzip them onto your computer.

Note that there is <u>no setup program to run</u>. Copying the files from one computer to another is sufficient.

Tip: A typical Android SDK folder can be quite large (over 1 GB, depending on which additional packages have been installed). To build an app, you do not actually need all of the Android SDK files. If you want to cut down the number of files, here is a list of the essential and optional folders:

Android SDK Folder	**Required for building apps?**
cmdline-tools (or tools)	Yes
build-tools	Yes (you only need the sub-folder for the latest version)
platforms	Yes (you only need android-30 for now)
platform-tools	Yes
add-ons	No
docs	No
emulator	No, unless you want to use an emulator
extras	No
licenses	Yes
sources	No
system-images	No, unless you want to use an emulator
temp	No

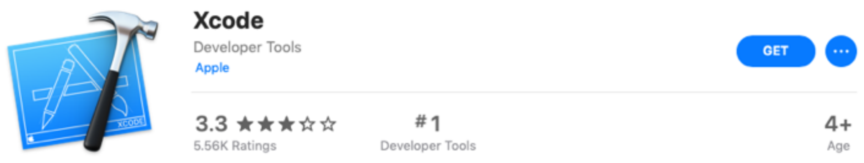
4. Installing Prerequisites for iOS

If you want to build iOS apps and upload them to the Apple App Store, you need to install the following components:

1. **Xcode**
2. **Transporter**

4.1. Install Xcode

Xcode is required to build iOS Apps. To install Xcode, simply search for Xcode in the Mac App Store and install it. Open Xcode at least once to agree to the licensing restrictions and install components.



4.2. Verify Xcode Installation

To verify that you have successfully installed Xcode and that it is will be correctly used by Dictionary App Builder, please open Terminal and run the following command to print the path to the active developer directory:

```
xcode-select -p
```

```
$ xcode-select -p
/Applications/Xcode.app/Contents/Developer
```

If you have had Xcode Command-line Tools installed previously, it might still be pointing to that installation directory and Dictionary App Builder will not work correctly.

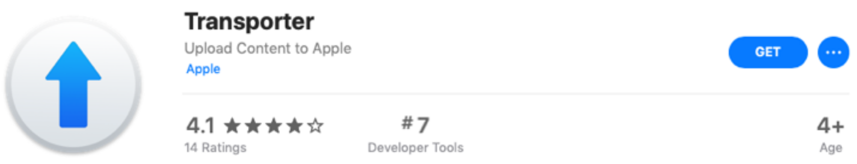
```
$ xcode-select -p
/Library/Developer/CommandLineTools
```

To correct this situation, run the following command to set the active developer directory.

```
sudo xcode-select -s /Applications/Xcode.app/Contents/Developer
```

4.3. Install Transporter

Transporter is used to upload iOS apps to App Store Connect. To install Transporter, simply search for Transporter in the Mac App Store and install it.



5. Testing App in iOS Simulator

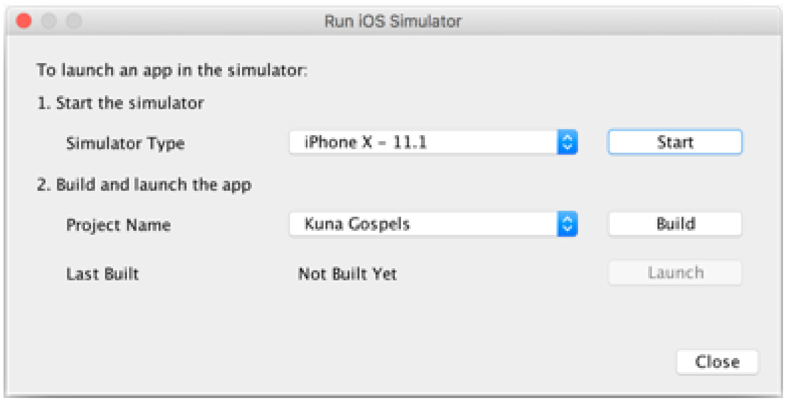
When you want to test your app, you can either use a device or a simulator. To test with a device, you will need a signing certificate and provisioning profile (see next section). To test with a simulator, you will need to download Xcode, the integrated development environment used to build and test iOS and Mac apps. Xcode is available for free in the Mac App Store and is quite large (5.46GB). DAB requires Xcode 9 or greater (which requires macOS Sierra). To install:

1. Go to https://itunes.apple.com/us/app/xcode/id497799835?mt=12 or search for “Xcode” in the App Store app on macOS.
2. Install from the App Store

3. Start Xcode at least once to complete the installation

5.1. Run the iOS Simulator

Once you have a project configured and ready to test, click on the **Run iOS App in Simulator** on the toolbar.



To launch an app in the simulator:

1. Start the simulator

Simulator Type: iPhone X – 11.1

2. Build and launch the app

Project Name: Kuna Gospels

Last Built: Not Built Yet

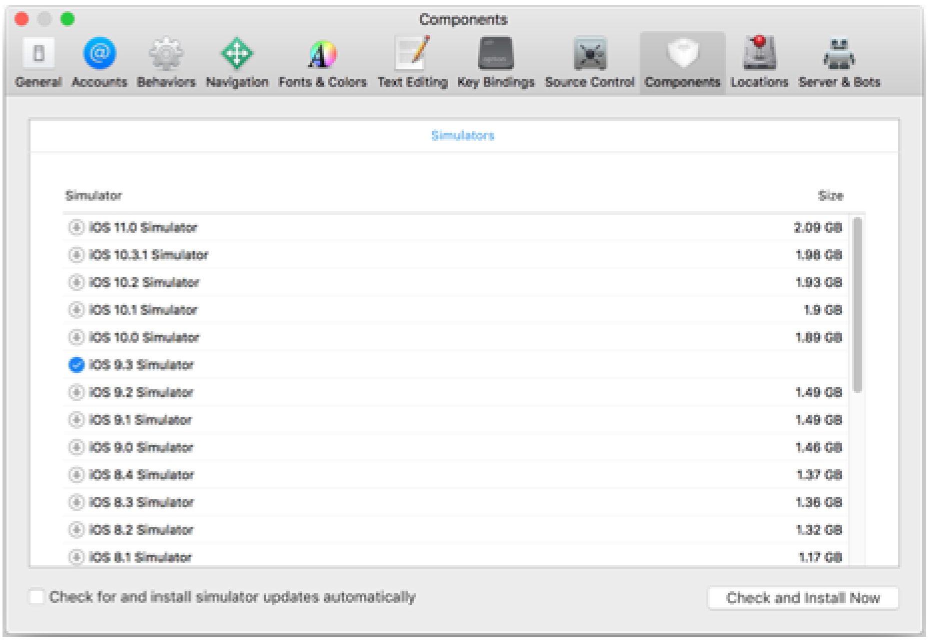
Select the simulator you would like to run on and click **Start**. It takes a little bit of time for the simulator to start. If you want to switch simulators, select a different Simulator from the **Simulator Type** combo box and click **Start** again. It will close the previous Simulator and start the new own.

Select the project you want to test (it defaults to the selected project in the Apps list) and click on **Build**. This will build the app for the Simulator in a separate terminal window. When the build is complete, you can click on **Launch** to run the app in the Simulator.

You may close the dialog and make changes to your project. When you restart the Run iOS Simulator dialog again, you will need to Build and Launch again for the changes to be included.

5.2. Installing Additional Simulators

You may install simulators for previous versions of iOS by launching Xcode and viewing the preferences dialog and selecting the Components tab.



Simulator	Size
iOS 11.0 Simulator	2.09 GB
iOS 10.3.1 Simulator	1.98 GB
iOS 10.2 Simulator	1.93 GB
iOS 10.1 Simulator	1.9 GB
iOS 10.0 Simulator	1.89 GB
✓ iOS 9.3 Simulator	
iOS 9.2 Simulator	1.49 GB
iOS 9.1 Simulator	1.49 GB
iOS 9.0 Simulator	1.46 GB
iOS 8.4 Simulator	1.37 GB
iOS 8.3 Simulator	1.36 GB
iOS 8.2 Simulator	1.32 GB
iOS 8.1 Simulator	1.17 GB

☐ Check for and install simulator updates automatically

5.3. Manually Installing Apps Into The Simulator

If there is some problem with launching the simulator from the **Run iOS Simulator** dialog, you can manually install the app by dragging the built app from the Simulator output folder (found in ~/App Builder/Dictionary Apps/Sim Output) onto the Simulator.

To start the Simulator, launch **Xcode** and from the **Xcode** menu select **Open Developer Tool** ➢ **Simulator**. You will still need to rebuild the app from the **Run iOS Simulator** dialog.

6. Creating iOS Certificates and Provisioning Profiles

6.1. Enroll in the Apple Developer Program

To build iOS apps and distribute them through the Apple App Store, you will need to be enrolled in the Apple Developer Program. You can do this as either (i) an individual, or (ii) an organisation. The cost is USD $99 per year.

To enroll:

1. Go to https://developer.apple.com/programs/enroll/
2. Press the **Start Your Enrollment** button to start.

6.2. Create Signing Certificate

When you create an iOS app, it needs to be signed with a certificate.

To create a certificate:

1. Go to the Apple Developer website and log in to your account if you are not already.
2. Select **Certificates, Identifiers & Profiles**.
3. Click the ⊕ button to the right of the **Certificates** header.
4. On the **Create a New Certificates** page, select **Apple Distribution**. Then press the **Continue** button.
5. Upload a Certificate Signing Request. Press the **Learn more** link for instructions on how to use Keychain Access on your Mac computer to complete this task. Then press the **Continue** button.
6. On the **Download Your Certificate** page, press the **Download** button to download the certificate (distribution.cer) to your Mac.
7. Open the Keychain Access application. Choose the **login** item from the Keychains section on the left. Choose the **File** ➢ **Import Items…** menu item and browse to the Downloads folder and select the certificate (distribution.cer).

Now that the certificate is installed in the Keychain, you will be able to access it from within Dictionary App Builder.

Note: To work with certificates, you will need the Apple Worldwide Developer Relations (WWDR) Certification Authority. It should have been installed with Xcode. When viewing your certificate in the Keychain Access application, if there is an error stating the certificate is not trusted, then install the certification authority.

To get the certification authority:

1. Download the Apple WWDR Certification Authority from:
   https://developer.apple.com/certificationauthority/AppleWWDRCA.cer

   For signing certificates created after January 28, 2021:
   https://www.apple.com/certificateauthority/AppleWWDRCAG3.cer
2. Open the Keychain Access application. Choose the **System** item from the Keychains section on the left. Choose the **File** ➢ **Import Items…** menu item and browse to the Downloads folder and select the certification authority (AppleWWDRCA.cer). You will be prompted for a username and password that has admin privileges in order to modify the **System** Keychain.
3. Choose the **File** ➢ **Lock Keychain "System"** menu item.

6.3. Create Provisioning Profile

Distribution provisioning profiles are used to for two primary purposes:

- AdHoc - to install your app on a limited number of registered devices for testing.
- App Store - to submit your app to the App Store.

Creating a provisioning profile

To create a new AdHoc provisioning profile, you will need an App ID and at least one registered device. To create a new App Store provisioning profile, you will need just the App ID.

To create an App ID:

1. Go to the Apple Developer website and log in to your account if you are not already.
2. Select **Certificates, Identifiers & Profiles**.
3. Select **Identifiers** on the left of the page.
4. Click the ⊕ button to the right of the **Identifiers** header.
5. Select **App IDs** and click the **Continue** button.
6. Select **App** and clock the **Continue** button
7. Enter a **Description** of your choice.
   It can be the **App Name** from the **App** ➢ **Package** page of Dictionary App Builder.
8. For Bundle ID, choose **Explicit** and enter your app package name from the **App** ➢ **Package** page of Dictionary App Builder.
9. Leave all the App Services unchecked. None are needed in the app.
10. Click the **Continue** button.

To register a device (i.e. a specific iPhone or iPad for testing):

1. Select **Devices** on the left of the page.
2. Click the ⊕ button to the right of the **Devices** header.
3. In the section **Register a Device**, enter a name of the device (such as “John Smith’s iPhone”) and its UDID (unique device identifier, a sequence of 40 letters and numbers). Press **Continue**.
4. On the next page, check that the device information is displayed correctly and click **Register** to confirm.

   Note that you can register up to 100 devices of each type (e.g. up to 100 iPhones, 100 iPads) per year of your Apple Developer Program membership. You can remove devices that you no longer need at the beginning of the next membership year.

To create a provisioning profile:

1. Select **Profiles** on the left of the page.
2. Click the ⊕ button to the right of the **Profiles** header.
3. On the **Register a New Provisioning Profile** page, under **Distribution**, select **Ad Hoc** (for testing) or **App Store** (for submission to the App Store). Click **Continue** to move to the next page.
4. On the 'Select an App ID' page, select the App ID from the list of App IDs you have already defined. Click the **Continue** button.
5. On the 'Select Certificates' page, select the certificate (Distribution) to include in the profile. Click the **Continue** button.
6. If creating an AdHoc provisioning profile:
   On the 'Select Devices' page, check the device(s) that you want to be able to install and run the app. Click the **Continue** button.
7. On the 'Review, Name, and Generate' page, give the profile a name of your choice. It is helpful to include "App Store" or "AdHoc" in the name. Click the **Continue** button.
8. On the 'Download and Install' page, click **Download** to save your new .mobileprovision file to your computer.

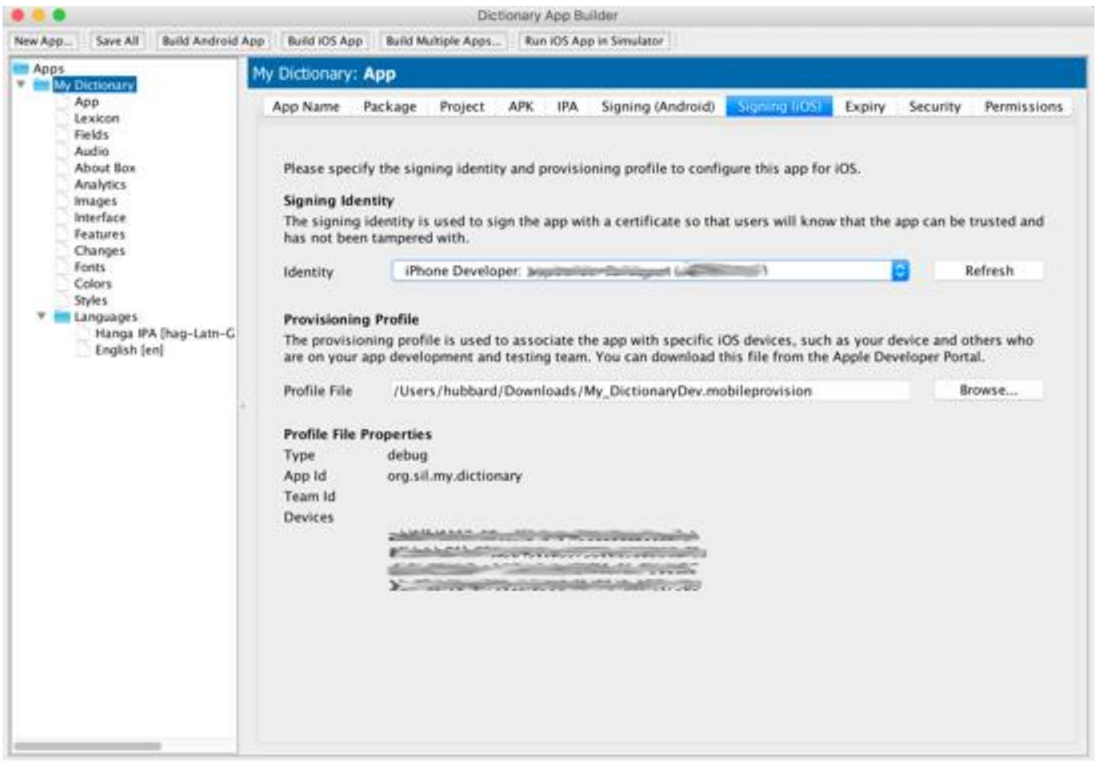
   This is the Provisioning Profile file that you will select in Dictionary App Builder.

Download existing mobile provision files
If other members of your team have already created provisioning profiles, there are several ways to download them. They can be downloaded from the App Developer website by selecting the profile to be downloaded and pressing the **Download** button.

7. Building an iOS App

To build an iOS app:

1. Create a new app project following the instructions in the DAB **Building Apps** document.
2. Select the **Signing (iOS)** tab to open the iOS signing options for the app.


Dictionary App Builder

New App... | Save All | Build Android App | Build iOS App | Build Multiple Apps... | Run iOS App in Simulator

Apps
- My Dictionary
  - App
  - Lexicon
  - Fields
  - Audio
  - About Box
  - Analytics
  - Images
  - Interface
  - Features
  - Changes
  - Fonts
  - Colors
  - Styles
  - Languages
    - Hanga IPA [hag-Latn-G
    - English [en]

My Dictionary: App

App Name | Package | Project | APK | IPA | Signing (Android) | Signing (iOS) | Expiry | Security | Permissions

Please specify the signing identity and provisioning profile to configure this app for iOS.

Signing Identity
The signing identity is used to sign the app with a certificate so that users will know that the app can be trusted and has not been tampered with.

Identity: iPhone Developer: [redacted]   Refresh

Provisioning Profile
The provisioning profile is used to associate the app with specific iOS devices, such as your device and others who are on your app development and testing team. You can download this file from the Apple Developer Portal.

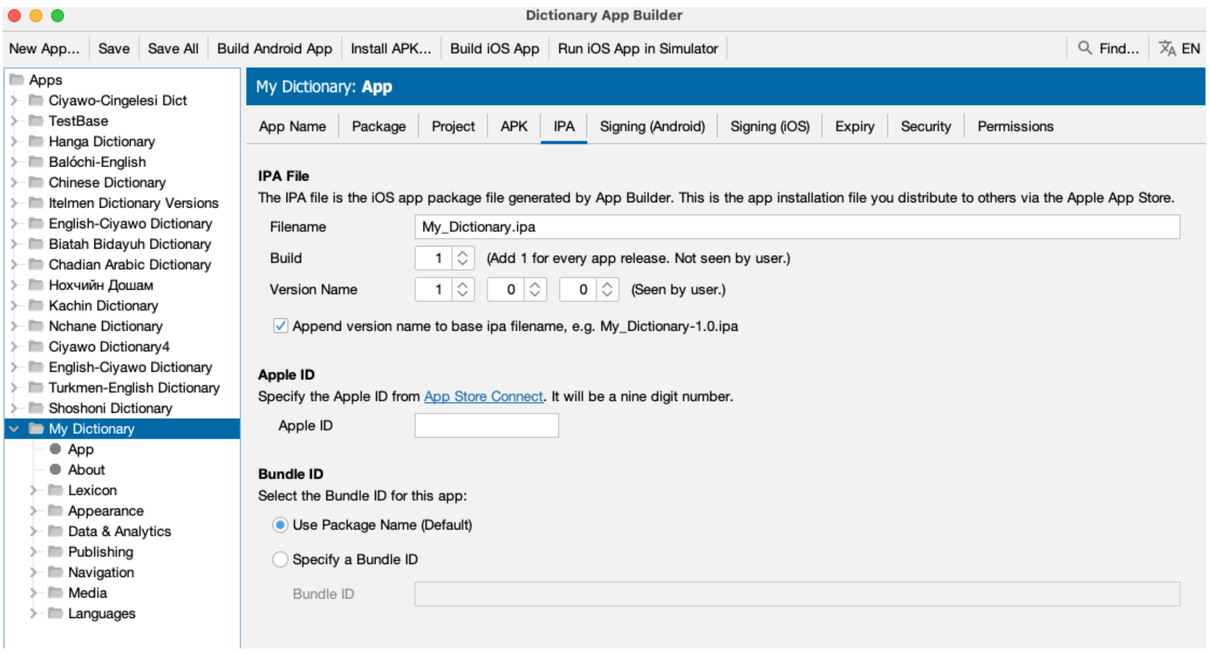
Profile File: /Users/hubbard/Downloads/My_DictionaryDev.mobileprovision   Browse...

Profile File Properties

Type	debug
App Id	org.sil.my.dictionary
Team Id	
Devices	[redacted]

3. Select the **Signing Identity** from the drop-down list of signing certificates which have been downloaded and installed to this system in the earlier steps.

4. For the **Provisioning Profile** entry, enter or browse to the mobile provisioning file associated with the app that was downloaded in the earlier steps.


Dictionary App Builder

New App... | Save | Save All | Build Android App | Install APK... | Build iOS App | Run iOS App in Simulator | Find... | EN

Apps
- Ciyawo-Cingelesi Dict
- TestBase
- Hanga Dictionary
- Balóchi-English
- Chinese Dictionary
- Itelmen Dictionary Versions
- English-Ciyawo Dictionary
- Biatah Bidayuh Dictionary
- Chadian Arabic Dictionary
- Нохчийн Дошам
- Kachin Dictionary
- Nchane Dictionary
- Ciyawo Dictionary4
- English-Ciyawo Dictionary
- Turkmen-English Dictionary
- Shoshoni Dictionary
- My Dictionary
  - App
  - About
  - Lexicon
  - Appearance
  - Data & Analytics
  - Publishing
  - Navigation
  - Media
  - Languages

My Dictionary: App

App Name | Package | Project | APK | IPA | Signing (Android) | Signing (iOS) | Expiry | Security | Permissions

IPA File
The IPA file is the iOS app package file generated by App Builder. This is the app installation file you distribute to others via the Apple App Store.

Filename: My_Dictionary.ipa
Build: 1 (Add 1 for every app release. Not seen by user.)
Version Name: 1 0 0 (Seen by user.)
[✓] Append version name to base ipa filename, e.g. My_Dictionary-1.0.ipa

Apple ID
Specify the Apple ID from App Store Connect. It will be a nine digit number.
Apple ID:

Bundle ID
Select the Bundle ID for this app:
(●) Use Package Name (Default)
( ) Specify a Bundle ID
Bundle ID:

5. Click on the **IPA** tab.

This allows you to set the name of the ipa file to be generated as well as the build and version information.

The ***Filename*** field on the screen specifies the base name of the ipa file to be generated. If the checkbox at the bottom of the screen for *Append version name to ipa filename* is checked, then the version indicated by the *Version Name* fields is added to the base filename.

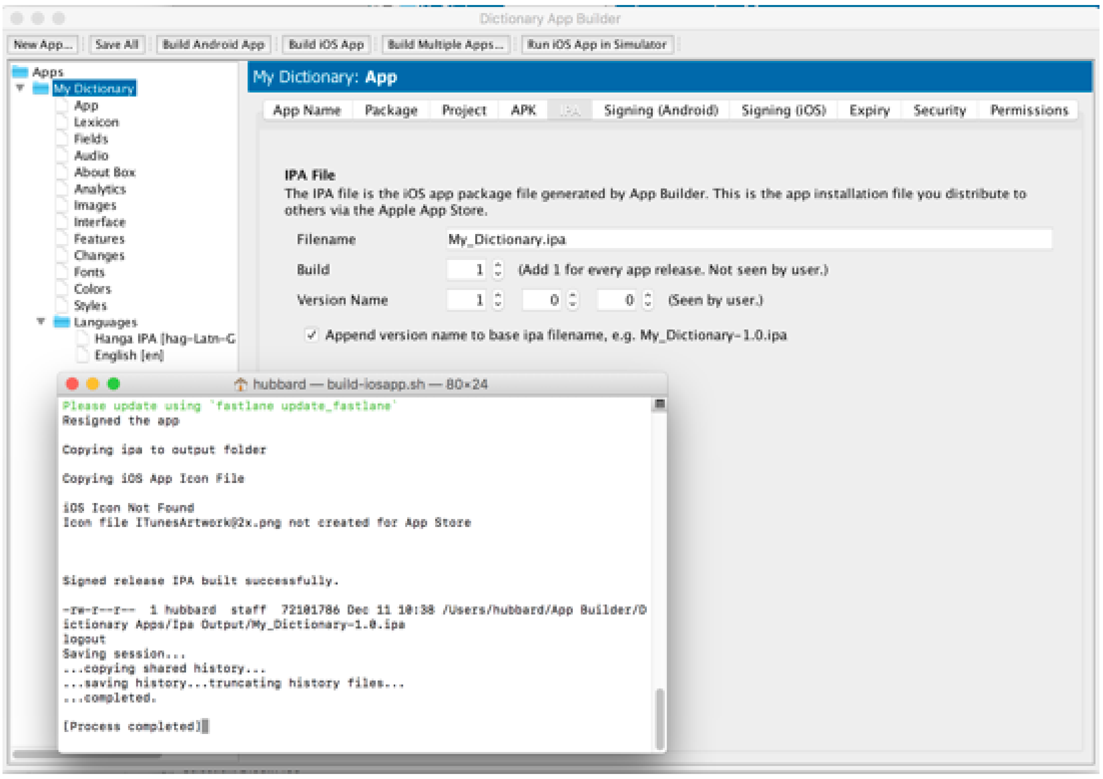
The ***Build*** field referenced as *Build* is also called *Bundle Version String, Bundle Version* or *CFBundleVersion* within Xcode and ITunes Connect. It represents the build number. The *Build* field expects an integer value and should be incremented with each file that is submitted to the ITunes Connect for release or testing.

The ***Version Name*** field is referenced as *Version, Bundle Short Version String, Bundle versions string, short* and *CFBundleShortVersionString* within Xcode and ITunes Connect. The field is created as a concatenation of the values of the three fields separated by a period. If the final field has a value of 0, then the version string is created from just the first two values. So for values of 1, 2 and 0, the resulting string is “1.2”. For values of 1, 2 and 3, the resulting version string is “1.2.3”.

The ***Apple ID*** field is the id assigned by App Store Connect to the application in the app store. It can be obtained from **Apple ID** filed on the App Information tab in the App Store Connect entry for the app.

The ***Bundle ID*** section allows the user to specify a Bundle ID for the app that is different than the package name. By default, the ***Package Name*** field from the ***Package*** tab is used as the Application Bundle ID that is used in creating the provisioning profiles and as the application identifier for the Apple App Store. If the user wants the iOS version of the app to have an identifier that is different than the package name that is used for the Android version of the app, press the ***Specify a Bundle ID*** button and then enter the ID to be used in the text box labelled ***Bundle ID.***

6. Click on the **Build iOS App** button at the top of the screen. A terminal window should open. The build script for the iOS app should run within that terminal window.



My Dictionary: **App**

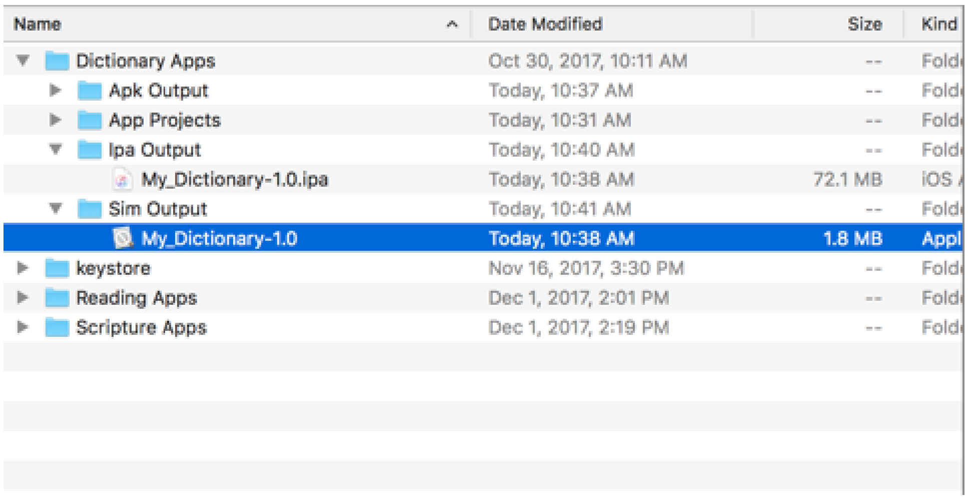
IPA File
The IPA file is the iOS app package file generated by App Builder. This is the app installation file you distribute to others via the Apple App Store.

Filename: My_Dictionary.ipa
Build: 1 (Add 1 for every app release. Not seen by user.)
Version Name: 1 0 0 (Seen by user.)
✓ Append version name to base ipa filename, e.g. My_Dictionary-1.0.ipa

```
Please update using `fastlane update_fastlane`
Resigned the app

Copying ipa to output folder

Copying iOS App Icon File

iOS Icon Not Found
Icon file ITunesArtwork@2x.png not created for App Store



Signed release IPA built successfully.

-rw-r--r--  1 hubbard  staff  72101786 Dec 11 10:38 /Users/hubbard/App Builder/D
ictionary Apps/Ipa Output/My_Dictionary-1.0.ipa
logout
Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.

[Process completed]
```

7. Examine the terminal window once the shell script has been completed. The message “Signed release IPA built successfully” should appear in the window if the app has been built successfully. (Note that occasionally the terminal window will appear behind the Dictionary App Builder and that you have to select the terminal to review the results).

8. The results of the build are an IPA file and an app that can be run in the simulator. They can be found in ~/App Builder/Dictionary Apps/Ipa Output/ and ~/App Builder/Dictionary Apps/Sim Output/.



Name	Date Modified	Size	Kind
▼ Dictionary Apps	Oct 30, 2017, 10:11 AM	--	Fold
▶ Apk Output	Today, 10:37 AM	--	Fold
▶ App Projects	Today, 10:31 AM	--	Fold
▼ Ipa Output	Today, 10:40 AM	--	Fold
My_Dictionary-1.0.ipa	Today, 10:38 AM	72.1 MB	iOS
▼ Sim Output	Today, 10:41 AM	--	Fold
My_Dictionary-1.0	Today, 10:38 AM	1.8 MB	Appl
▶ keystore	Nov 16, 2017, 3:30 PM	--	Fold
▶ Reading Apps	Dec 1, 2017, 2:01 PM	--	Fold
▶ Scripture Apps	Dec 1, 2017, 2:19 PM	--	Fold

8. Testing an iOS App

After building the iOS IPA file, you will want to install it and test it on one or more devices before you submit it for publication to the Apple App Store. This manual describes two ways of doing this:

1. Use **Xcode** to install the IPA file to an iPhone or iPad that is connected to your computer.

   This method is recommended if you have your test devices with you. It does not involve uploading and downloading the IPA to and from the internet.

2. Use **DeployGate** to upload the IPA file to the internet and share it with limited number of devices to download, install and test.

   This method is recommended if you have good internet access and/or you have a team of testers who are elsewhere.

9. Using Xcode to Test an iOS App

To test your iOS IPA file using **Xcode**, do the following:

1. Launch **Xcode** and select **Window** ➢ **Devices and Simulators**.
2. Connect an iPhone or iPad to the Mac using a cable and unlock the device.
3. On the Mac, iTunes may launch and show a dialog asking “Do you want to allow this computer to access information…” Click on the **Cancel** button.
   - Note: This feature can be turned off in iTunes. Select **iTunes** ➢ **Preferences…**, select the **Devices** tab, and click on the **Prevent iPods, iPhone, and iPad from syncing automatically** checkbox.
4. On the device, it may prompt to **Trust This Computer**. Tap on the **Trust** button. This may require to enter the Passcode to trust this computer. The device will show up in the **Xcode Devices** window.
5. The first time the device is connected to the Mac, Xcode will take some time to Prepare debugger support. This may take some time. Please wait.
6. In the **Xcode Devices** window, there will be a section named **INSTALLED APPS**. Click on the **+** button.
7. Find the IPA file to add. Click **Open** after you have selected it.
8. Click the **Install** button next to the name of the app.
9. Wait until the Mac installs the app to your device.

After the install is complete, you will see the app icon on your device and you can test it.

10. Using DeployGate to Test an iOS App

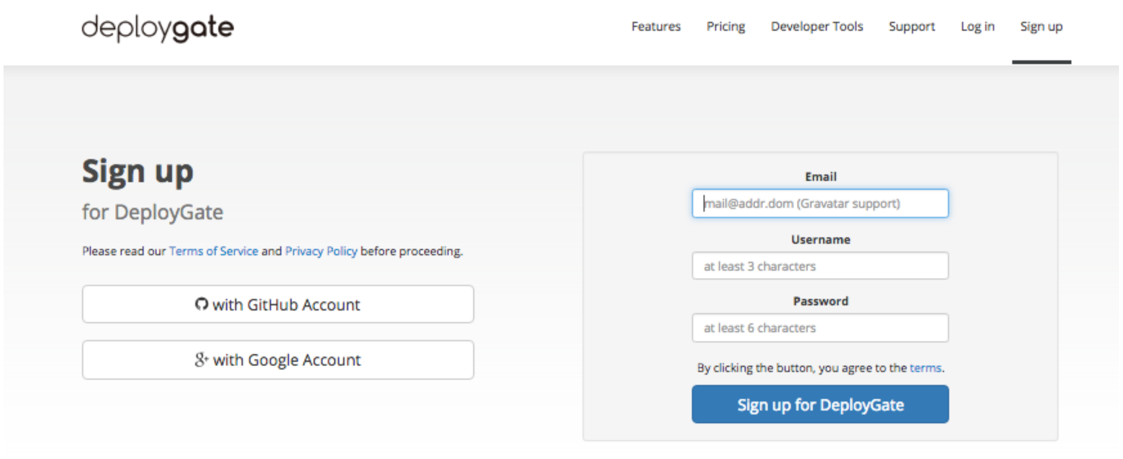
DeployGate enables you to test your app and share it with a limited number of users to test. You upload the IPA file for iOS apps or the APK for Android apps and then download them to your phone or tablet device. You can also invite testers to install your app and help with the testing.

A DeployGate app is installed on the testing device. It will show all of the apps that you have uploaded to DeployGate and allows them to be installed on the device.

10.1. Creating a DeployGate Account

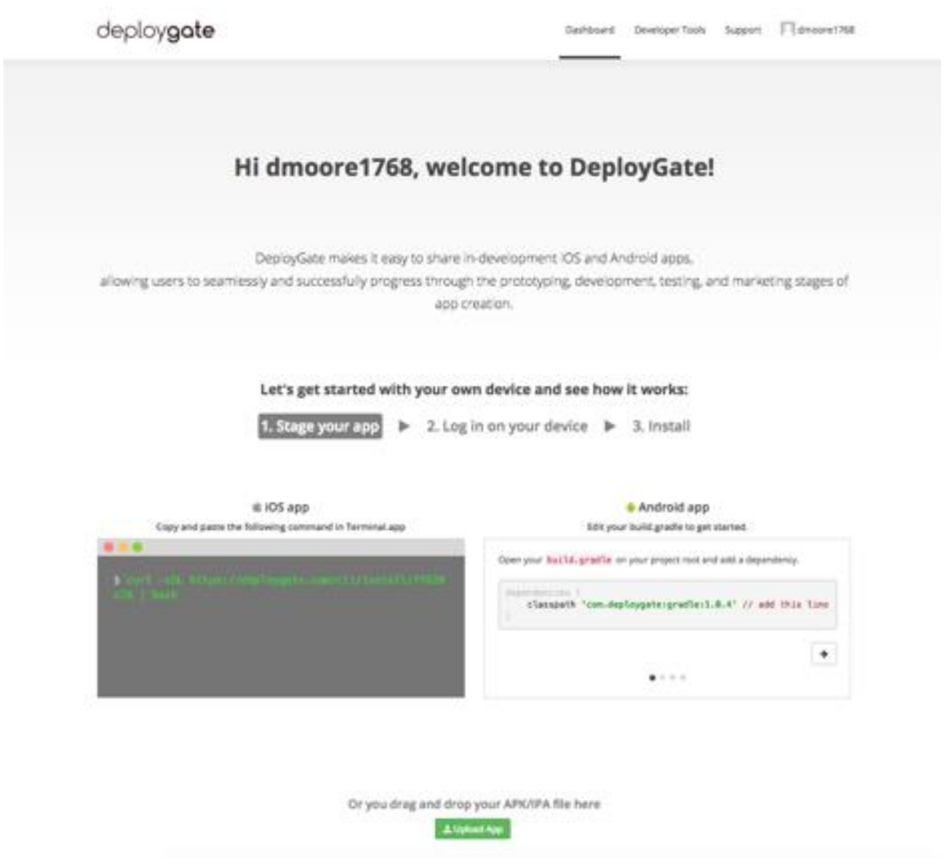
The first step is to create a DeployGate account. To do this:

1. Go to https://deploygate.com
2. Press the **Get Started** button.
3. Enter an email address, a user name and a password.
4. Press the **Sign up for DeployGate** button.



10.2. Uploading Your First App

Once the sign in screen has been successfully completed, you are presented with a screen that prompts you to upload your app. While there are several methods described for embedding it as part of your build process, the way we have used this to date is to simply upload the IPA file that has been created by locating the ipa file in Finder and then dragging it to the bottom area of the screen where it has a green **Upload App** area:



Hi dmoore1768, welcome to DeployGate!

DeployGate makes it easy to share in-development iOS and Android apps, allowing users to seamlessly and successfully progress through the prototyping, development, testing, and marketing stages of app creation.

Let's get started with your own device and see how it works:

1. Stage your app ▶ 2. Log in on your device ▶ 3. Install

iOS app

Copy and paste the following command in Terminal.app

```
$ curl -sSL https://deploygate.com/[illegible] | bash
```

Android app

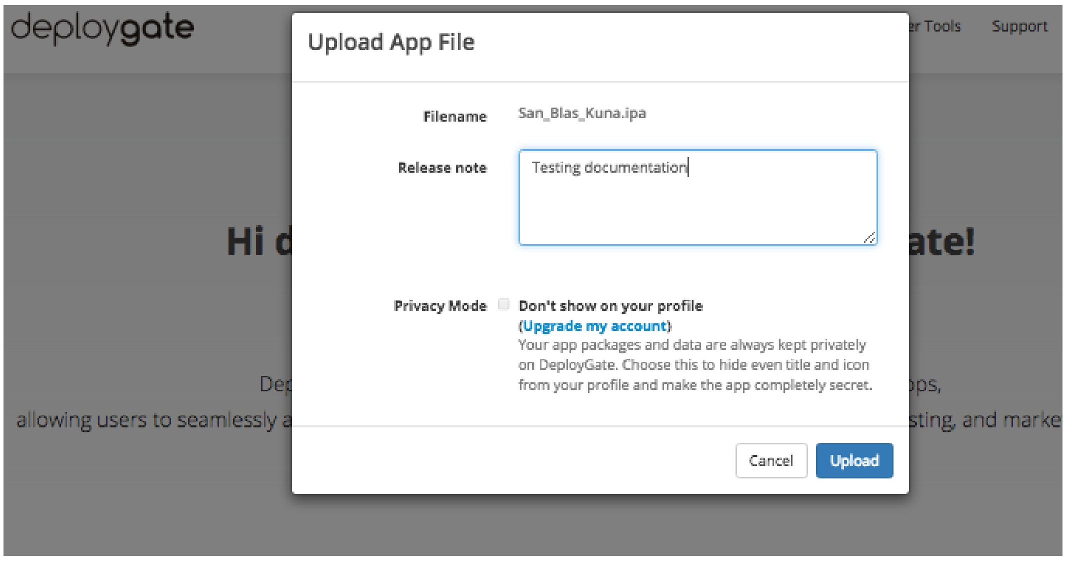
Edit your build.gradle to get started.

Open your `build.gradle` on your project root and add a dependency.

```
dependencies {
    classpath 'com.deploygate:gradle:1.0.4' // add this line
}
```

Or you drag and drop your APK/IPA file here

Dragging the file to the upload area causes a new dialog to be displayed with the name of the file and a text box where you can enter a short note that will be displayed on your profile window and also on the DeployGate app when the user is selecting the app. It is a good place to write a short note on the reason for the update so that it is easy for the testers to see that the app has been updated and to see what the primary reason they need to update is. Complete the screen and press the **Upload** button.



Upload App File

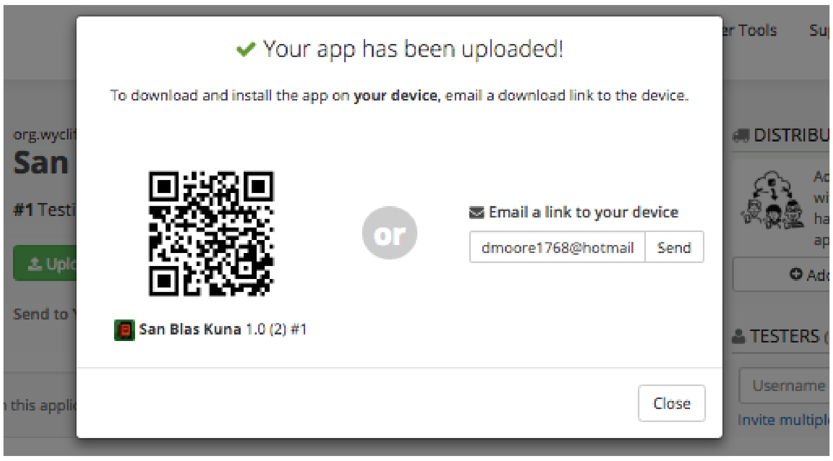
Filename San_Blas_Kuna.ipa

Release note: Testing documentation

Privacy Mode ☐ Don't show on your profile
(Upgrade my account)
Your app packages and data are always kept privately on DeployGate. Choose this to hide even title and icon from your profile and make the app completely secret.

When the upload is complete, a new dialog is displayed with a QCode bar code and the option to send an email to your device. The QCode can be read in with your iPhone or iPad which will trigger an installation of the DeployGate app using your app profile.

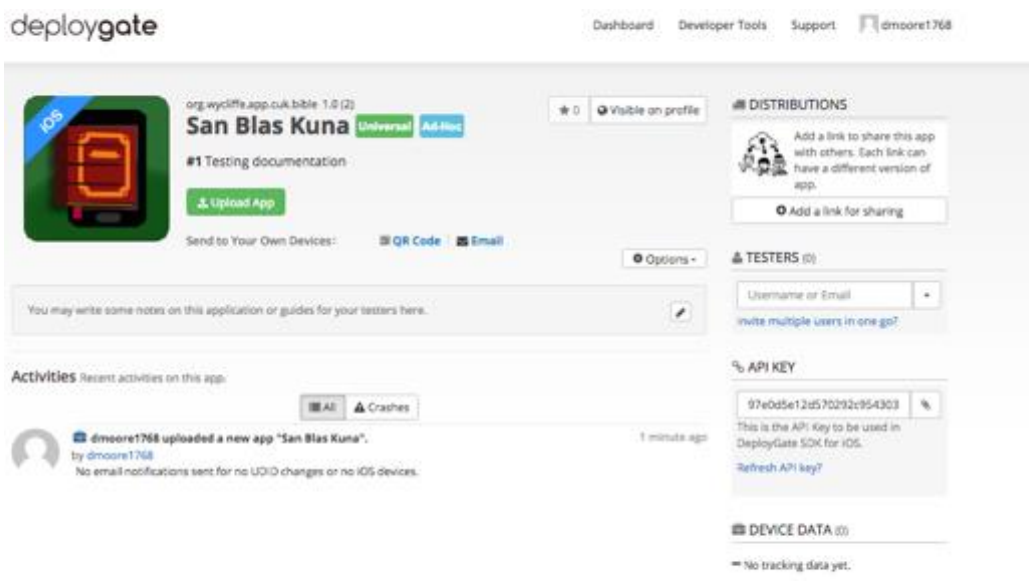
Alternatively, you can enter an email address at this point, which you would also open on the iPhone to install the DeployGate app with the correct profile. Or you can simply go on and add users and devices later.



✓ Your app has been uploaded!

To download and install the app on **your device**, email a download link to the device.

After this, the screen that is displayed is what you will normally see when you login. The screen has an entry for each app that you have uploaded. It has an **Upload App** button that can be used to upload new versions of the same app or to upload a new app.



org.wycliffe.app.cuk.bible 1.0 (2)

San Blas Kuna

#1 Testing documentation

Send to Your Own Devices:

You may write some notes on this application or guides for your testers here.

Activities Recent activities on this app.

dmoore1768 uploaded a new app "San Blas Kuna". 1 minute ago
by dmoore1768
No email notifications sent for no UDID changes or no iOS devices.

DISTRIBUTIONS

Add a link to share this app with others. Each link can have a different version of app.

TESTERS (0)

API KEY

97e0d5e12d570292c954303

This is the API Key to be used in DeployGate SDK for iOS.

DEVICE DATA (0)

No tracking data yet.

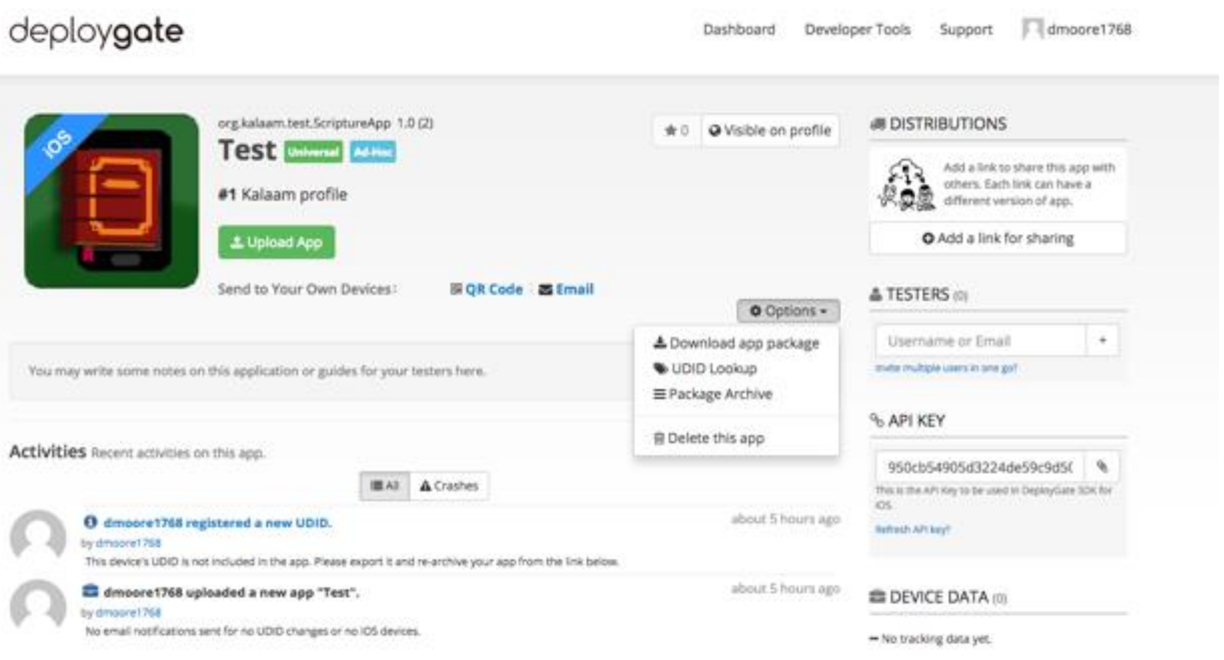
10.3. Registering a Device

If the iOS device was not originally in the mobile provisioning profile and if the device has not been previously registered in your Apple Developer account, you need to add it to both.

There is a method for manually doing this, but DeployGate provides a way of simplifying the process so that you don't need to go and look up UDID for the device.

First, try to install the app using the method described above. You will not be able to install it because the UDID is not registered for the app. However this will result in the device being registered in DeployGate which allows the following steps.

After attempting to install the app, re-enter DeployGate in your browser and open the entry for your app. As you can see in the screen below, it will show that a new UDID has been registered for the device. Press the **Options** button below and select the **Package Archive** option. Next click the little tag symbol inside the app box to open the UDID list.



Add a link to share this app with others. Each link can have a different version of app.

Invite multiple users in one go?

950cb54905d3224de59c9d5(

This is the API Key to be used in DeployGate SDK for iOS.

You may write some notes on this application or guides for your testers here.

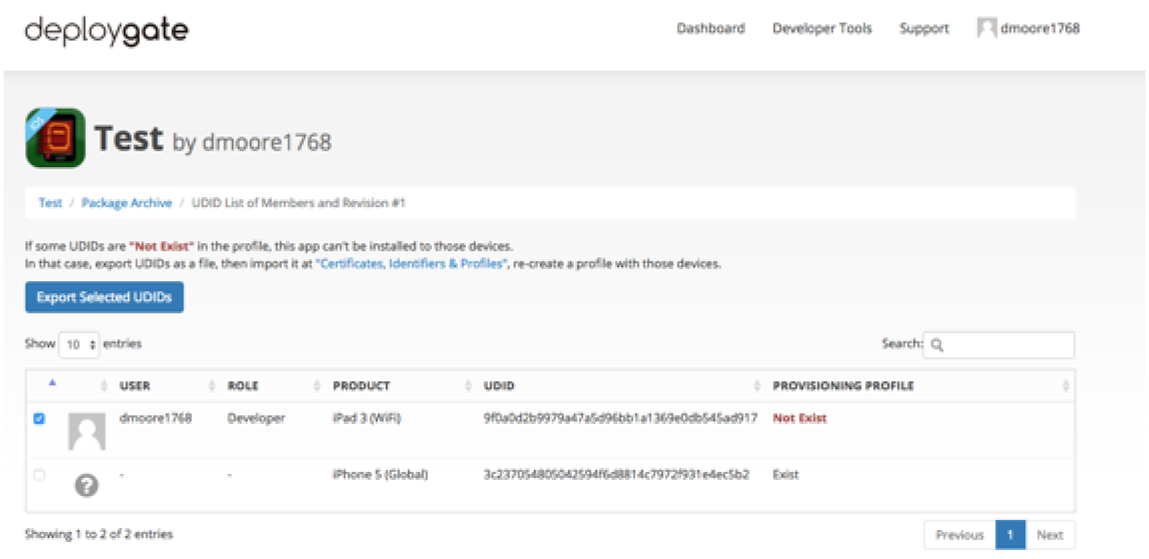
Activities Recent activities on this app.

dmoore1768 registered a new UDID. about 5 hours ago
by dmoore1768
This device's UDID is not included in the app. Please export it and re-archive your app from the link below.

dmoore1768 uploaded a new app "Test". about 5 hours ago
by dmoore1768
No email notifications sent for no UDID changes or no IOS devices.

— No tracking data yet.

The next screen shows a list of the devices that have been observed by DeployGate or that were included in the provisioning profile. Your new device entry will show up on the screen with a **Not Exist** entry. Make sure that the entry for the new device is checked and then press the **Export Selected UDIDs** button. This will create a file "multiple-device-upload-ios.txt" that can be used on the Apple Developer website to add these devices to the mobile provisioning file.



Test by dmoore1768

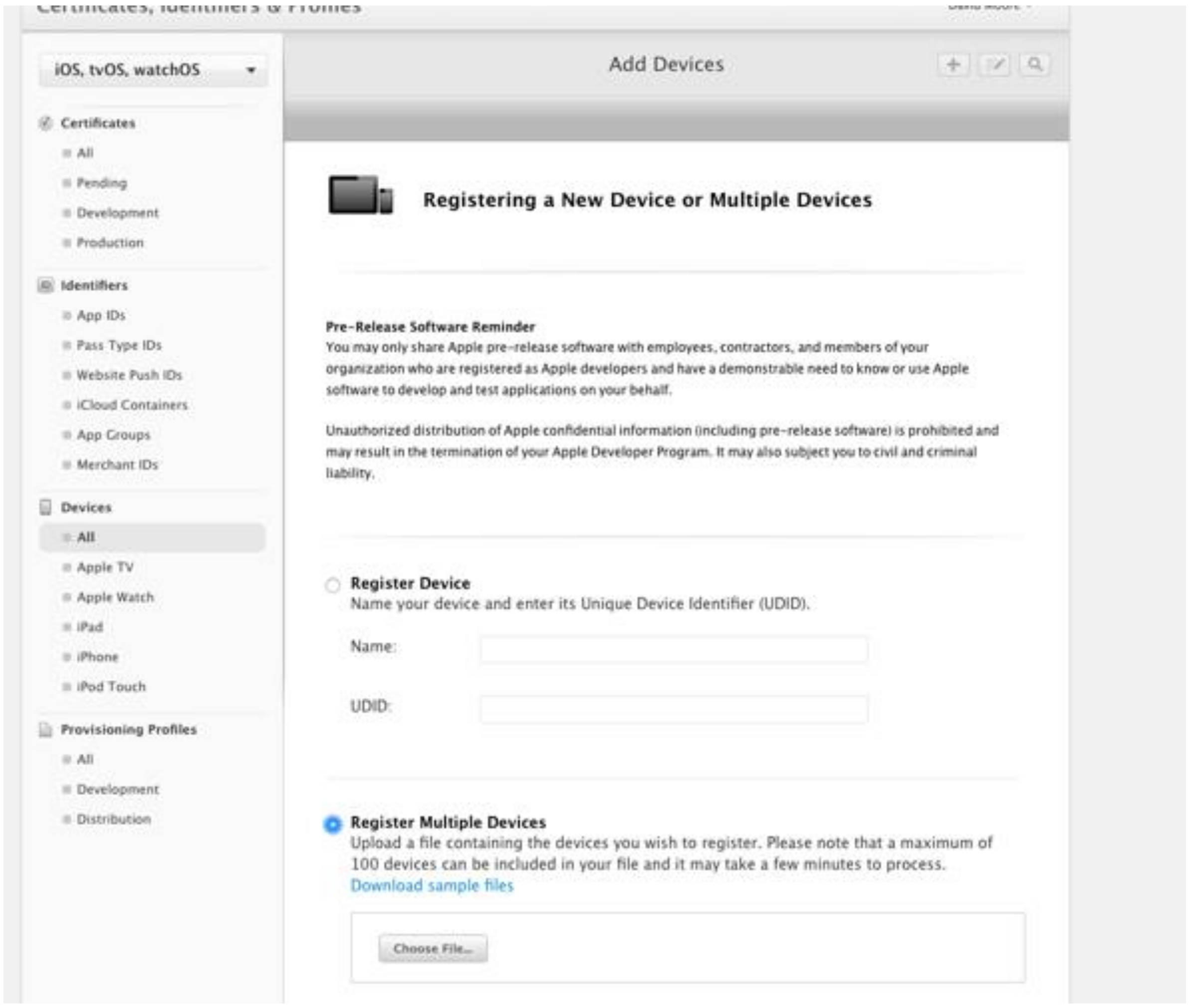
Test / Package Archive / UDID List of Members and Revision #1

If some UDIDs are "Not Exist" in the profile, this app can't be installed to those devices.
In that case, export UDIDs as a file, then import it at "Certificates, Identifiers & Profiles", re-create a profile with those devices.

		USER	ROLE	PRODUCT	UDID	PROVISIONING PROFILE
✓		dmoore1768	Developer	iPad 3 (WiFi)	9f0a0d2b9979a47a5d96bb1a1369e0db545ad917	Not Exist
		-	-	iPhone 5 (Global)	3c237054805042594f6d8814c7972f931e4ec5b2	Exist

Showing 1 to 2 of 2 entries

After logging in to Apple Developer, click on **Certificates, IDs and Profiles**. Press the **All** selection under **Devices** as shown in the illustration below and then select **Register Multiple Devices**. Then press the **Choose File** button.



Registering a New Device or Multiple Devices

Pre-Release Software Reminder
You may only share Apple pre-release software with employees, contractors, and members of your organization who are registered as Apple developers and have a demonstrable need to know or use Apple software to develop and test applications on your behalf.

Unauthorized distribution of Apple confidential information (including pre-release software) is prohibited and may result in the termination of your Apple Developer Program. It may also subject you to civil and criminal liability.

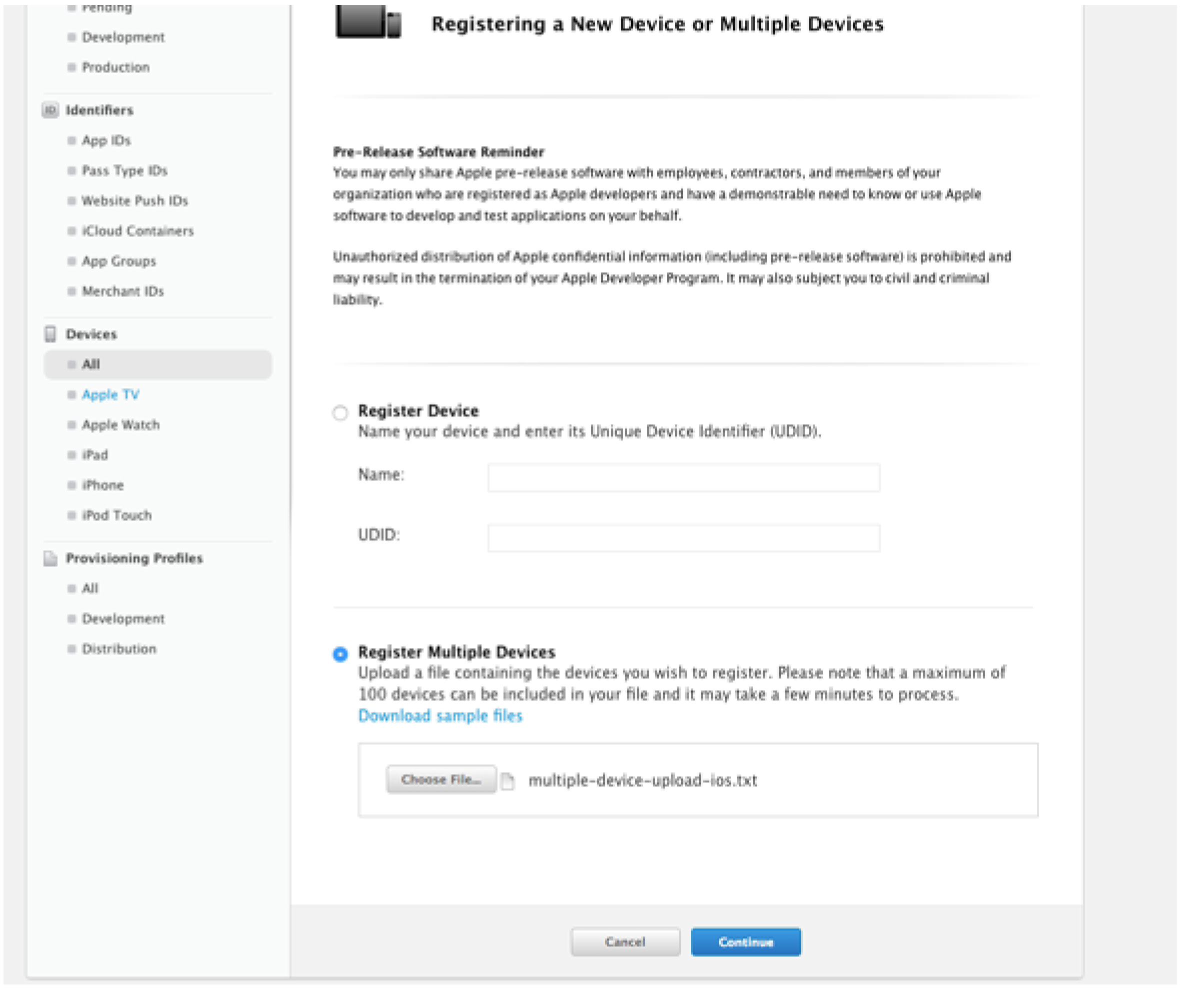
Register Device
Name your device and enter its Unique Device Identifier (UDID).

Name:

UDID:

[✓ Register Multiple Devices]
Upload a file containing the devices you wish to register. Please note that a maximum of 100 devices can be included in your file and it may take a few minutes to process.
Download sample files

Find the multiple-device-upload-ios.txt file that was created by DeployGate and then press **Continue**.



Registering a New Device or Multiple Devices

Pre-Release Software Reminder
You may only share Apple pre-release software with employees, contractors, and members of your organization who are registered as Apple developers and have a demonstrable need to know or use Apple software to develop and test applications on your behalf.

Unauthorized distribution of Apple confidential information (including pre-release software) is prohibited and may result in the termination of your Apple Developer Program. It may also subject you to civil and criminal liability.

Register Device
Name your device and enter its Unique Device Identifier (UDID).

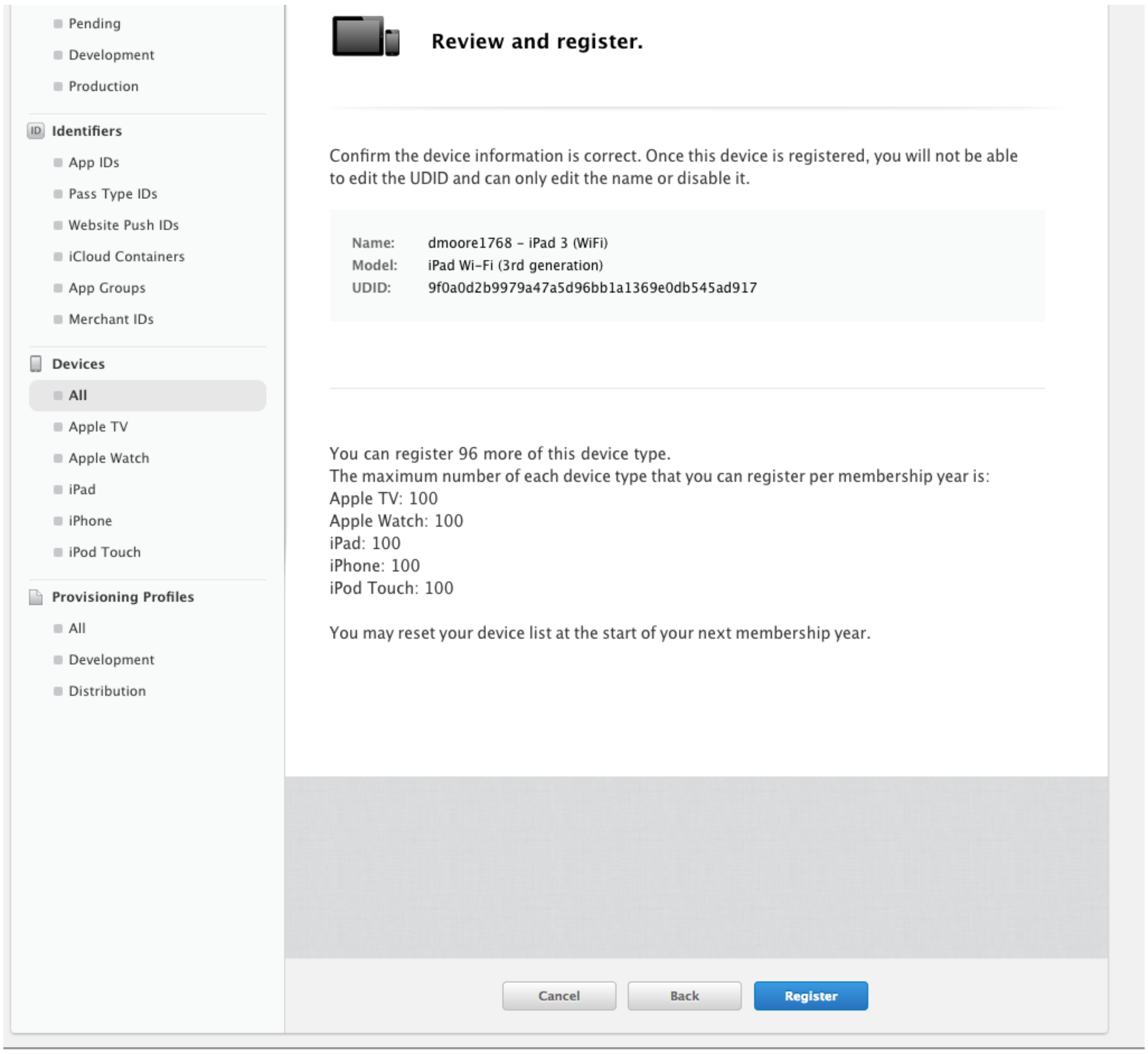
Name:

UDID:

[✓ **Register Multiple Devices**]
Upload a file containing the devices you wish to register. Please note that a maximum of 100 devices can be included in your file and it may take a few minutes to process.
Download sample files

Choose File... multiple-device-upload-ios.txt

A review screen will be displayed which should list the name to be assigned to the device along with the UDID associated with the device. Review to ensure this is correct and press **Register**.



Review and register.

Confirm the device information is correct. Once this device is registered, you will not be able to edit the UDID and can only edit the name or disable it.

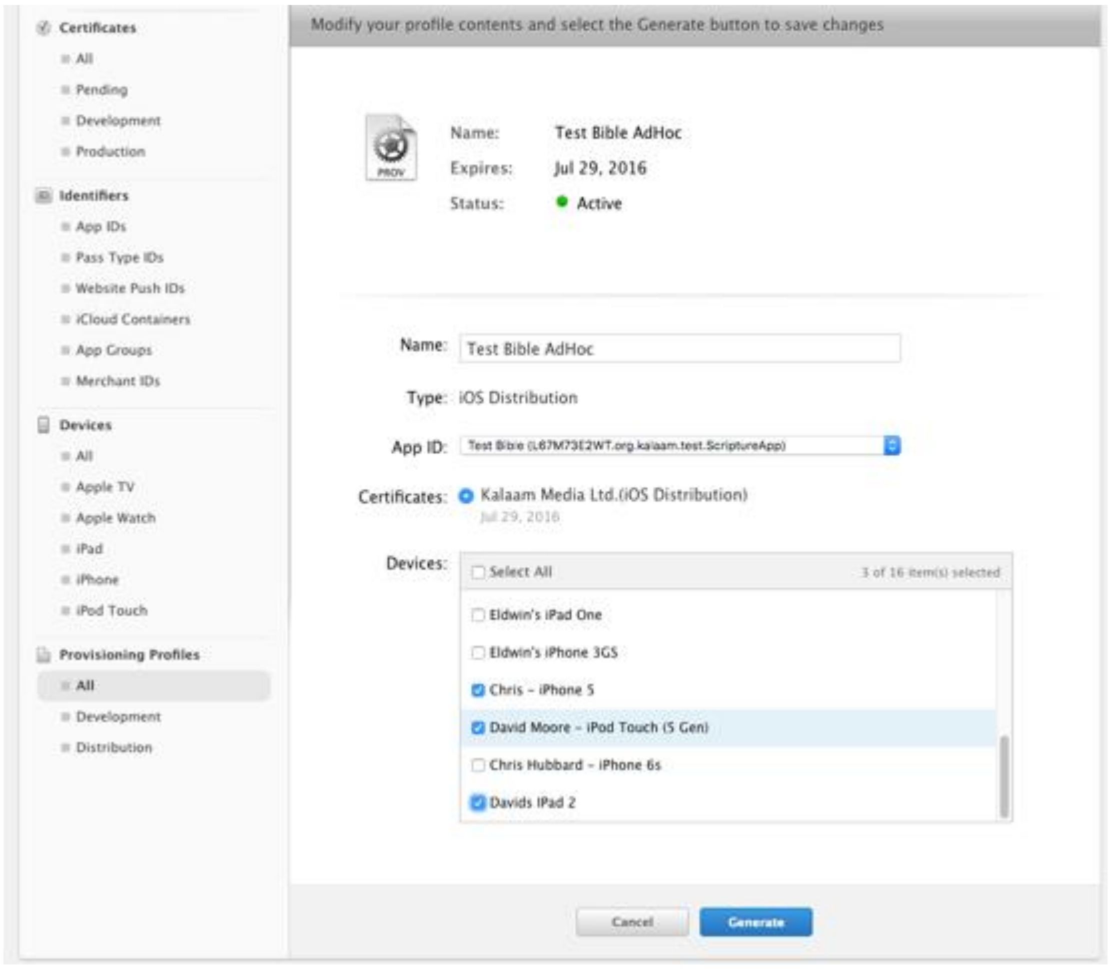
Name:	dmoore1768 – iPad 3 (WiFi)
Model:	iPad Wi-Fi (3rd generation)
UDID:	9f0a0d2b9979a47a5d96bb1a1369e0db545ad917

You can register 96 more of this device type.
The maximum number of each device type that you can register per membership year is:
Apple TV: 100
Apple Watch: 100
iPad: 100
iPhone: 100
iPod Touch: 100

You may reset your device list at the start of your next membership year.

At this point your device has been registered to your Apple Developer account. You now need to add the device to the provisioning profile for your app. Select the provisioning profile being used to test the app. Make sure your test device is checked in the list of devices at the bottom of the screen. Press **Generate**.

Note: For the purposes of testing with DeployGate, an AdHoc type of provision profile must be created and used. If you have not selected AdHoc, the list of devices will not be available on the screen.

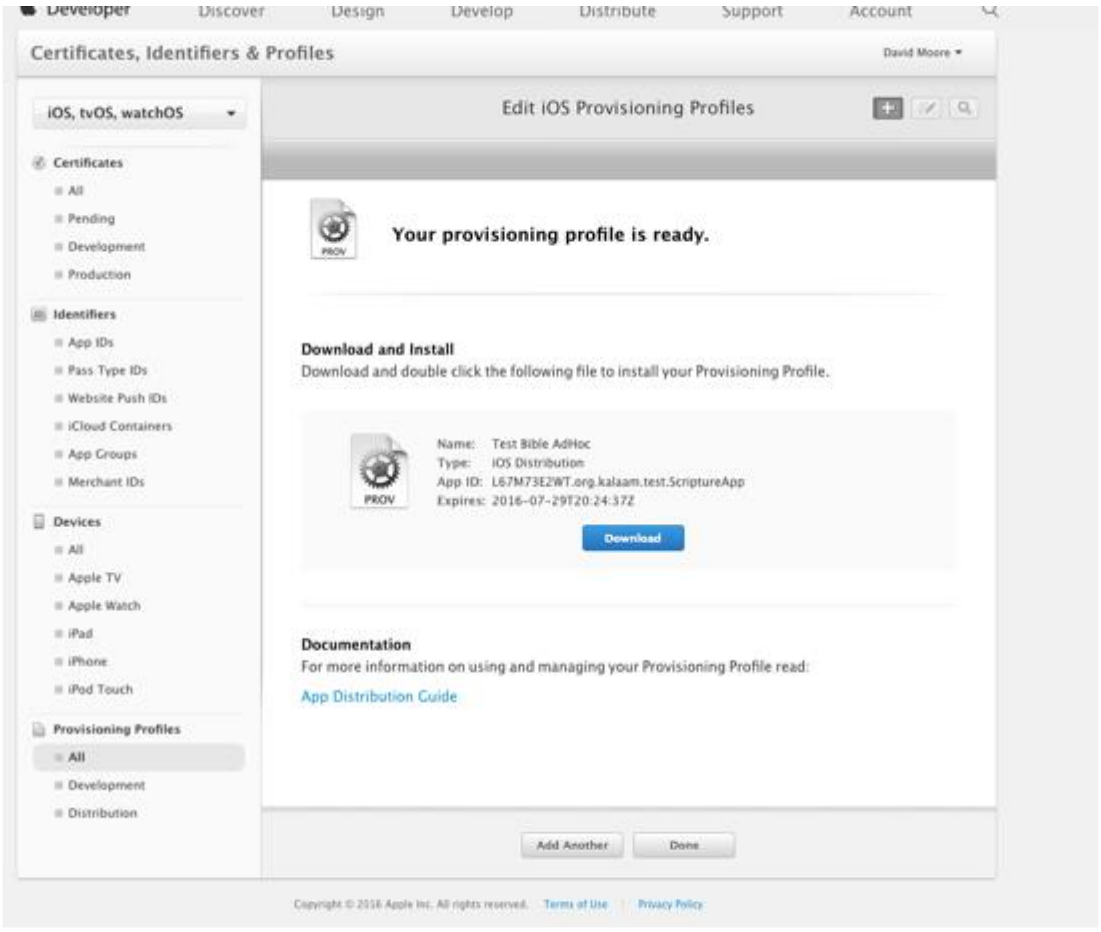


Modify your profile contents and select the Generate button to save changes

Name: Test Bible AdHoc
Expires: Jul 29, 2016
Status: ● Active

Name: Test Bible AdHoc
Type: iOS Distribution
App ID: Test Bible (L67M73E2WT.org.kalaam.test.ScriptureApp)
Certificates: ● Kalaam Media Ltd.(iOS Distribution)
Jul 29, 2016
Devices: ☐ Select All — 3 of 16 item(s) selected
☐ Eldwin's iPad One
☐ Eldwin's iPhone 3GS
☑ Chris – iPhone 5
☑ David Moore – iPod Touch (5 Gen)
☐ Chris Hubbard – iPhone 6s
☑ Davids iPad 2

The following screen will display:



Edit iOS Provisioning Profiles

Your provisioning profile is ready.

Download and Install

Download and double click the following file to install your Provisioning Profile.

Name: Test Bible AdHoc
Type: iOS Distribution
App ID: L67M73E2WT.org.kalaam.test.ScriptureApp
Expires: 2016-07-29T20:24:37Z

Documentation

For more information on using and managing your Provisioning Profile read:

App Distribution Guide

Copyright © 2016 Apple Inc. All rights reserved.

Now you can download the new provisioning profile that has been generated.

Next you need to rebuild the iOS app using the new profile and upload it to DeployGate again. This time when you attempt to install it, the **Install** button should be enabled and will allow you to install your app.

11. Uploading an iOS App to Apple App Store

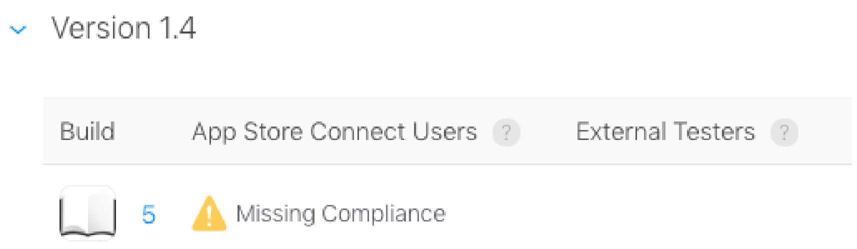
Before attempting to upload the app, you will need to create an entry in App Store Connect (https://appstoreconnect.apple.com).

Select the appropriate distribution **Signing Identity** and **App Store** provisioning profile on the **Signing (iOS)** tab and click Build iOS App. This will create an IPA file in the IPA Output Folder.

Launch **Transporter** and click **Sign In** using the Apple ID for your Apple Developer Account. Drag and drop the IPA file from the IPA Output Folder to Transporter. Once the app is added to Transporter, click on the **Deliver** button. After it is uploaded, you can click on the **…** button and select **View in App Store Connect**. Selecting the Activity tab will show the status of the processing of the upload.

12. Using Test Flight to Test an iOS App

It will take a little bit of time (around 20 minutes) for the app to be processed in App Store Connect. You will receive an email when the app is done processing. Switching to the Test Flight tab in App Store Connect, you will see that app is Missing Compliance.

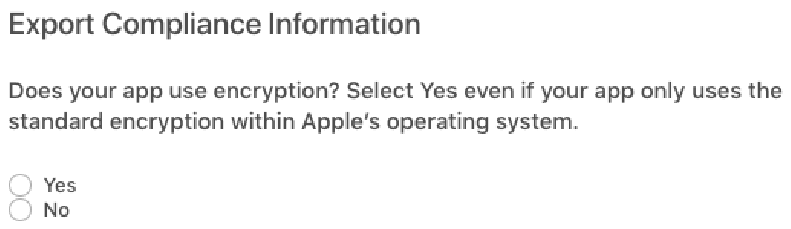


Version 1.4

Build	App Store Connect Users	External Testers
5	Missing Compliance	

Click on the build number and you will be taken to a page where you can click on **Provide Export Compliance Information**. The app uses an encryption algorithm to protect the text of the app.

Click **Yes** to the first dialog that the app uses encryption and then click **Next**.

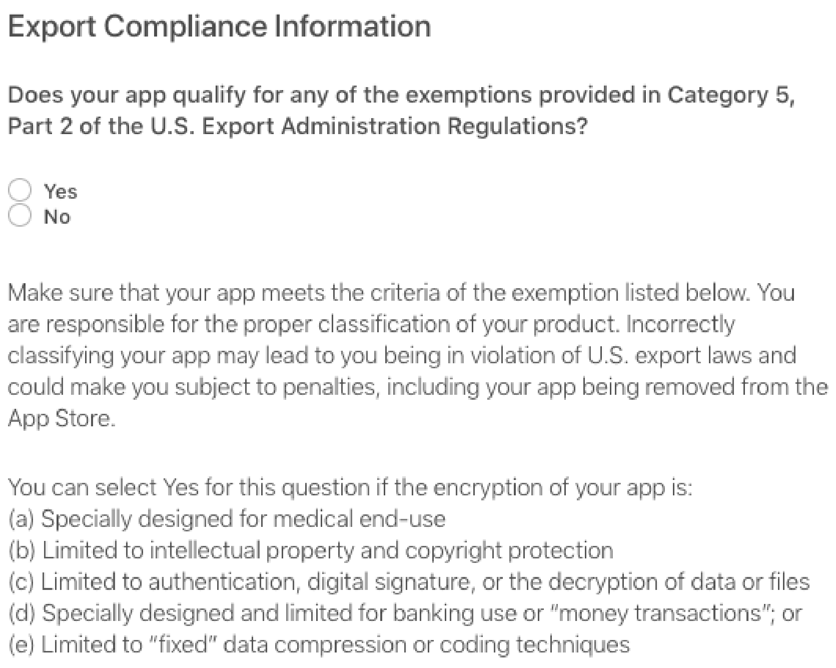


Export Compliance Information

Does your app use encryption? Select Yes even if your app only uses the standard encryption within Apple's operating system.

Yes
No

Click **Yes** to the second dialog to indicate that the app qualifies for an exemption due to **(b) Limited to intellectual property and copyright protection** and then click **Start Internal Testing**.



Export Compliance Information

Does your app qualify for any of the exemptions provided in Category 5, Part 2 of the U.S. Export Administration Regulations?

Yes
No

Make sure that your app meets the criteria of the exemption listed below. You are responsible for the proper classification of your product. Incorrectly classifying your app may lead to you being in violation of U.S. export laws and could make you subject to penalties, including your app being removed from the App Store.

You can select Yes for this question if the encryption of your app is:
(a) Specially designed for medical end-use
(b) Limited to intellectual property and copyright protection
(c) Limited to authentication, digital signature, or the decryption of data or files
(d) Specially designed and limited for banking use or "money transactions"; or
(e) Limited to "fixed" data compression or coding techniques

You can add App Store Connect users (normally users in your organization) to test the app. There is a link at the left for **Add External Testers**. This will require the app to go through Beta App Review.

13. Building from Terminal

Dictionary App Builder has a command line interface which allows you to create a new app and build it, or load an existing app and build it.

The base command calls java to access the jar file within the Dictionary App Builder application followed by a series of options described below. The base command is:

```
java -jar "/Applications/Dictionary App Builder.app/Contents/Java/bin/reading-app-builder.jar"
```

The available parameters are:

Option	Description
-new	Create a new app project
-load <project>	Load an existing app project
-build	Build app project (use with either -new or -load)
-no-save	Do not save changes to app (use with -load)
-resign	Resign iOS Template App (use with either -new or -load)
-?	Show command line help
-n <app-name>	Set app name. Enclose the name in "double quotes" if it contains spaces.
-p <package-name>	Set package name, e.g. com.myorg.language.appname
-b <filename>	Add book or bundle file. This could be a USFM file or a zipped set of USFM files. It could also be a Digital Bible Library text release bundle.
-i <filename>	Include additional parameters file. Use the full path of the file and enclose it in "double quotes" if there is a space in the path.
-a <filename>	Set about box text, contained in text file.

	Use the full path of the file and enclose it in "double quotes" if there is a space in the path.
-f <fontname>	Set font name or filename, e.g. "Charis SIL Compact", "c:\fonts\myfont.ttf" The font name must be one of the items in the list of fonts in the New App wizard. For other fonts, specify the full path to the font filename.
-g	Use Grandroid
-ic <filename>	Add launcher icon (one or more .png files). Use the full path of the files and enclose them in "double quotes" if there is a space in the path.
-l <lang-code>	Set language for menu items and settings, e.g. en, fr, es
-ft <feature=value>	Set a feature, e.g. book-select=grid
-vc <integer>	Set version code, e.g. 1, 2, 3, or +1 to increment the current version code by 1.
-vn <string>	Set version name, e.g. 1.0, 2.1.4, or use +1, +0.1, +0.0.1 to increment the current value.
-ks <filename>	Set keystore filename. Use the full path of the file and enclose it in "double quotes" if there is a space in the path.
-ksp <password>	Set keystore password
-ka <alias>	Set key alias
-kap <password>	Set key alias password
-fp <folder=path>	Set a folder path, e.g. "app.builder=/Developer/Dictionary App Builder".
-ta <target-api>	Set Target API, e.g. 21 for Android 5.0, 22 for Android 5.1.
-si <signing identity>	Set Signing Identity to use for iOS Resigning
-pp <provisioning profile>	Set full path to provisioning profile for iOS resigning
-bn <integer>	Set build number for ipa file, e.g. 1, 2, 3, or +1 to increment by 1

-vs <string>	Set version string for ipa file, e.g. 1.0, 2.1.4 or +1, +0.1, +0.0.1

Examples:

```
Java -jar

"/Applications/Dictionary App Builder.app/Contents/Java/bin/reading-app-
builder.jar" -load "Mali" -resign -bn "5" -vs "2.3.2" -si "iPhone Distribution:
Summer Institute of Linguistics, Inc (SIL) (4YF5X97M4H) " -pp
"/Users/builder/Documents/MobileProvision/AdHoc_org.wycliffe.app.mali.mobileprovisi
on"
```