



# BuzzTouch iOS Push Notifications

# BuzzTouch iOS Push Notifications

## 1 Introduction

- 1.1 An overview of Apple's Push Notifications 5

## 2 On the Mac with Keychain Access utility

- 2.1 Create a Certificate Request 8

## 3 On Apple's Developer Provisioning Portal

- 3.1 Log into Apple's Developer Provisioning Portal 16
- 3.2 Create an App ID with Push Notifications enabled 19
- 3.3 Create APS Certificate 22
- 3.4 Create .P12 certificate 30
- 3.5 Convert .P12 file into an .PEM file 35
- 3.6 Create and install Provisioning Profile 38

## 4 Settings for BuzzTouch Control Panel

- 4.1 Core settings for Push Notifications 44
- 4.2 Uploading .PEM Certificate into BuzzTouch 48
- 4.3 Download your App's source code from BuzzTouch server 53

## 5 Initial App setup in Xcode

- 5.1 Open the App in Xcode 58
- 5.2 Modify Bundle Identifier 62

## 6 Use Xcode to load App onto iPhone

- 6.1 Setup Xcode Project to use new Provisioning Profile 66
- 6.2 Run app on iPhone for testing of Push Notifications 68

**7 Send a notification from BuzzTouch Control Panel**

7.1 Send Push Notifications 72

7.2 Check the iPhone for a push notification 77

**8 Production usage of Push Notifications**

8.1 Production Push Notifications 80

# Introduction

## **ABOUT THIS TUTORIAL**

We walk you through the necessary steps to get your App ready to receive Push Notifications for alerting Users to items about an app:

- Setting up and configuring the BuzzTouch Control Panel
- Making changes to the downloaded code from the BuzzTouch packager
- Creating and obtaining Push Certificates from Apple's Provisioning Portal
- Composing, sending and testing Push Notifications to registered devices
- Changes to make for Production push notifications

## **OVERVIEW OF PUSH NOTIFICATIONS**

Push notifications are ways for an (non-foreground) application to let its users know it has information for them. The information could be:

- a message,
- an impending calendar event, or
- new data on a remote server.

Push notifications can:

- display an alert message,
- badge the application icon and
- play a sound

When users are notified that the application has a message, event, or other data for them, they can launch the application and see the details. They can also choose to ignore the notification, in which case the application is not activated nor launched.

## **DETAILS ON THE PUSH NOTIFICATION SERVICE**

Push notifications—also known as remote notifications—arrive from outside a device. They originate on a BuzzTouch server—the application's provider—and are pushed to applications on devices (via the Apple Push Notification service) when there are messages to see or data to download. To receive push notifications, an application must register to receive the notifications and then pass to BuzzTouch a device token it gets from the operating system.

Apple Push Notification service (APNs) propagates push notifications to devices having applications registered to receive those notifications. Each device establishes an accredited and encrypted IP connection with the service and receives notifications over this persistent connection.

The BuzzTouch server connects with APNs through a persistent and secure channel while monitoring incoming data intended for the applications. When new data for an application arrives, the BuzzTouch server prepares and sends a notification through the channel to APNs, which pushes the notification to the target device.

(The details were excerpted from Apple's article on "[About Local and Push Notifications](#)" )

## **QUICK TIPS ON USING THIS TUTORIAL**

For easy browsing of the PDF, try these techniques with the Preview or the Adobe Reader application on the Mac:

- Enable the Sidebar: View menu, show Table of Contents
- Show all items in Sidebar: View menu, expand all
- Search for a phrase or term: Command-F and view results in Sidebar

## **VERSION HISTORY**

- 1.0 Initial release (Jan 27, 2013)
- 1.1 Moved creation of .PEM certificate to the beginning for overall chronological sequencing of steps (Jan 27, 2013)

# **On the Mac with Keychain Access utility**

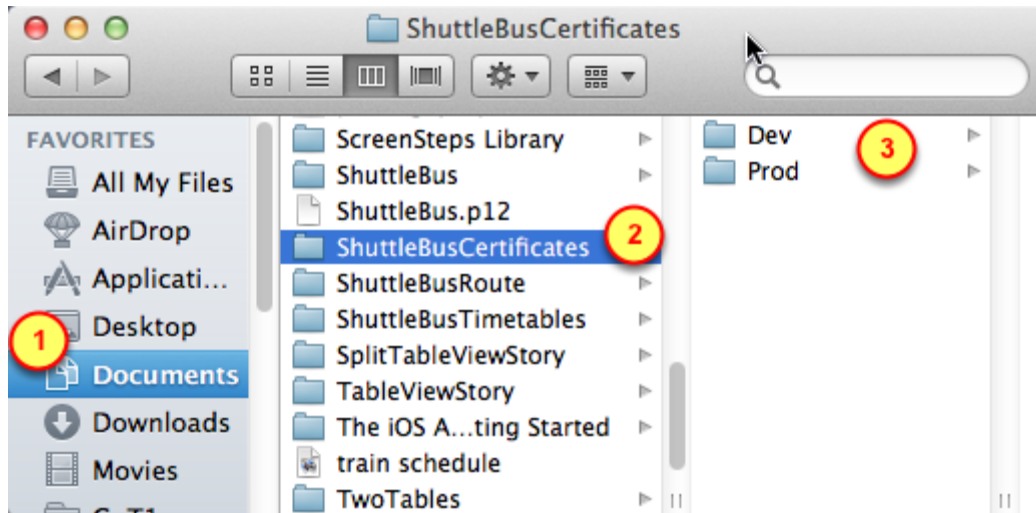
## Create a Certificate Request

---

We have to request a Certificate from Apple to do the Push Notifications for the App.

Follow these steps to create a **Certificate Request**.

### 1. Create a folder to save the App-specific files

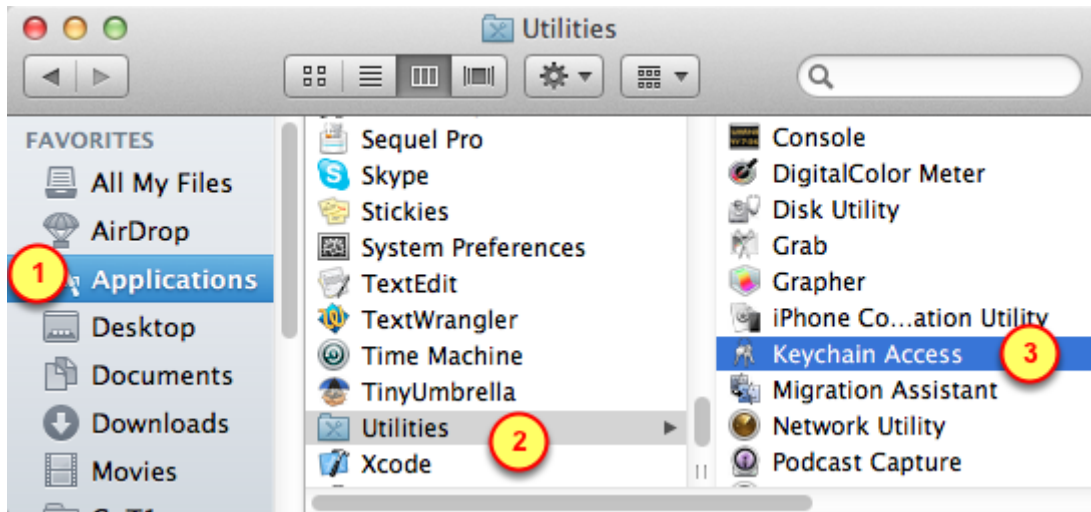


Create a folder on your computer to save some important files. It's important that you are able to locate this folder later and in the future. When you're done completing this process, do not move it to the trash.

1. It is best to keep this folder next to the App's Xcode folder.
2. The name of the folder should be the name of the **App** along with the **Certificates** designation.
3. Within that folder, create two new folders named **Dev** and **Prod**



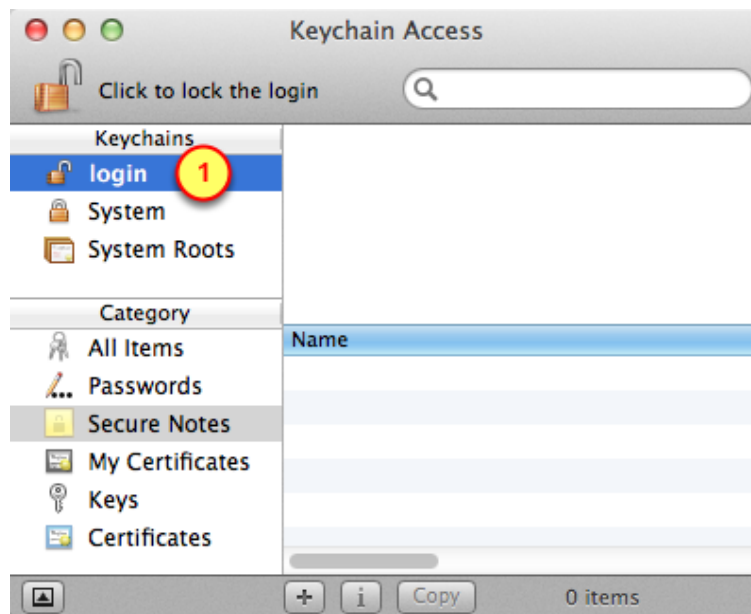
## 2. Open the Keychain Access application on the Mac



Find and launch the Keychain Access application

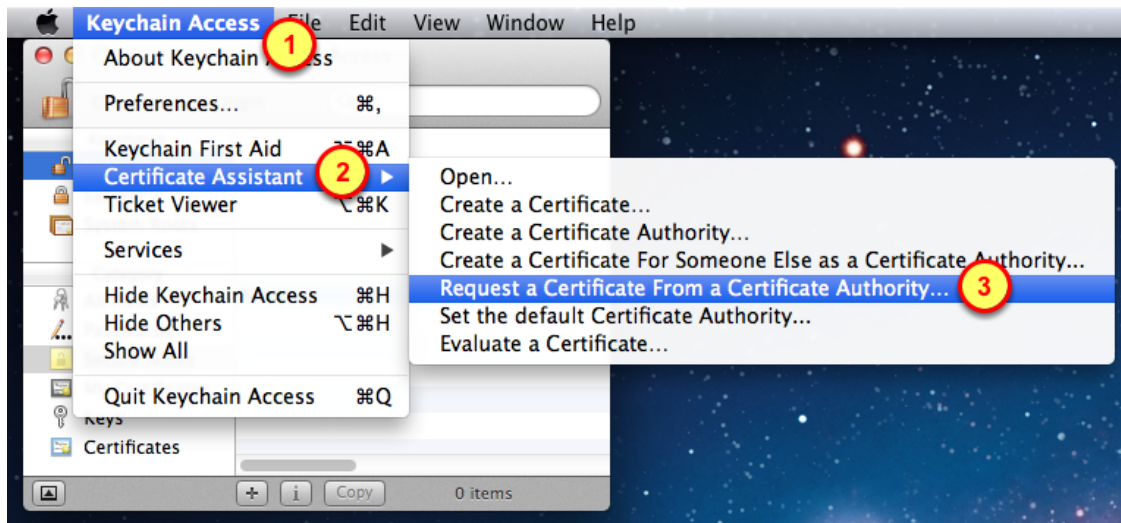
1. Go into the **Applications** folder
2. Go into the **Utilities** folder
3. Launch the **Keychain Access** application

## 3. Select the Login keychain



1. Click on the **Login** keychain item.

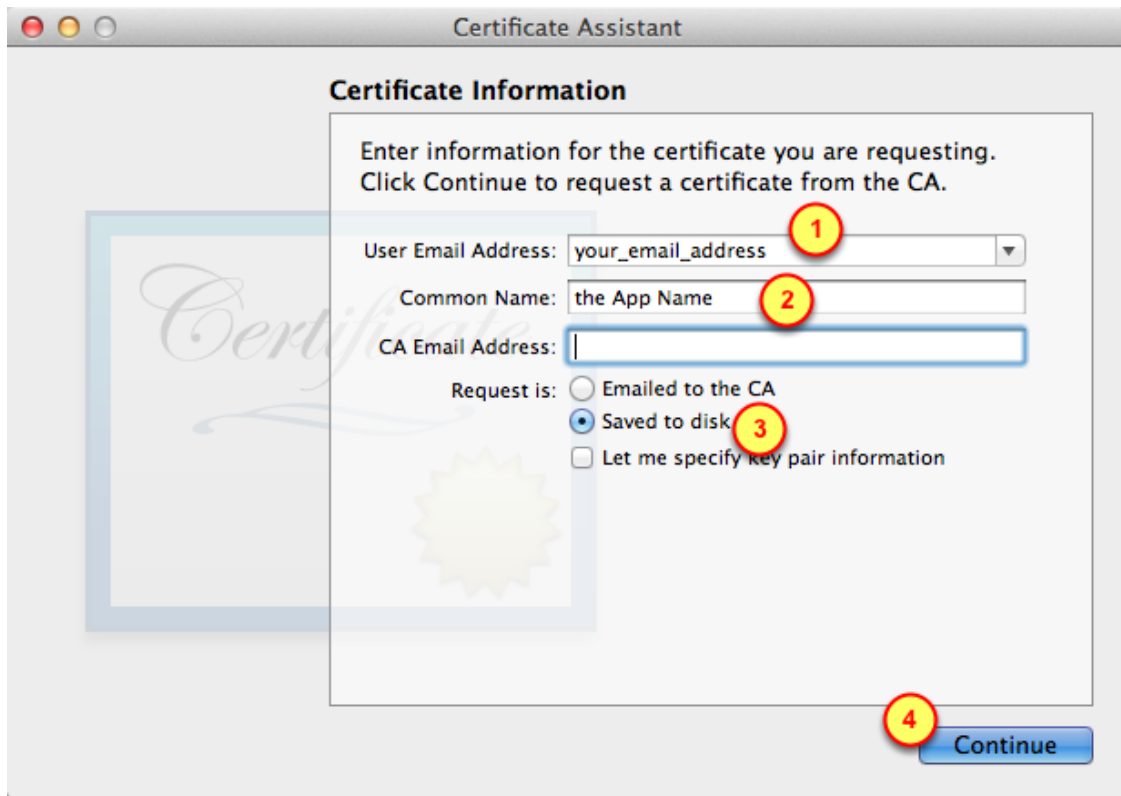
## 4. Request a Certificate from a Certificate Authority



We have to make a special type of request to Apple, it is called a Certificate Request.

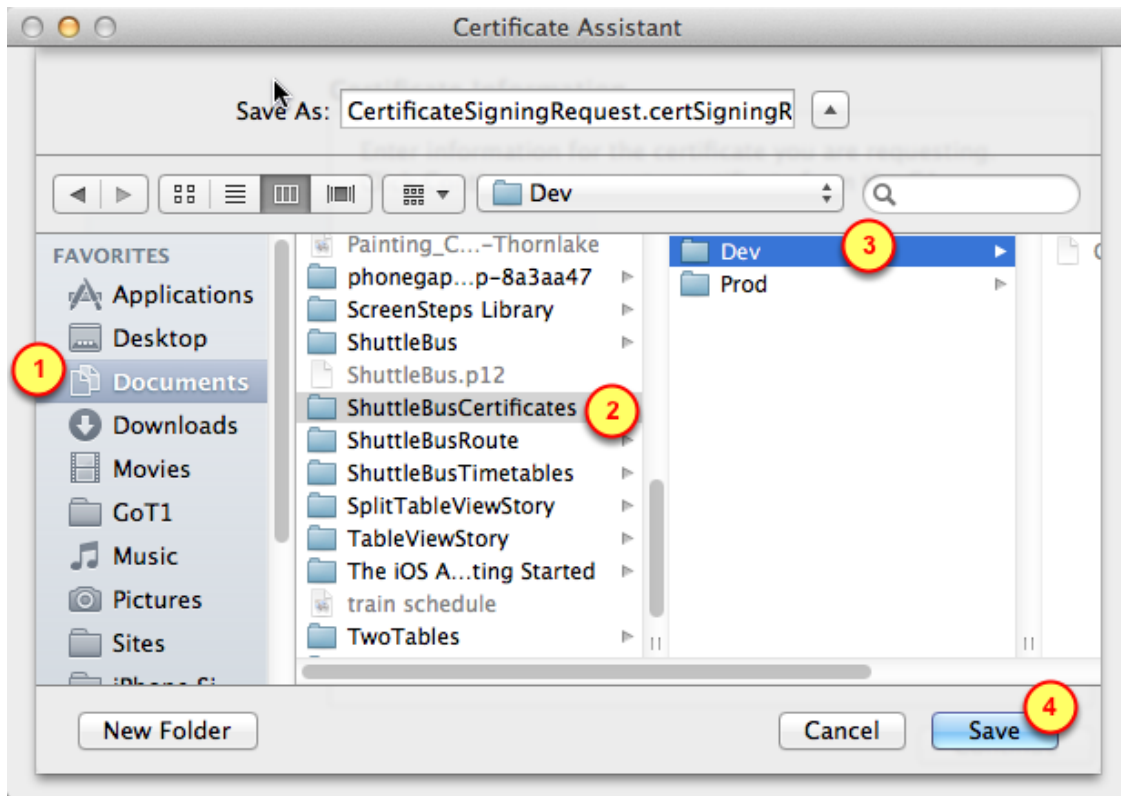
1. Click on the **Keychain Access** menu
2. Select the **Certificate Assistant** item in the menu
3. Choose to **Request a Certificate from a Certificate Authority**

## 5. Details for the Certificate



1. Use your real **email** address
2. For the **Common Name**, put the name of your App (spaces in the name are okay)
3. Specify to **save** the request to disk
4. Click on **Continue**

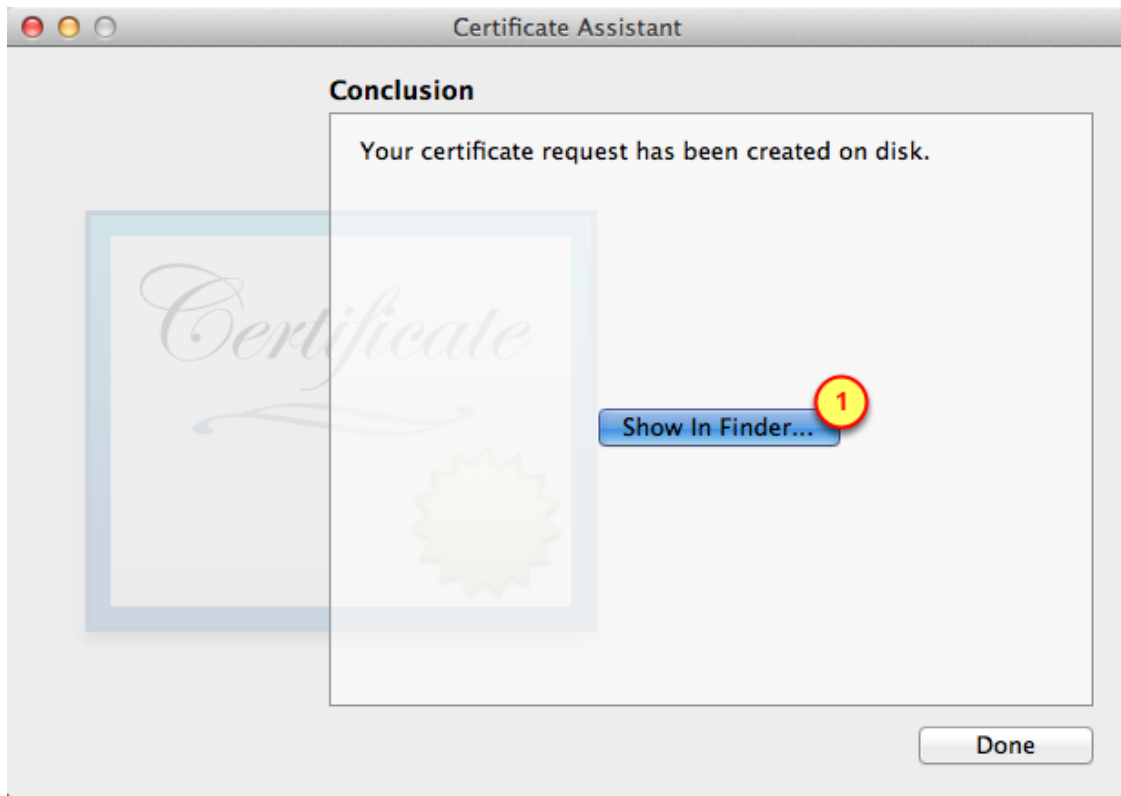
## 6. Save the Certificate Request



Save the Certificate Request in a known place:

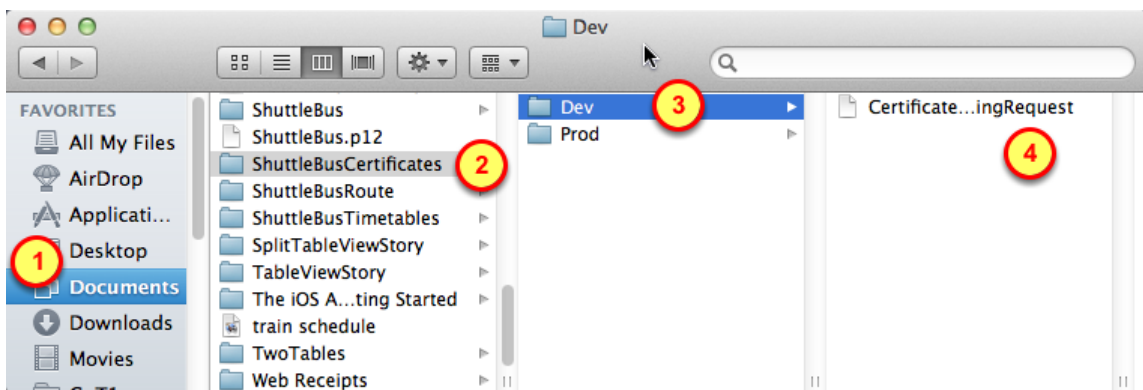
1. **Documents** folder
2. **App's** Certificates folder
3. Folder meant to hold the Certificate items for **Development**
4. Click on the **Save** button

## 7. Show the Certificate Request



1. Click on **Show in Finder** button to see where the Certificate Request was saved

## 8. Verify the Certificate Request was saved



Check to see the Certificate Request was saved in a known place:

1. **Documents** folder
2. **App's Certificates** folder

3. Folder meant to hold the Certificate items for **Development**

4. The **Certificate Request** file

# On Apple's Developer Provisioning Portal

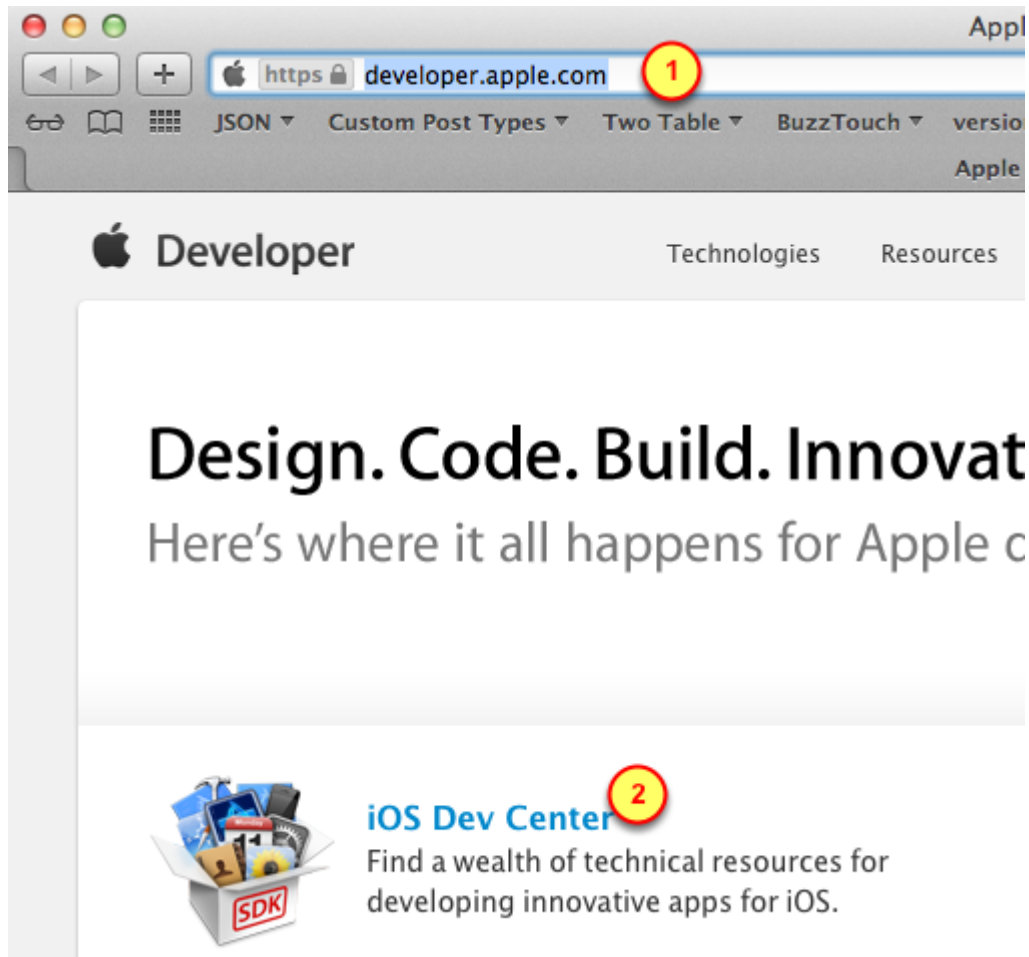
## Log into Apple's Developer Provisioning Portal

---

Log into Apple's Developer site to get to the Provisioning Portal.

In a later chapter, we will then create an App ID and a Provisioning Profile for the App.

### 1. Go to Apple's Developer site and to the iOS Dev Center



1. Go to this site: <http://Developer.Apple.com>

2. Click on the **iOS Dev Center**



## 2. Log in to the iOS Dev Center



1. Click on the **Log in** button

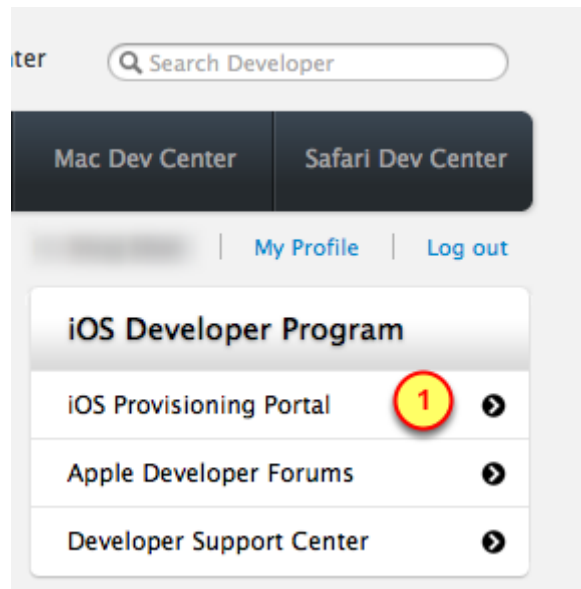
## 3. Specify your Apple ID credentials



1. Use the Apple ID that is registered with the Apple's Developer program
2. Give the corresponding password

3. Click on the **Sign In** button

#### 4. Enter the iOS Provisioning Portal



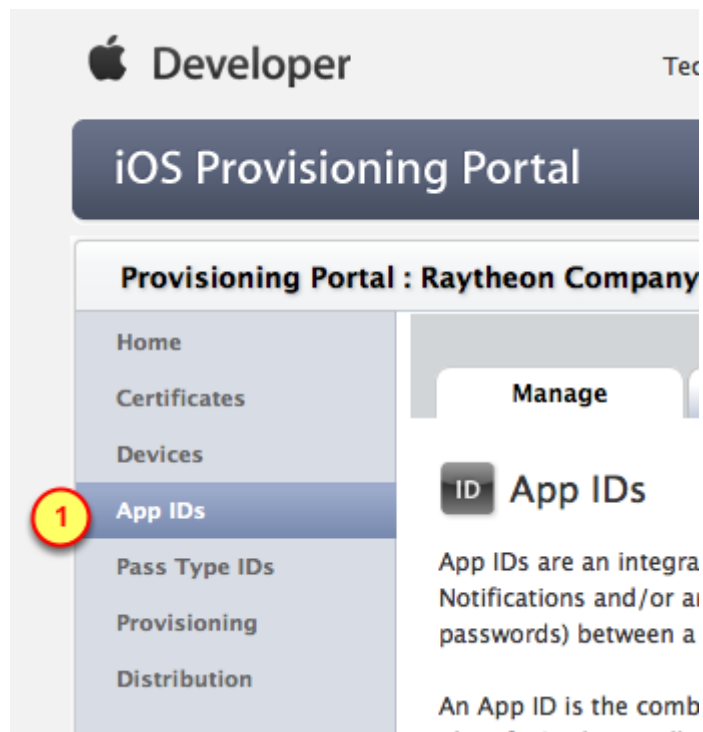
1. Click on the link for the **iOS Provisioning Portal**

## Create an App ID with Push Notifications enabled

---

We have to Push-enable the App by creating an App ID for it.

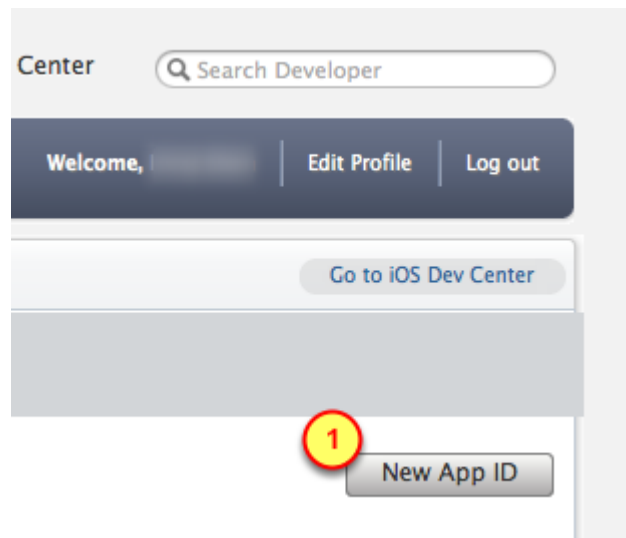
### 1. Create an App ID with Push Notifications enabled



In the iOS Provisioning Portal:

1. Click on the **App IDs** menu item

## 2. Create a new App ID



1. Click on the **New App ID** button

## 3. Details of the App ID

**Description**

Enter a common name or description of your App ID using throughout the Provisioning Portal to identify this App ID.

You cannot

**Bundle Seed ID (App ID Prefix)**

Use your Team ID or select an existing Bundle Seed ID for

Use Team ID   If you are creating a suite of applications that application's App IDs.

**Bundle Identifier (App ID Suffix)**

Enter a unique identifier for your App ID. The recommend Identifier portion of the App ID.

Example: co

1. Use the name of the App as the **name** for the App ID
2. Use a number, not the Team ID, for the **Bundle Seed**

3. For the **Bundle Identifier**, use this format: com.YourCompanyName.YourAppName (with no spaces)

#### 4. Possible Configurations for the App ID

915.com.elsegundo... Shuttle Bus

Passes:	Configurable	Configurable
Data Protection:	Configurable	Configurable
iCloud:	Configurable	Configurable
In-App Purchase:	Enabled	Enabled
Game Center:	Enabled	Enabled
Push Notification:	Configurable	Configurable

[Configure](#)

1

1. Click on the **Configure** link to setup the App ID

#### 5. Push Notification configuration for the App ID

ID Shuttle Bus .com.elsegundo.shuttlebus

1

**Enable for Apple Push Notification service**

Push SSL Certificate	Status	Expiration Date	Action
Development Push SSL Certificate	Configurable		<input type="button" value="Configure"/>
Production Push SSL Certificate	Configurable		<input type="button" value="Configure"/>

2

1. **Enable** the Push Notification service

2. Click on the **Configure** button

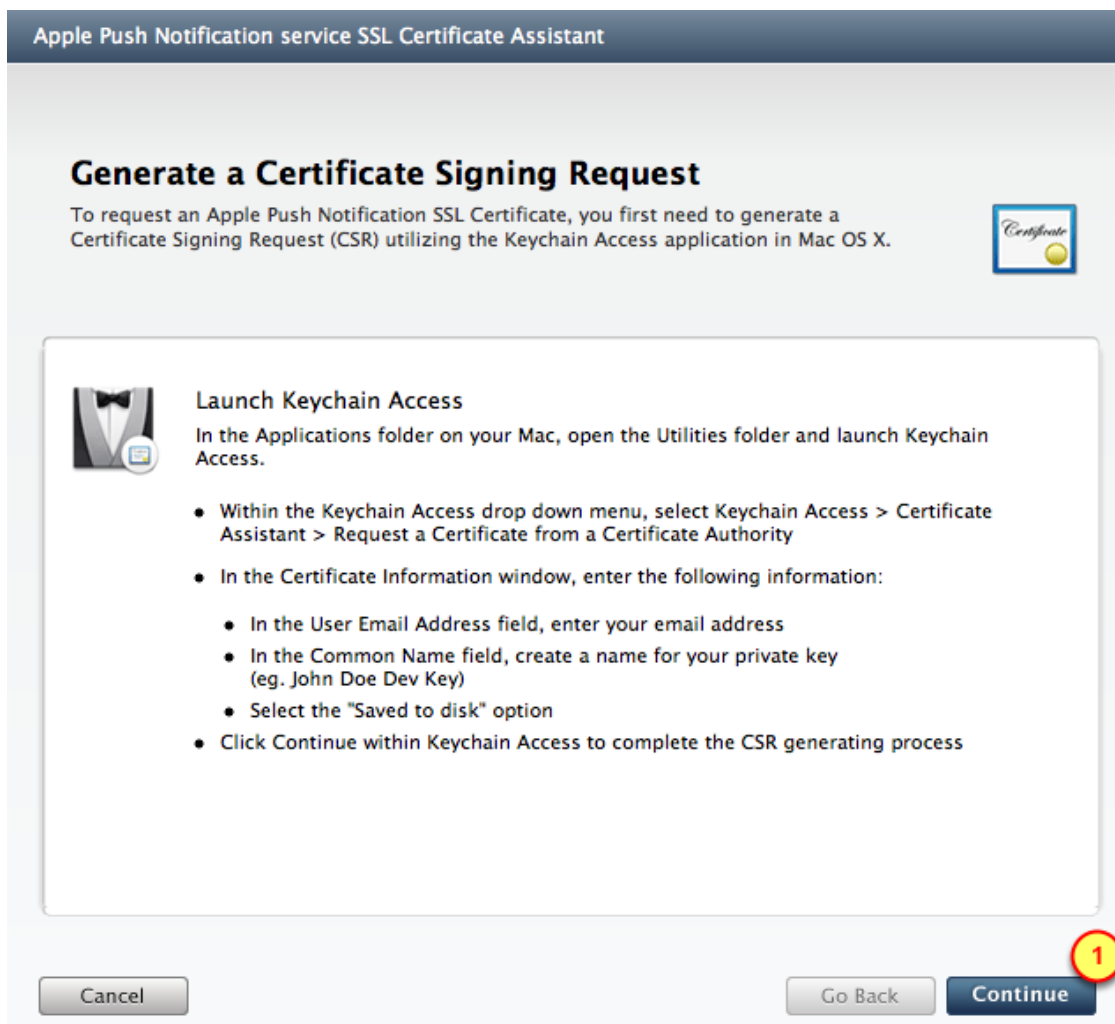
## Create APS Certificate

---

We will use the Certificate Request file (created previously) to generate a Push Notification certificate.

That Apple Push Notification System (APS) certificate will be uploaded to the BuzzTouch server (in later steps).

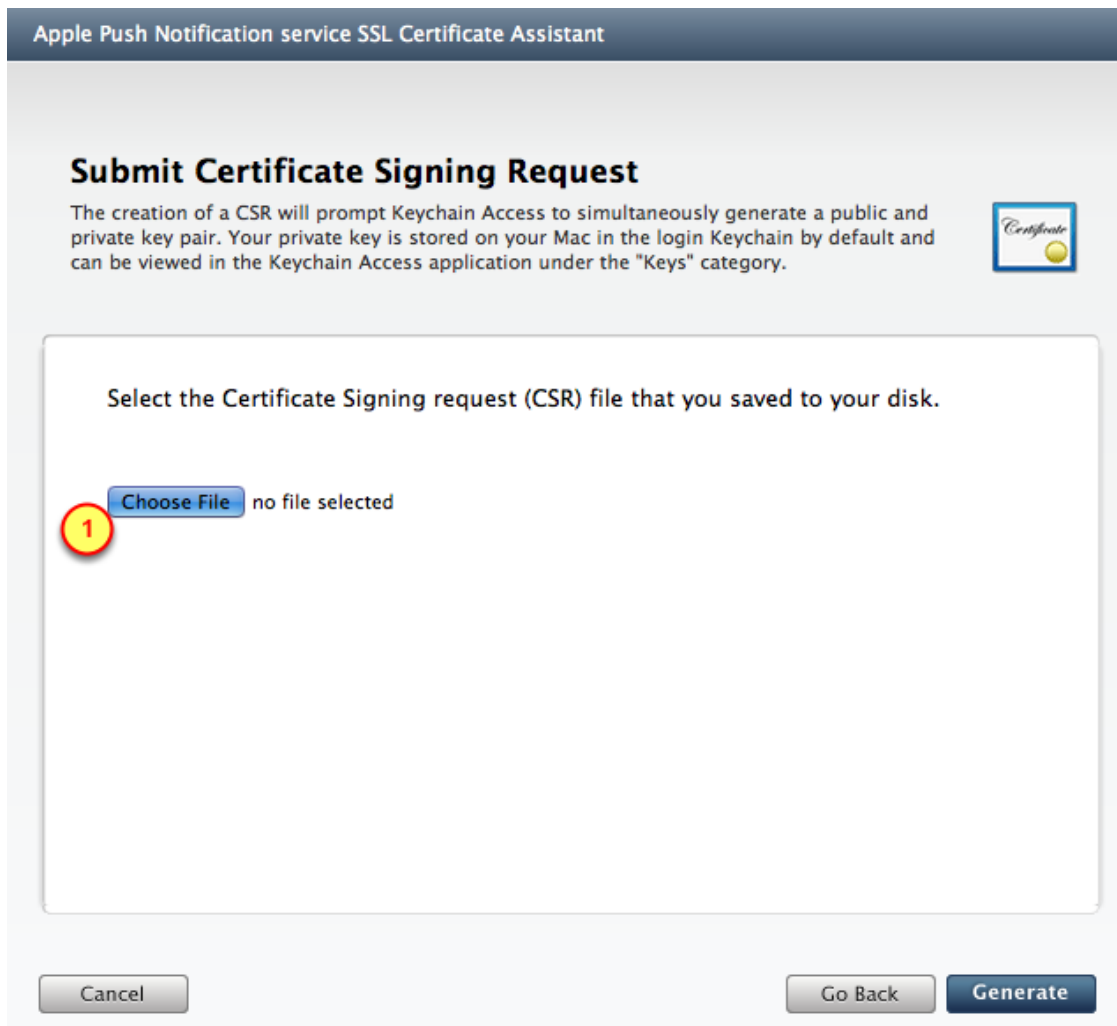
### 1. Create APS Certificate for Push Notification



You've already created the Certificate Request file in previous steps.

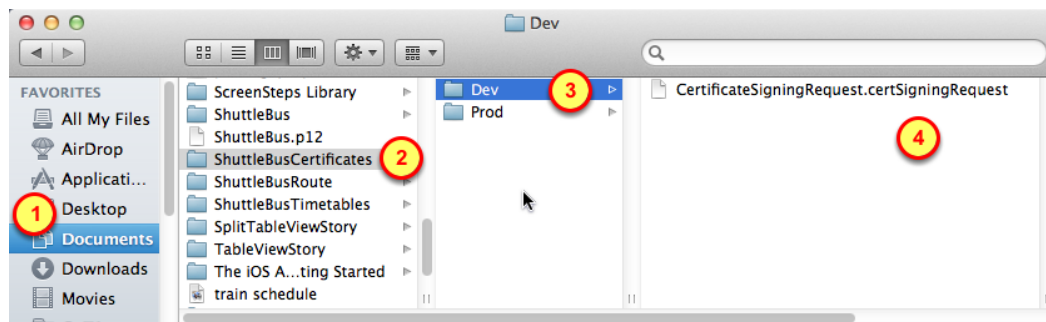
1. Click on the **Continue** button

## 2. Choose file



1. Click the **Choose File** button

## 3. Select the Certificate Request file

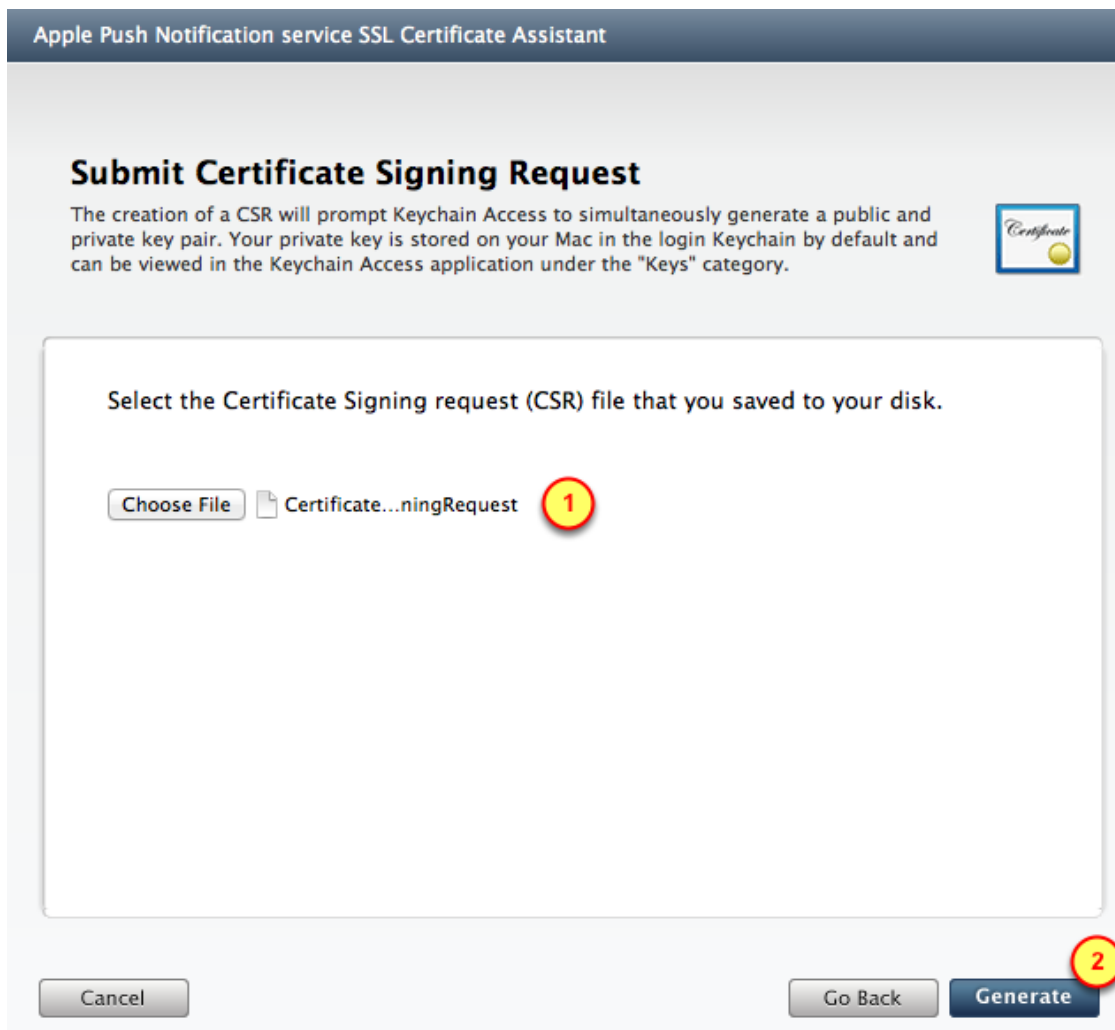


1. **Documents** folder
2. The **App's** certificates folder

3. Folder for **Development** items

4. The **Certificate Request** file

#### 4. Generate the Push Notification certificate



Use the Certificate Request file to generate the Push Notification certificate

1. Verify the **Certificate Request** file was selected

1. Click on the **Generate** button



## 5. Generation in progress ...

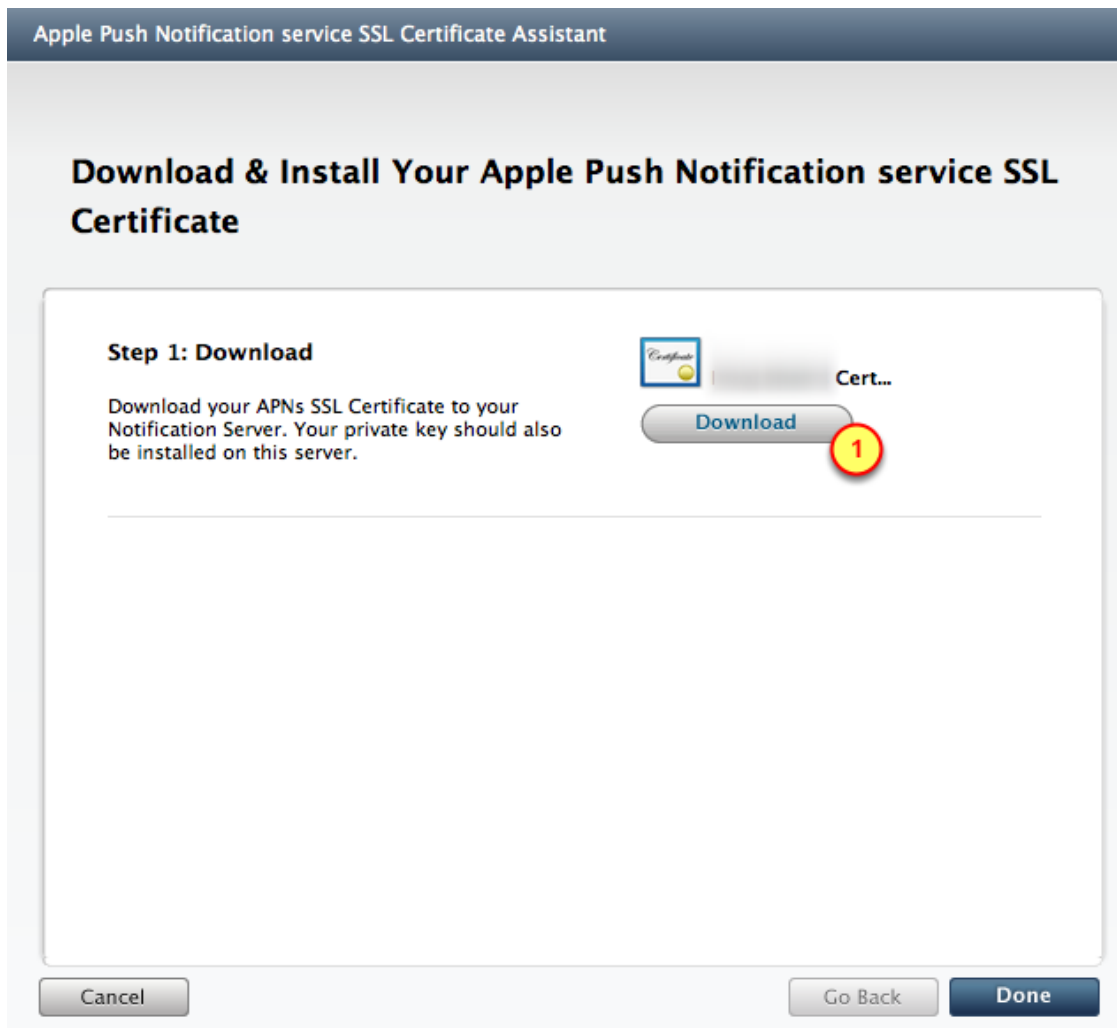


## 6. Push Notification certificate created



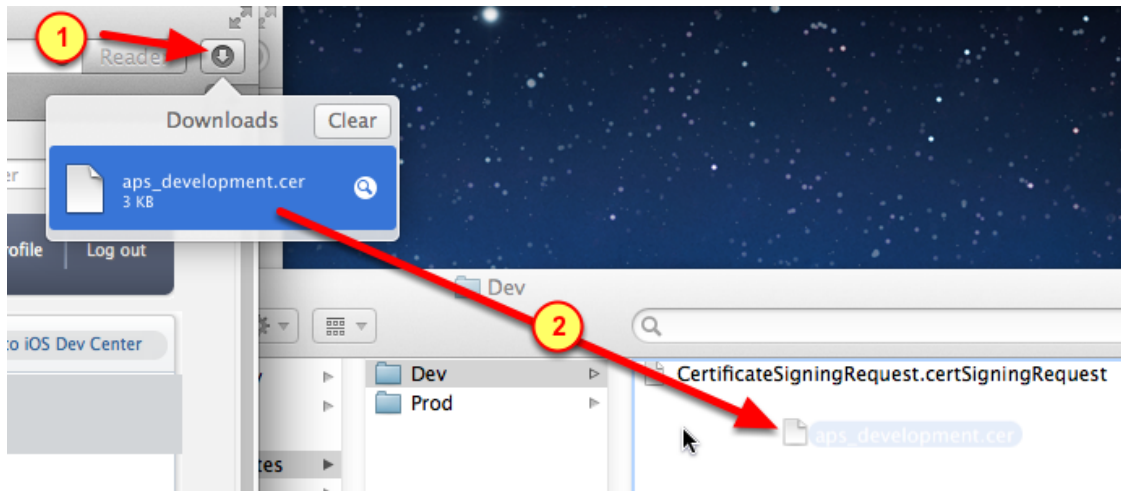
1. Verify the Push Notification certificate was **generated**
2. Click the **Continue** button

## 7. Download the Push Notification certificate



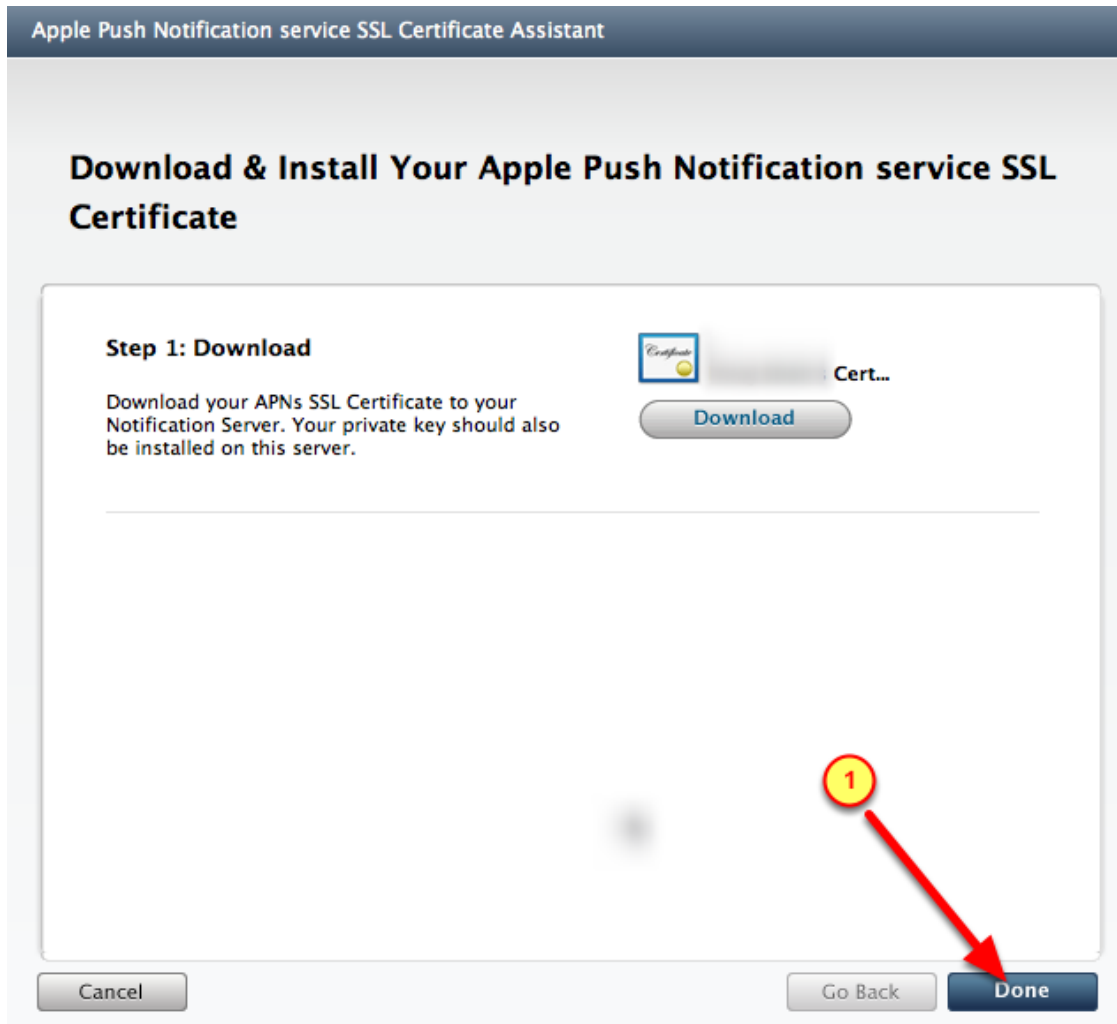
1. Click on the **Download** button to get the Push Notification certificate

## 8. Download and save the APS certificate



1. Click on the Downloads icon
2. Drag-n-drop the APS certificate into the Dev folder

## 9. Done with the Push Notifications process



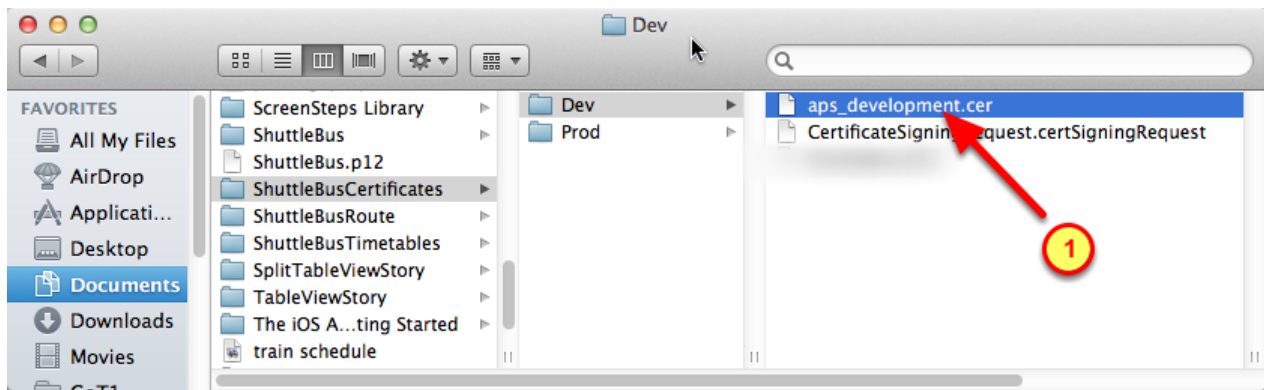
## Create .P12 certificate

---

Using the APS certificate from the Provisioning Portal, we will create a .P12 certificate.

In later steps, the .P12 file will then be used to generate a .PEM certificate file to be used by the BuzzTouch server

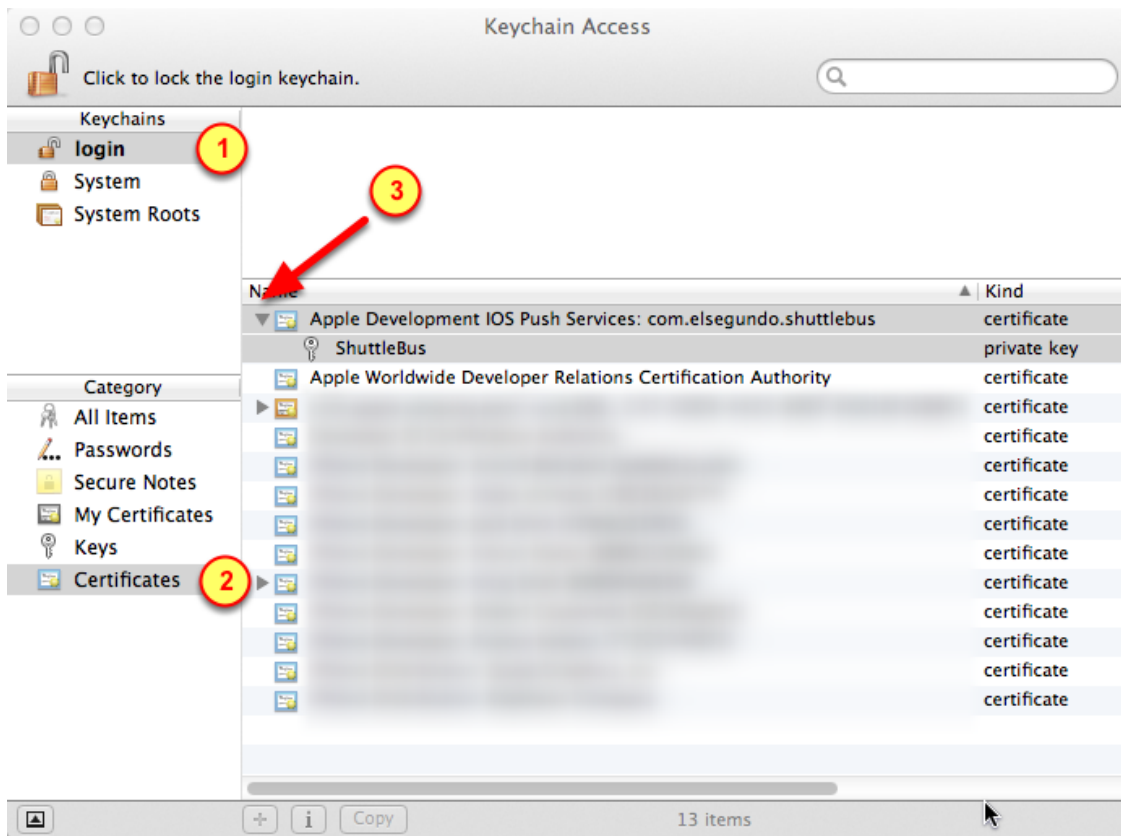
### 1. Install APS certificate into KeyChain



Install the APS Push Notification certificate into the Keychain

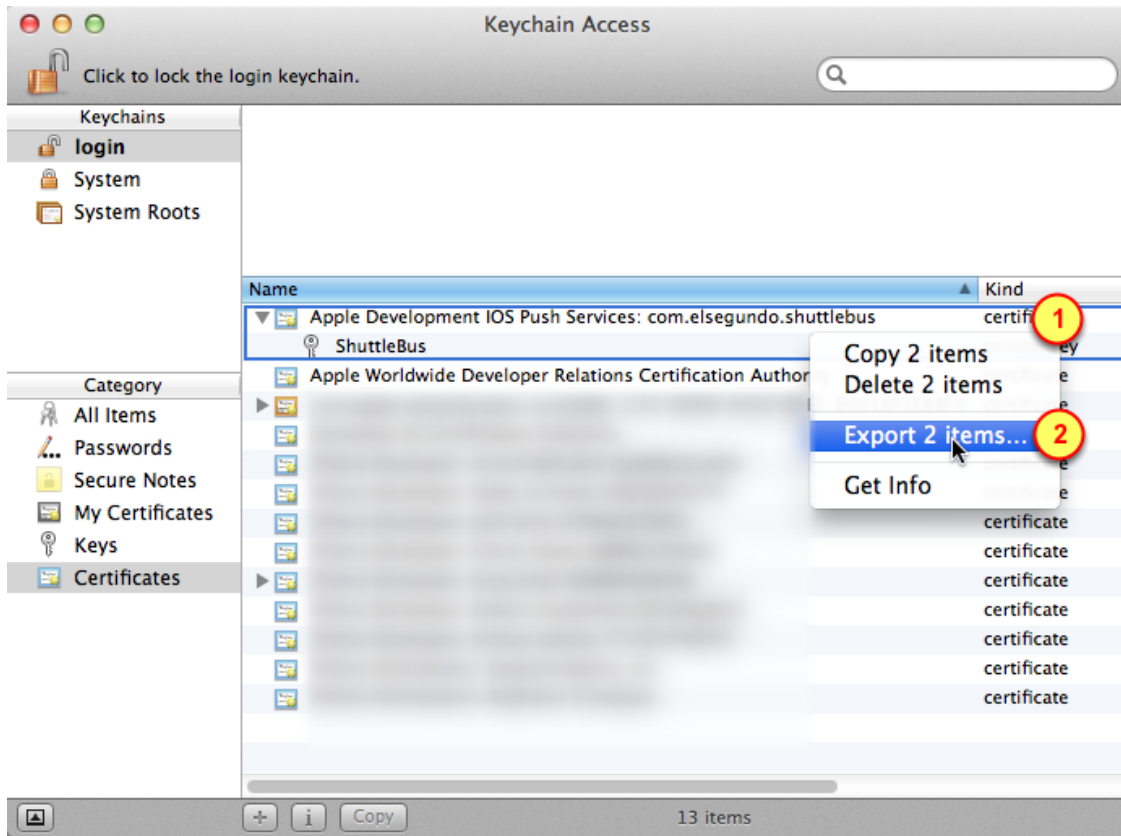
1. Double-click on the APS file

## 2. Select the Push Notification certificates for exporting



1. Select the **login** keychain
2. Select the Certificates category
3. Click on the triangle to expand-open the Push Notification certificate

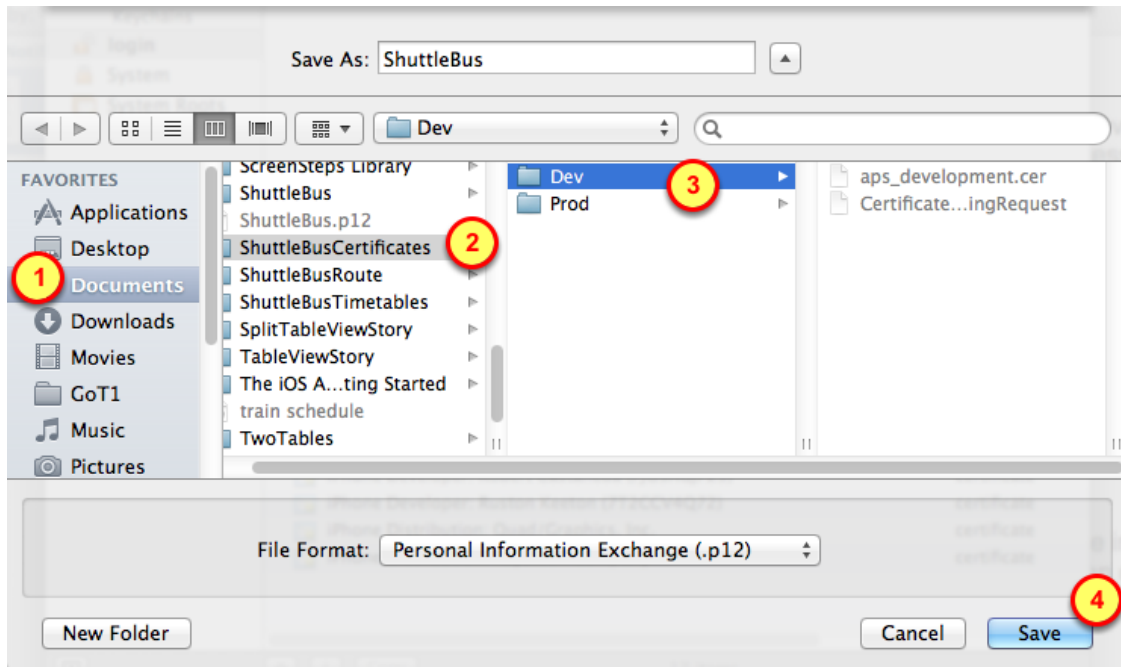
### 3. Select certificates for exporting



1. With the two certificates selected, **Right-Click** on those items
2. Select **Export 2 items** (in the form of a .P12 certificate)



## 4. Navigate to the desired folder



1. Select the **Documents** folder
2. Select the **App's** folder for certificates
3. Select the folder that holds items for **Development** (or Production)
4. Click on the **Save** button

## 5. No passwords



Do not specify a password for this prompt.

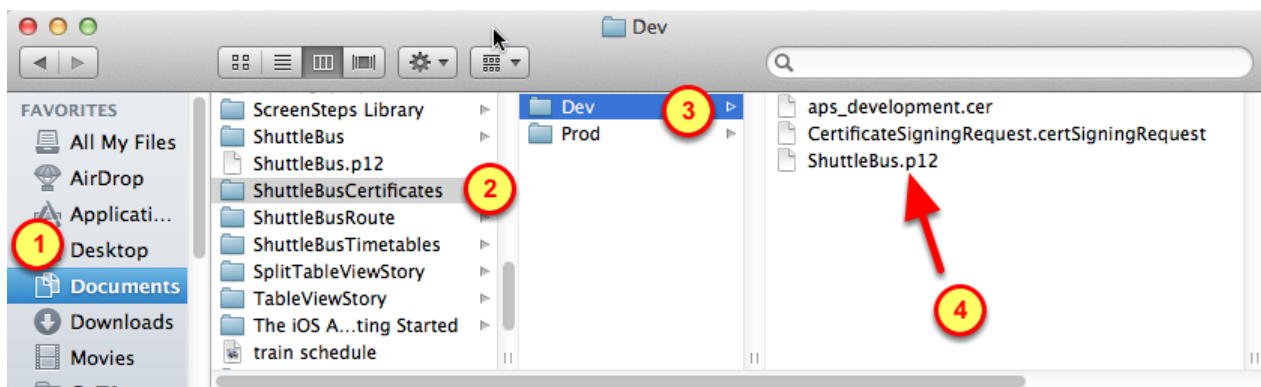
1. Click the **OK** button to proceed to the next step

## 6. Mac account's password



1. Specify the password for the Mac's account
2. Click on the **Allow** button to save the P12 certificate onto the Mac's drive

## 7. Verify you have a .P12 Certificate



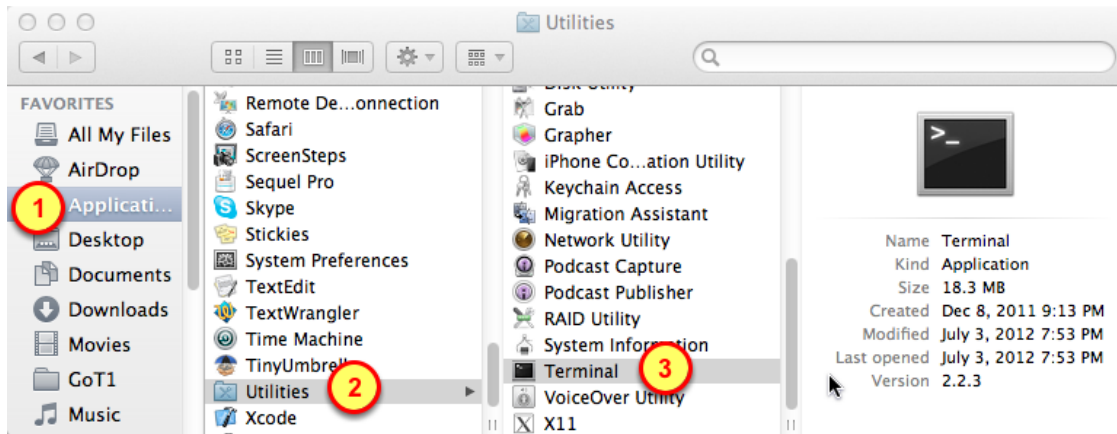
1. Select the **Documents** folder
2. Select the **App's** folder for certificates
3. Select the folder that holds items for **Development** (or Production)
4. Verify the App's .P12 certificate is in that folder

## Convert .P12 file into an .PEM file

The BuzzTouch server uses a .PEM file to communicate with Apple's Push Notification Service.

Using the .P12 certificate created in previous steps, we will use a secure certification process to generate a .PEM file for the BuzzTouch server.

### 1. Launch the Terminal application



1. Open the **Applications** folder
2. Go into the **Utilities** folder
3. Double-click the **Terminal** application to launch it

### 2. Navigate to the folder holding the certificates

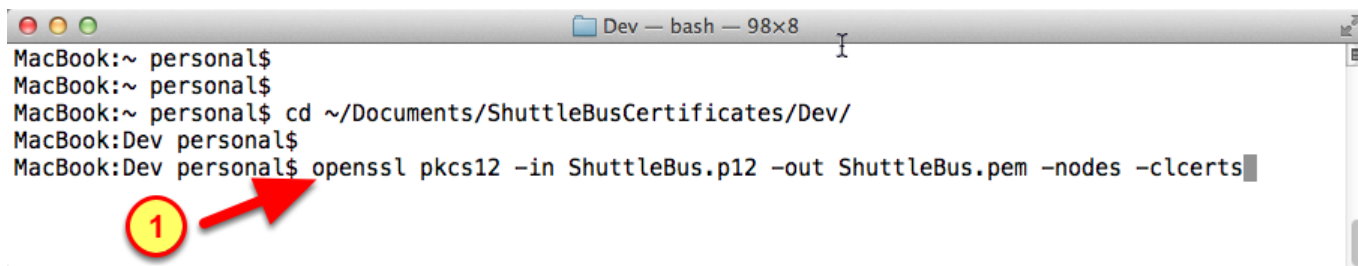


1. Change Directory to the folder holding the certificate items for Development (or Production)

```
cd ~/Documents/ShuttleBusCertificates/Dev/
```

( At the Finder-level, try not to use spaces in any of the folder names )

### 3. Create .PEM file



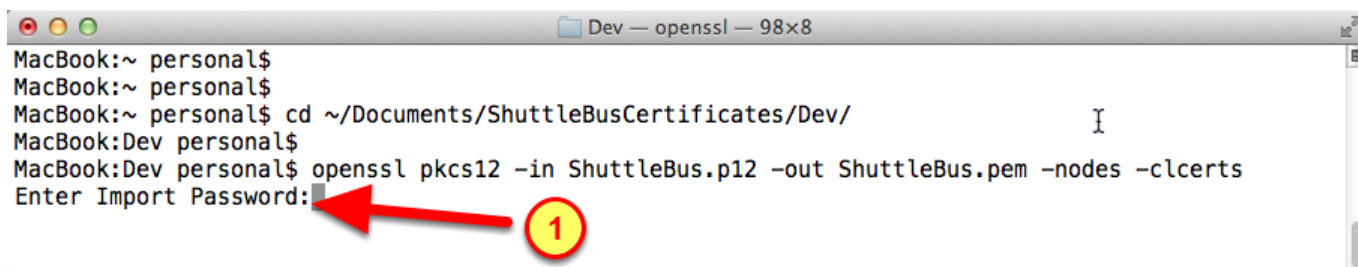
```
MacBook:~ personal$  
MacBook:~ personal$  
MacBook:~ personal$ cd ~/Documents/ShuttleBusCertificates/Dev/  
MacBook:Dev personal$  
MacBook:Dev personal$ openssl pkcs12 -in ShuttleBus.p12 -out ShuttleBus.pem -nodes -clcerts
```

A red circle with the number '1' and a red arrow pointing to the `openssl` command in the terminal.

1. Using the OpenSSL utility, use the .P12 file to generate a .PEM certificate

```
openssl pkcs12 -in APPNAME.p12 -out APPNAME.pem -nodes -clcerts
```

### 4. Ignore password prompt

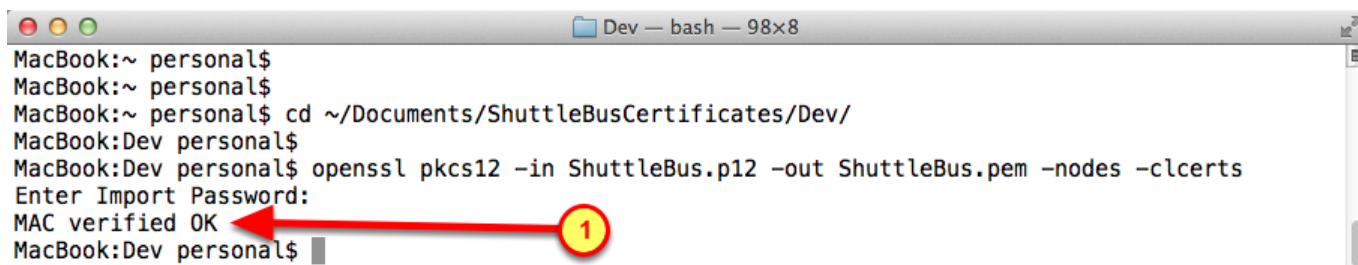


```
MacBook:~ personal$  
MacBook:~ personal$  
MacBook:~ personal$ cd ~/Documents/ShuttleBusCertificates/Dev/  
MacBook:Dev personal$  
MacBook:Dev personal$ openssl pkcs12 -in ShuttleBus.p12 -out ShuttleBus.pem -nodes -clcerts  
Enter Import Password:
```

A red circle with the number '1' and a red arrow pointing to the `Enter Import Password:` prompt in the terminal.

1. When prompted for the password, ignore it by pressing the Return key on the keyboard

### 5. Verify there were no errors

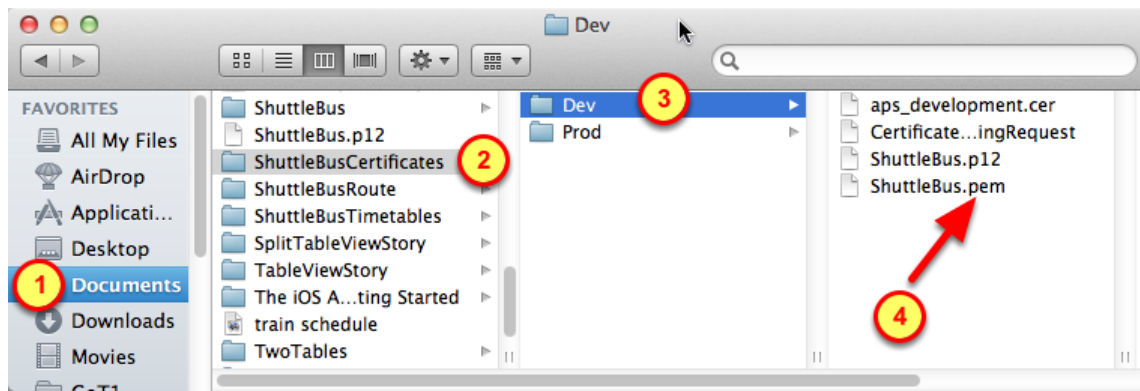


```
MacBook:~ personal$  
MacBook:~ personal$  
MacBook:~ personal$ cd ~/Documents/ShuttleBusCertificates/Dev/  
MacBook:Dev personal$  
MacBook:Dev personal$ openssl pkcs12 -in ShuttleBus.p12 -out ShuttleBus.pem -nodes -clcerts  
Enter Import Password:  
MAC verified OK  
MacBook:Dev personal$
```

A red circle with the number '1' and a red arrow pointing to the `MAC verified OK` message in the terminal.

1. Verify the .PEM was generated with no errors

## 6. Validate .PEM file was created

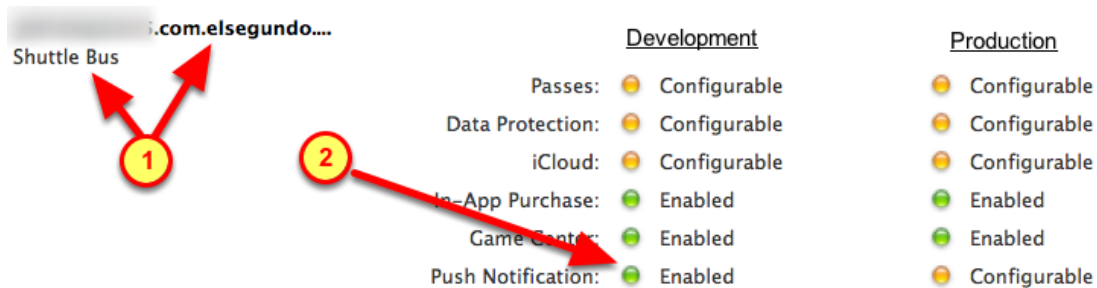


1. Select the **Documents** folder
2. Select the **App's** folder for certificates
3. Select the folder that holds items for **Development** (or Production)
4. Verify the App's **.PEM** certificate is in that folder

## Create and install Provisioning Profile

We need a Provisioning Profile that is enabled for Push Notifications. Using the new App ID, we will create a valid Provisioning Profile that can be used to install the Development version of the App onto the enabled devices. That same Provisioning Profile is also associated with selected Developers to allow them to compile that App.

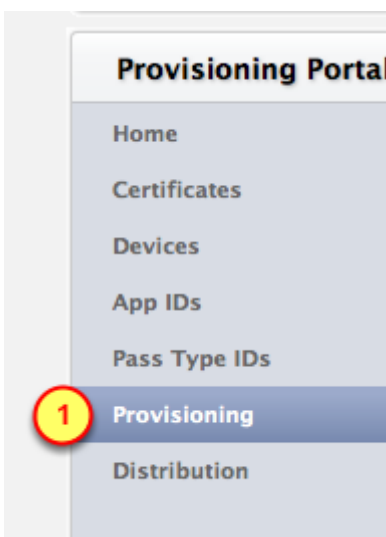
### 1. Check App ID for Push Notification



Verify the new App ID has been enabled for Push Notification for Development (or Production)

1. In the App ID section, find the App ID with the name of the App
2. Look for a green light for Push Notification in the Development column (or Production column)

### 2. Provisioning section of the iOS Provisioning Portal



1. Click on the **Provisioning** menu item

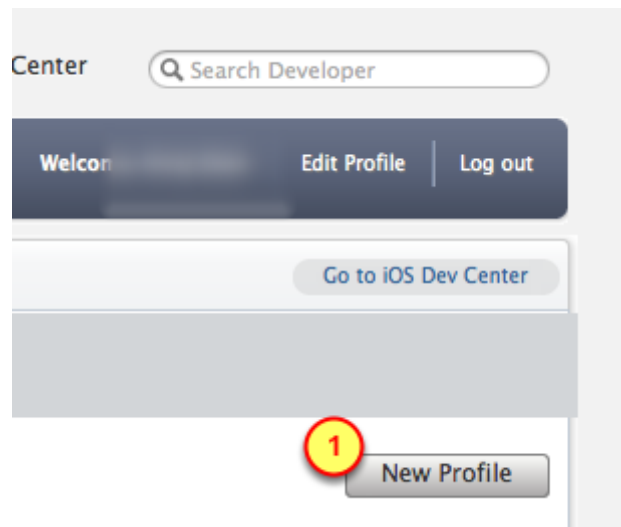
### 3. Development folder tab (or Distribution)



1. Click on the **Development** folder tab for testing push notifications

( or click on the Distribution folder tab for an App that is ready for publishing to the Apple App Store)

### 4. Create a New Provisioning Profile



1. Click on the **New Profile** button to create a new provisioning profile

## 5. Details for the new provisioning profile

Development Distribution History Ho

### Create iOS Development Provisioning Profile

Generate provisioning profiles here. All fields are required unless otherwise

**Profile Name** 1 Shuttle Bus Dev

**Certificates** Select All

2

**App ID** 3 Shuttle Bus

**Devices** Select All

4

1. Specify the **name** of the App and the **Dev** or **Prod** term.

Example: Shuttle Bus Dev or Shuttle Bus Prod (spaces are allowed)

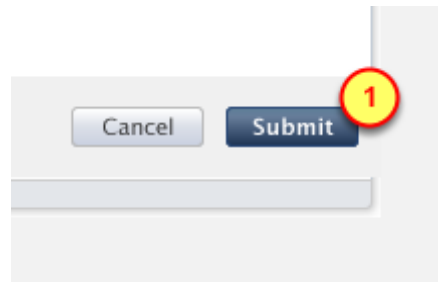
2. Select which **People** to be included on that Provisioning Profile. These people are usually your Testers. Remember to include yourself!

3. Select the **App ID** that we had previously created

4. Select the **devices** that will be used by the Testers from item #2

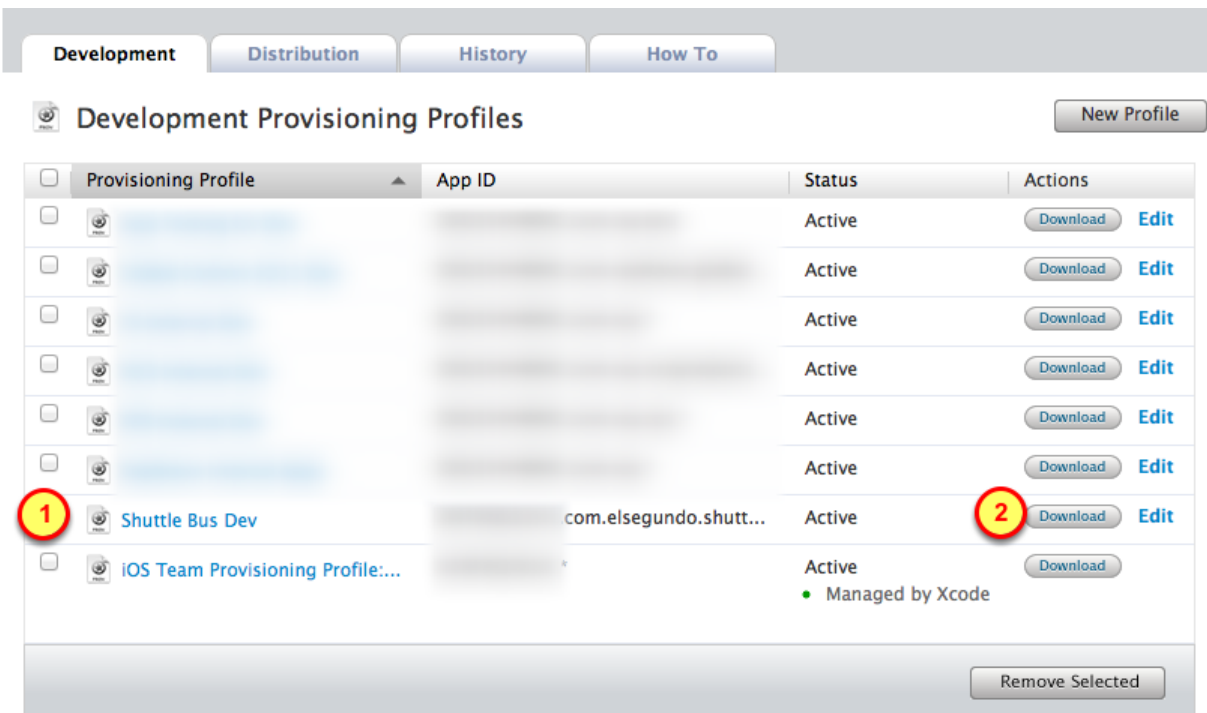


## 6. Create the provisioning profile



1. Click on the **Submit** button

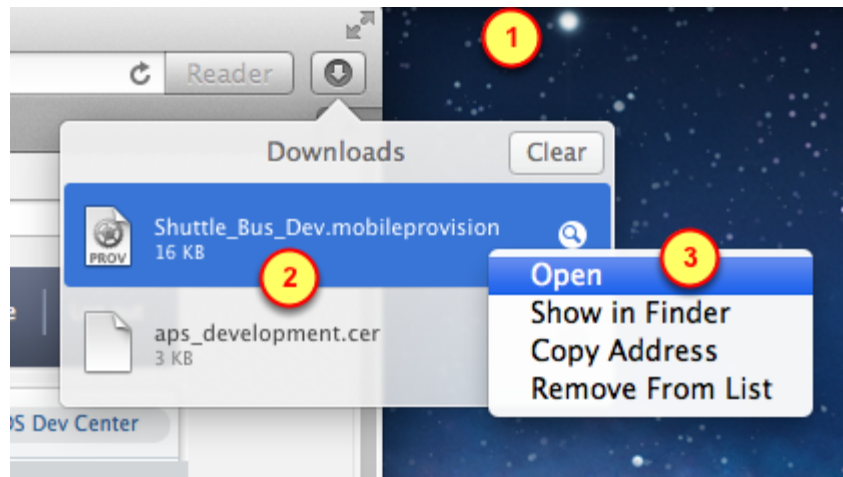
## 7. Download the new provisioning profile



After submitting the request for a new provisioning profile, the browser should be refreshed.

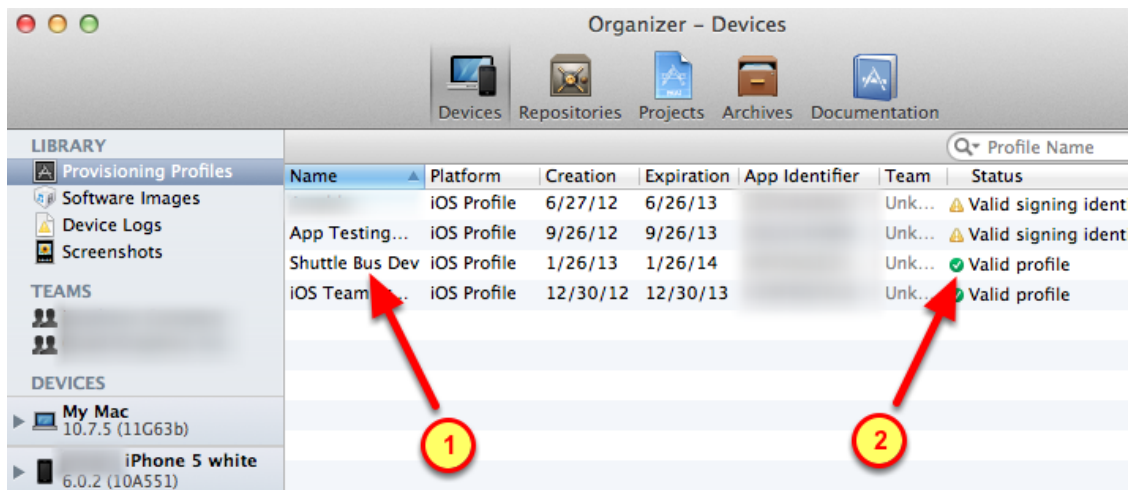
1. Find the row for the newly created provisioning profile
2. Click on the **Download** button for the new provisioning profile

## 8. Open the new Provisioning Profile



- 1 Open the Downloads view of the Browser
2. Right-click on the row for the Provisioning Profile
- 3 Select the **Open** item in the list

## 9. Verify the provisioning profile



Opening a Provisioning Profile will cause it to be imported into the Organizer (on the Mac)

1. Looking at the list of Provisioning Profiles, verify the App's name is in the list
2. Verify the Status of that Provisioning Profile is **"Valid profile"** and it has a green checkmark

# Settings for BuzzTouch Control Panel

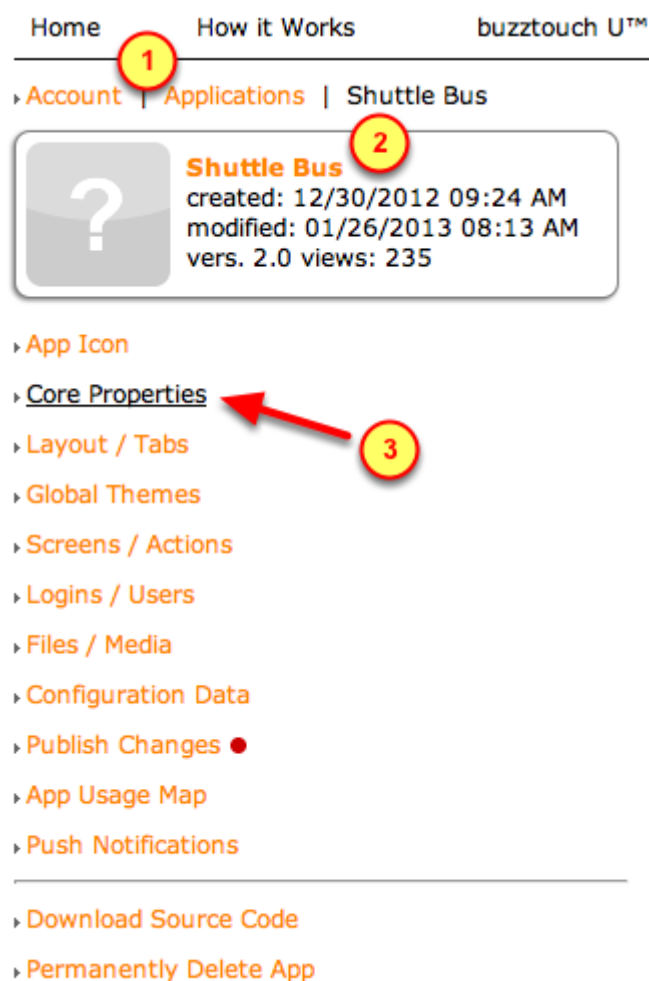
## Core settings for Push Notifications

---

There are two core settings for Push Notifications, both of which need to be changed from the default settings:

- Prompting the User to allow push notifications to be sent to the app installed on the device
- The URL to register the device with push notifications Provider (the BuzzTouch server)

### 1. Go into the Core Settings for the App



Go into the Core Settings for the App:

1. For your BuzzTouch account, click on **Applications**

2. Click on the **App** that needs Push Notifications

3. Click on "**Core Properties**"

## 2. Open the settings for Push Notifications

Home      How it Works      buzztouch U™      Self Hosting

▸ Applications | Shuttle Bus | App Icon | Core | Layout | Themes

### Manage Core Properties for Shuttle Bus

**Core Properties** allow you to control the app's basic behavior when it launches. The name you enter here will appear under the app's icon on the device. You can use either Xcode or Android project.

- Application Name
- Application Id, API Key, API Secret
- Project Name
- Configuration Data URL
- Report to Cloud URL
- Start Tracking Location (turn on GPS)
- **Push Notification Settings**
- Allow / Prevent Rotation (landscape, portrait)
- Application Location (address, city, state, lat/long)

To change the core settings for Push Notifications:

1. Click on "**Push Notification Settings**"

### 3. Default settings for Push Notifications

Home      How it Works      buzztouch U™      Self Hosting

› Applications | Shuttle Bus | App Icon | Core | Layout | Themes |

#### Manage Core Properties for Shuttle Bus

**Core Properties** allow you to control the app's basic behavior when it launch name you enter here will appear under the app's icon on the device. You can Xcode or Android project.

› Application Name

› Application Id, API Key, API Secret

› Project Name

› Configuration Data URL

› Report to Cloud URL

› Start Tracking Location (turn on GPS)

› Push Notification Settings

**Prompt for Push Notifications** 1

No, do not prompt for Push Notifications

**Register Device URL** 2 › Re-set to the the default control panel URL

These settings are used to prompt the app user to allow or disallow Push No device token to a remote server where it is saved and used when push noti

save

We have to change the default settings, which are:

1. **Prompt for Push Notifications** = No (default)
2. **Register Device URL** = blank (default)

## 4. Change settings for Push Notifications

Home      How it Works      buzztouch U™      Self Hosting

› Applications | Shuttle Bus | App Icon | Core | Layout | Themes |

### Manage Core Properties for Shuttle Bus

**Core Properties** allow you to control the app's basic behavior when it launch name you enter here will appear under the app's icon on the device. You can Xcode or Android project.

- › Application Name
- › Application Id, API Key, API Secret
- › Project Name
- › Configuration Data URL
- › Report to Cloud URL
- › Start Tracking Location (turn on GPS)
- › Push Notification Settings
  - Prompt for Push Notifications**
  - Register Device URL** › Re-set to the the default control panel URL  
<https://www.buzztouch.com/api/app/?command=registerForPush&appGuid=JA59>

These settings are used to prompt the app user to allow or disallow Push No device token to a remote server where it is saved and used when push noti

**3**

● Saved!

Change the default settings to:

1. **Prompt for Push Notifications** = Yes
2. **Register Device URL** = click on the "Re-set" link
3. Click on the "**save**" button

## *Uploading .PEM Certificate into BuzzTouch*

---

The **.PEM certificate** is the connection between the App and the BuzzTouch server for Push Notifications.

Two different types of certificates are used:

- Development
- Production / Distribution

For this tutorial, the Development .PEM certificate is used.

The same procedure is used for the Production .PEM certificate.



## 1. Push Notification Menu

The screenshot shows the BuzzTouch application control panel. At the top, the BuzzTouch logo is displayed. Below the logo, there are navigation links: Home, How it Works, and BuzzTouch U™. A breadcrumb trail shows: Account | Applications | Shuttle Bus. A card for the 'Shuttle Bus' application is shown, with a yellow circle containing the number 1 next to a question mark icon. The card details include: created: 12/30/2012 09:24 AM, modified: 01/01/2013 09:19 PM, and vers. 2.0 views: 227. Below the card is a list of menu items: App Icon, Core Properties, Layout / Tabs, Global Themes, Screens / Actions, Logins / Users, Files / Media, Configuration Data, Publish Changes (with a red dot), App Usage Map, Push Notifications (with a red arrow and a yellow circle containing the number 2), Download Push Notifications (with a yellow highlight), and Permanently Delete App.

1. Ensure you are in the correct **Application's** Control Panel.

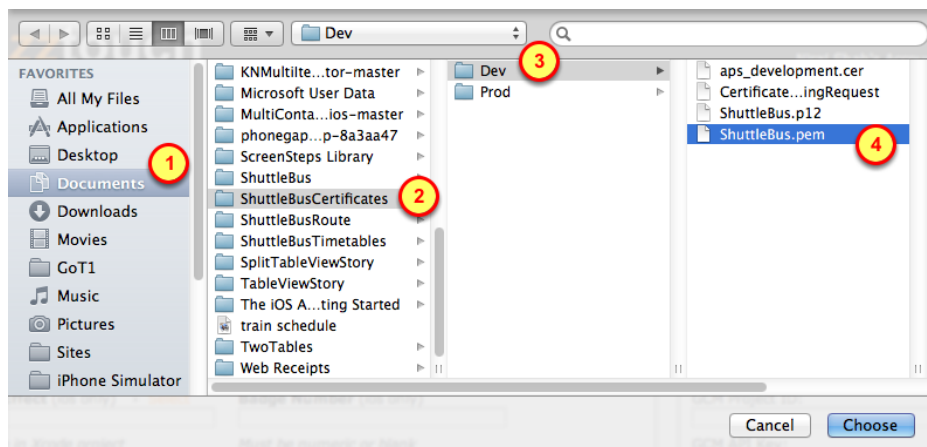
2. Click on the "**Push Notifications**" menu

## 2. .PEM Certificates section



1. Due to the **red-dots**, notice that neither type of certificate has not yet been uploaded.
  2. Select "**Development**" as the type of certificate to be uploaded
  3. Type in a random **passphrase**, you won't have to remember it's value
  4. Click on the **Plus-sign** to invoke the file browser (to point it to the .PEM certificate file)
- ( In the next step, you will specify the actual .PEM certificate file )

## 3. Find the .PEM Certificate file on the Mac



In previous steps, we had exported the Certificate-pair from the KeyChain as a .PEM certificate. Lets upload that .PEM certificate into the BuzzTouch Control Panel.

1. **Documents** folder
2. App-specific **certificates** folder
3. **Dev** certificates folder (or Prod folder)
4. App's **.PEM** certificate file

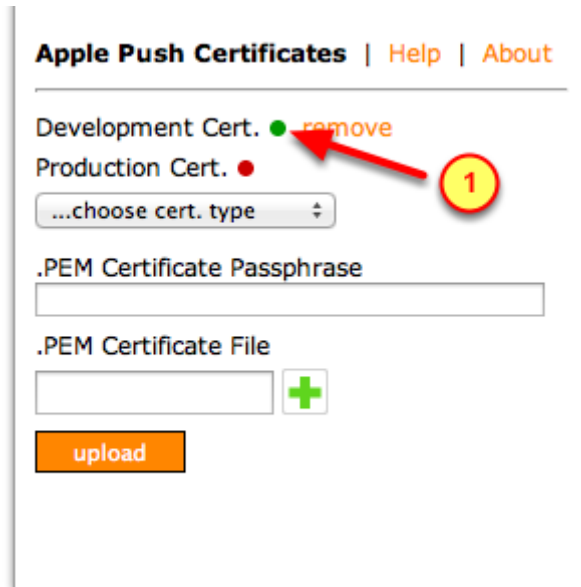
#### 4. Upload **.PEM** certificate into BuzzTouch server



Now that you've located the Dev (or Prod) **.PEM** certificate file, upload it into the BuzzTouch server.

1. Verify the field shows the name of the **.PEM** certificate file
2. Click on the **"upload"** button

## 5. Verify the Development Certificate is installed on BuzzTouch server



Now that you have uploaded the .PEM certificate onto the BuzzTouch server, let's make sure BuzzTouch liked it.

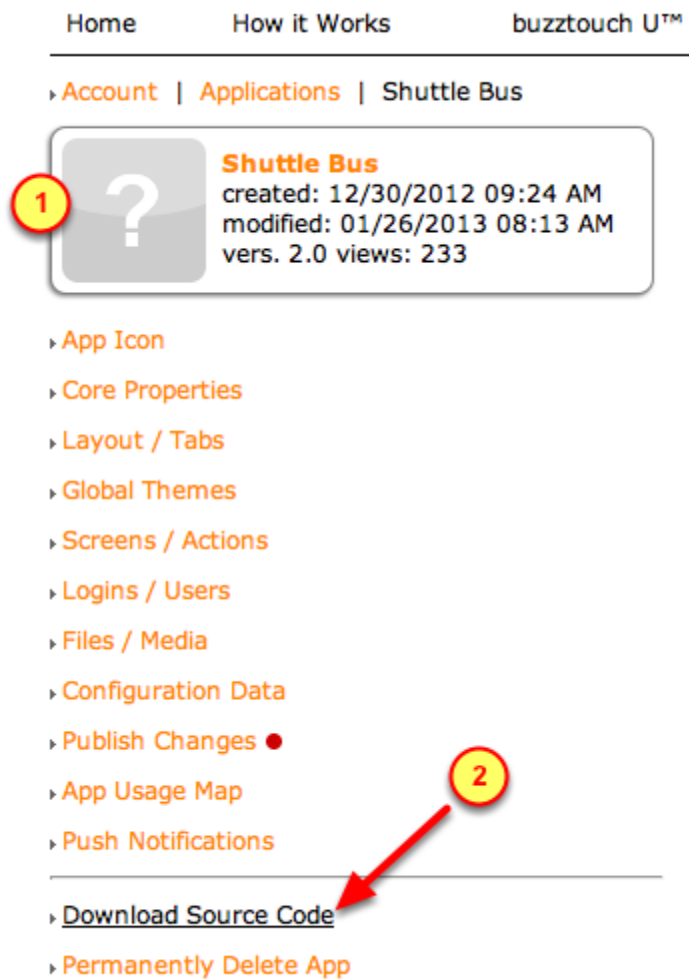
1. Verify the Development (or Production) certificate has a green-dot

## Download your App's source code from BuzzTouch server

---

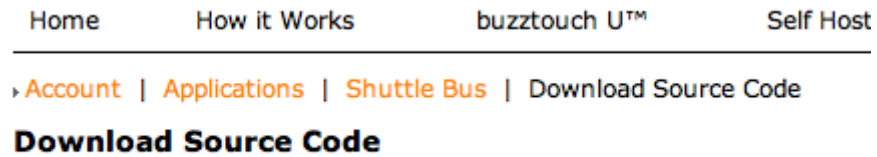
To ensure you have the latest version of software, download the Source Code from the BuzzTouch server.

### 1. App's Control Panel



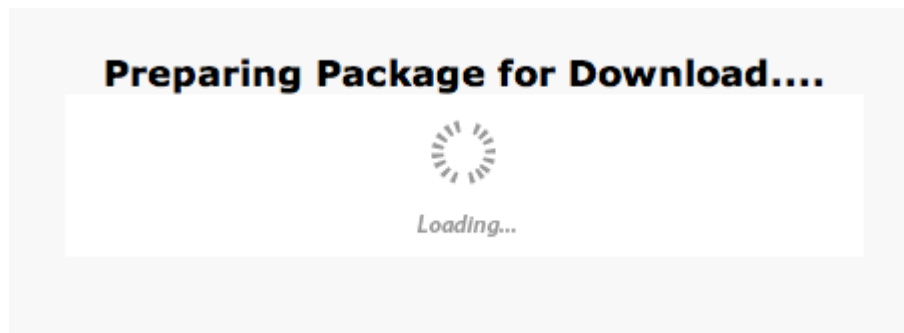
1. Ensure you are in the correct App's Control Panel on the BuzzTouch server
2. Click on the **Download Source Code** menu item

## 2. Prepare iOS source code package for download

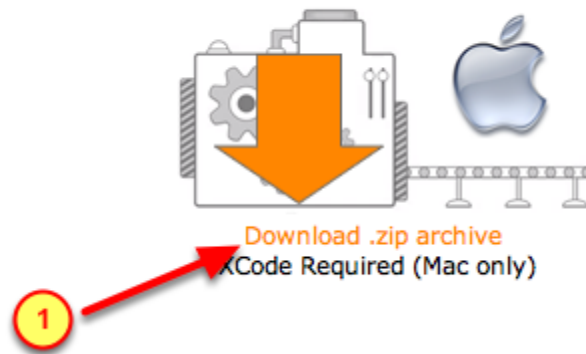
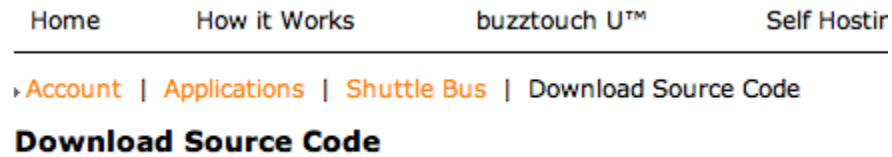


1. Click on the **Prepare package for download** link

## 3. Package being generated

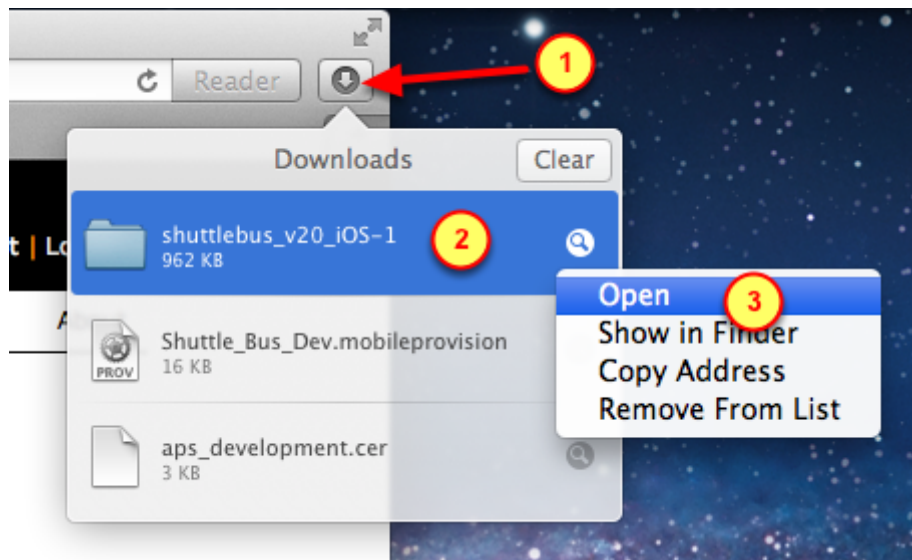


## 4. Download source code as a zip package



1. Click on the **Download .zip archive** link

## 5. Open the downloaded source code folder



1. Click on the **Downloads** icon
2. **Right-click** on the source-code package

3. Select **Open** to view the contents of that folder

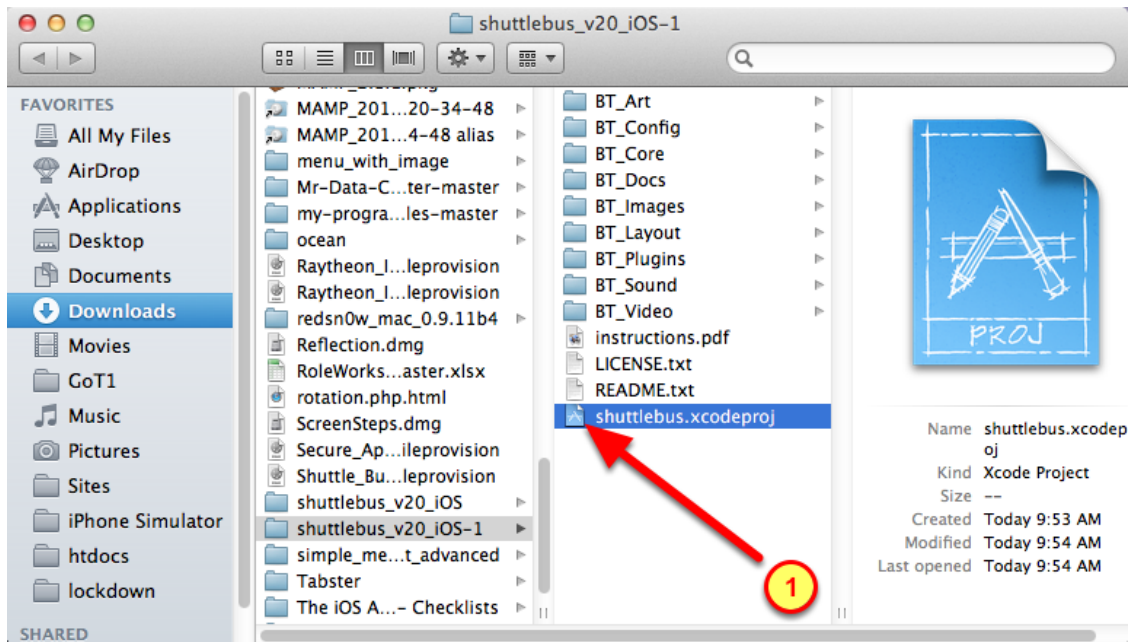


# Initial App setup in Xcode

## Open the App in Xcode

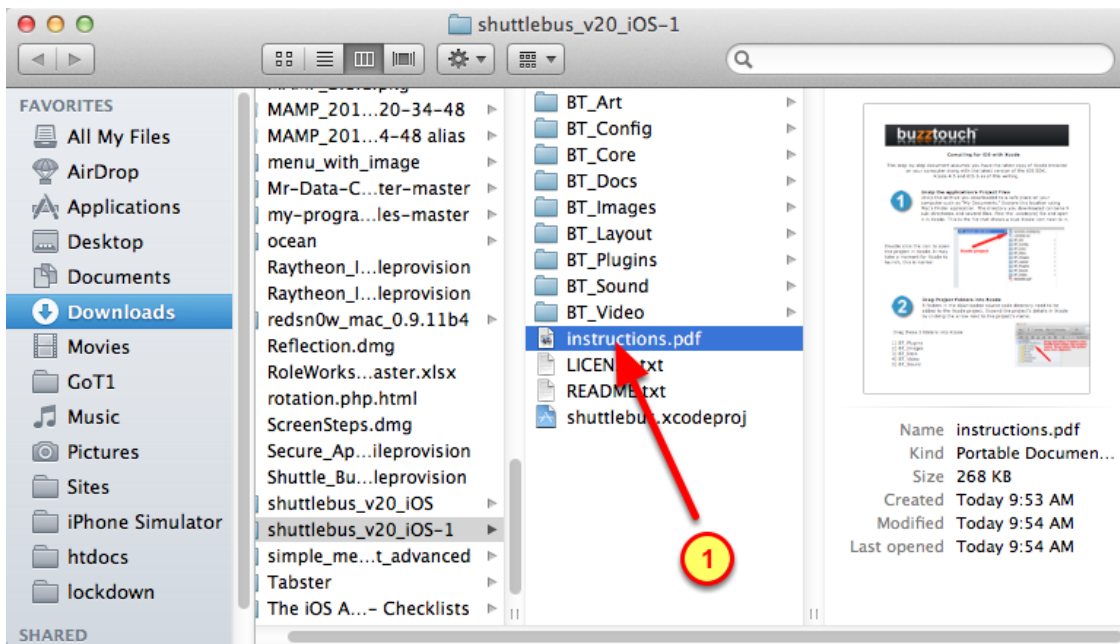
After downloading the Source Code of your app from the BuzzTouch server, five of the folders must be "connected" with the Xcode project.

### 1. Open the App in Xcode



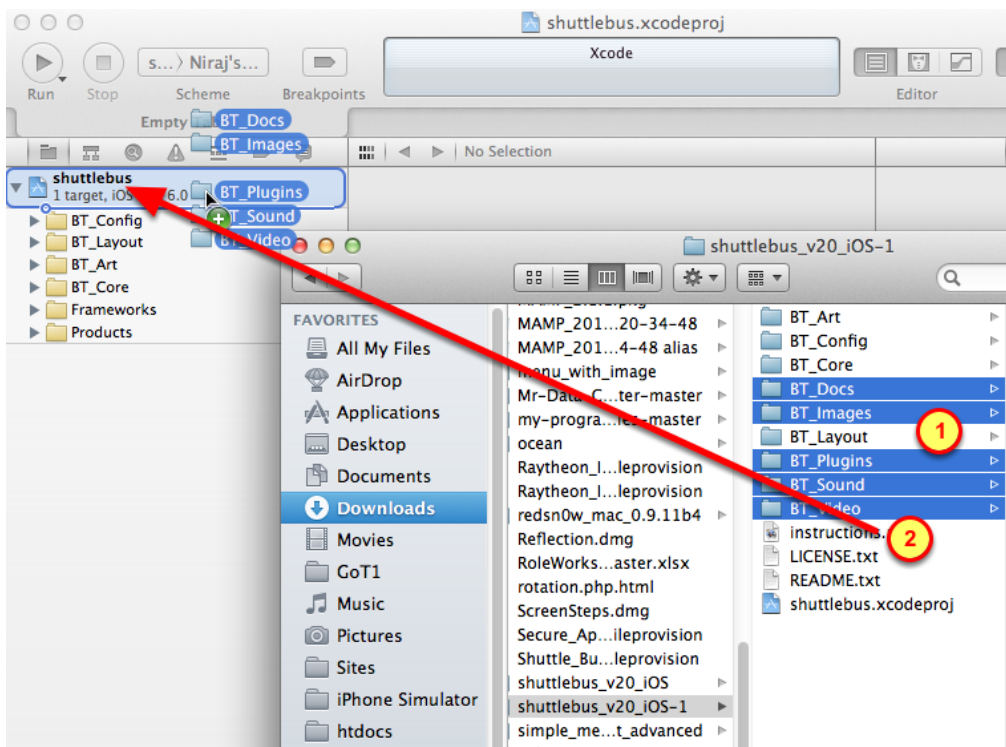
1. Open the App in Xcode by double-clicking its Xcode project xcodeproj file

## 2. Open the PDF of Instructions



Understand what to do with the source code by opening and reading the initial set of instructions.

## 3. Import folders into the Xcode project



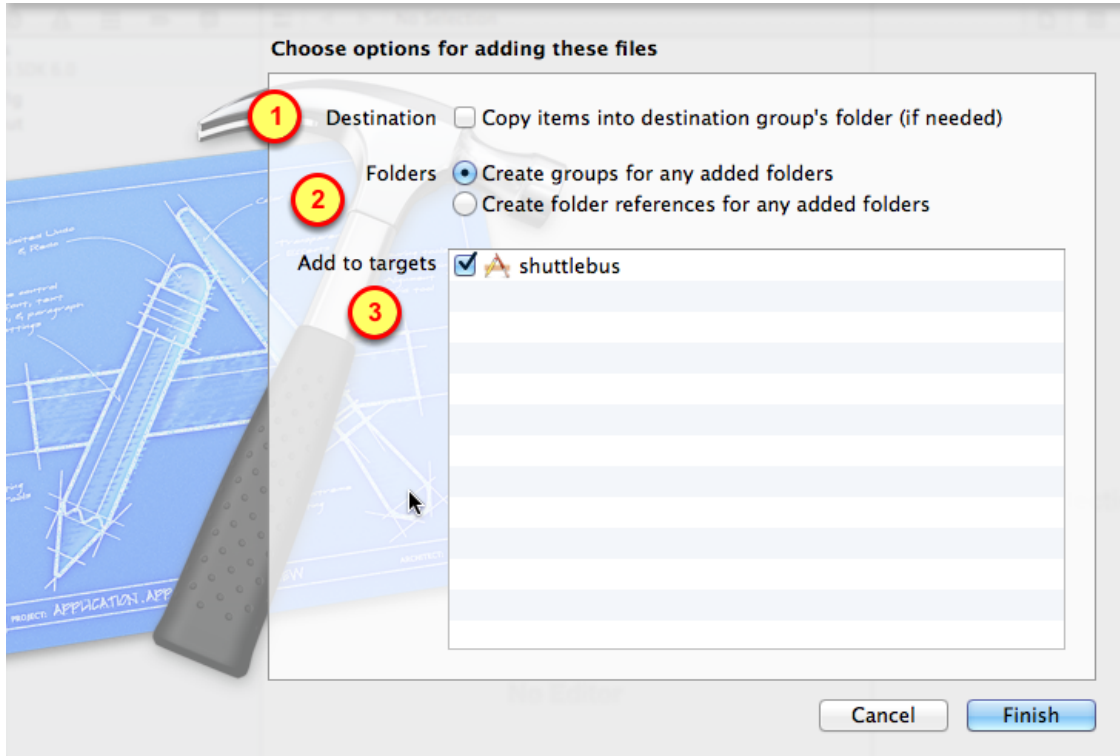
1. Using the Command key, select these folders

- BT\_Docs
- BT\_Images

- BT\_Plugins
- BT\_Sound
- BT\_Video

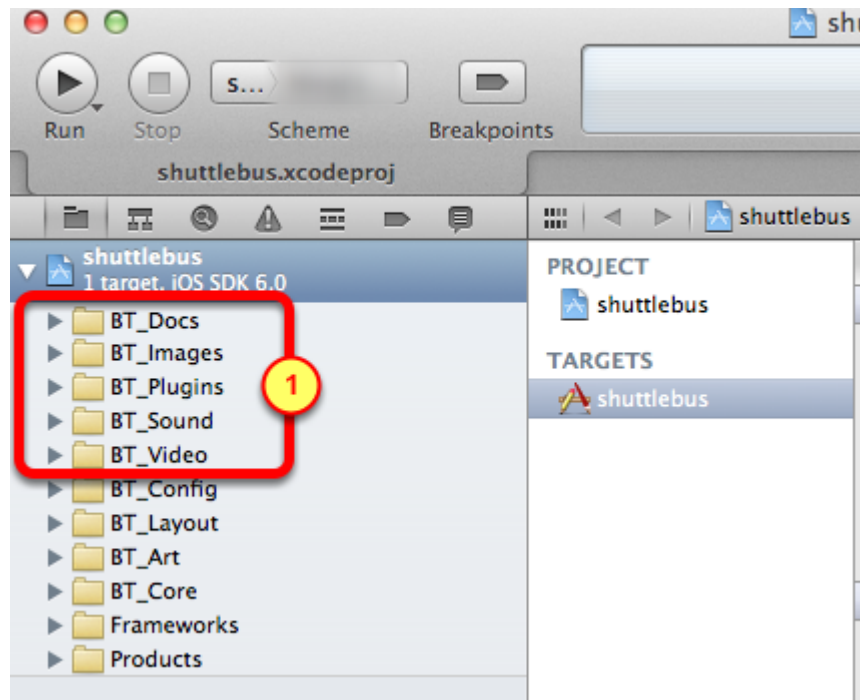
2. Drag-n-drop that group into the Xcode project

#### 4. Set parameters for import



1. Disable (do not check) the **Copy items ...**
2. Ensure Folders is set to "Create groups for any added folders"
3. Verify target is the new App

## 5. Verify new groups are in the Xcode project



1. Verify those groups were created when we imported the folders from the downloaded package

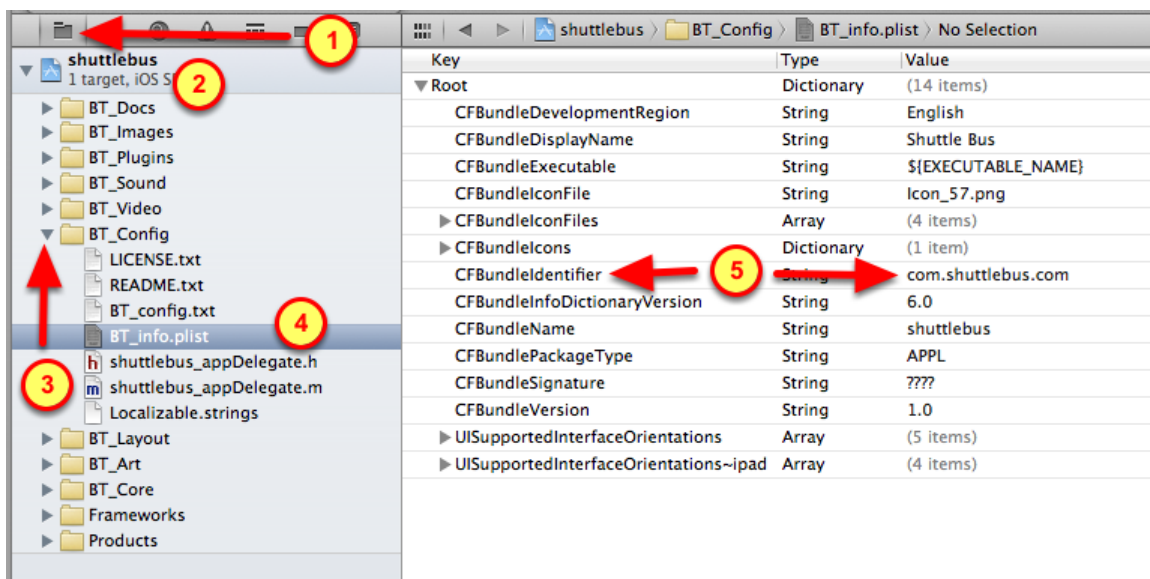
## Modify Bundle Identifier

When packaging the source code for downloading, the BuzzTouch server sets the Bundle Identifier to it's own naming convention of `com.AppName.com`

However, when we had created the App ID in earlier steps, the Bundle Identifier was specified in the format of `com.CompanyName.AppName`

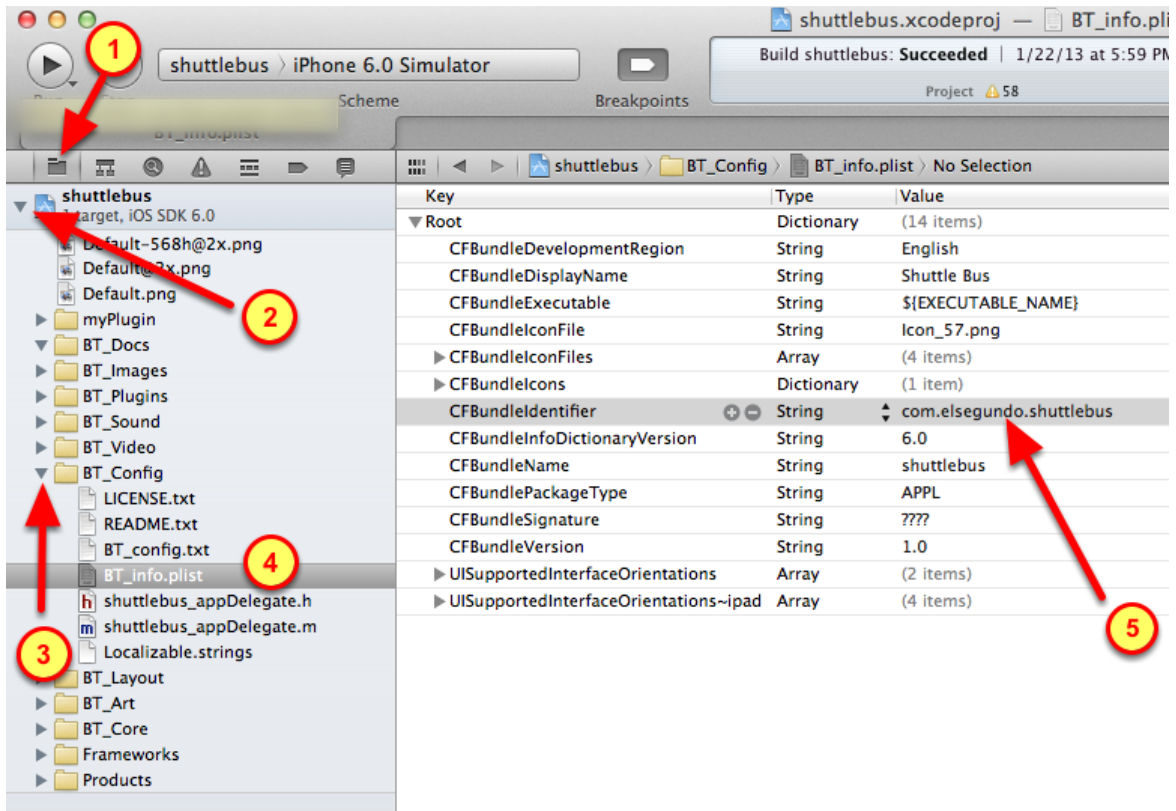
These steps will show how to change the Bundle Identifier within Xcode to match that of the App ID.

### 1. Find the Bundle Identifier



1. Open the **File Browser**
2. Expand the Xcode project by clicking on it's Left-triangle
3. Expand the **BT\_Config** group by clicking on it's Left-triangle
4. Click on the **BT\_info.plist** file for editing
5. Observe the value of the **CFBundleIdentifier**

## 2. Change the Bundle Identifier

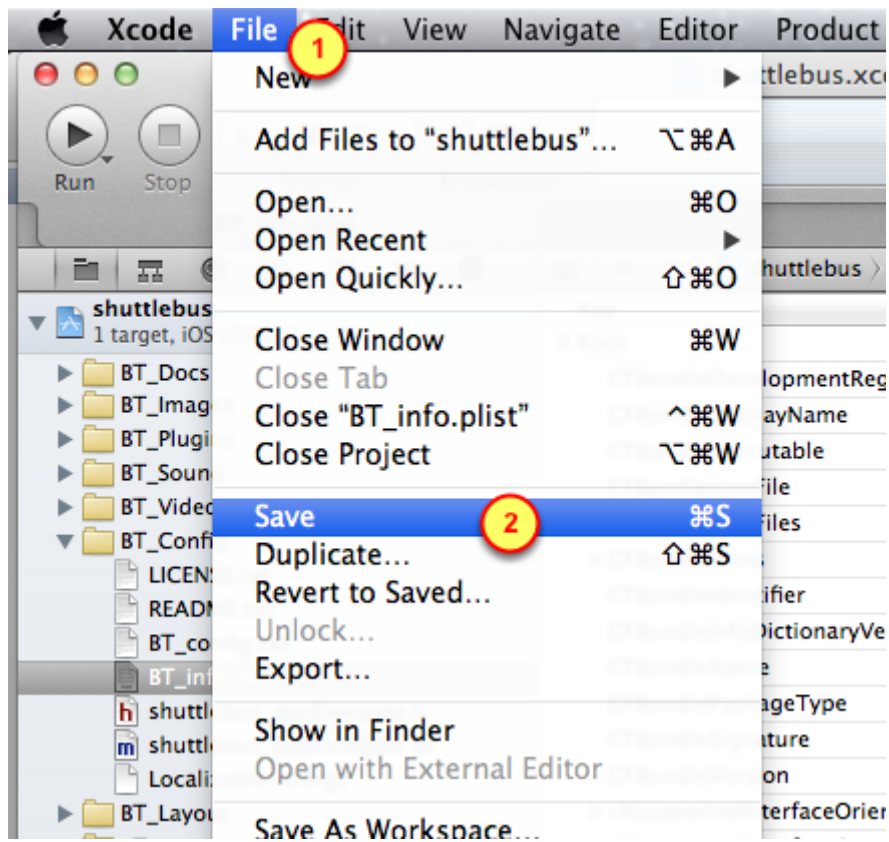


Lets change the Bundle Identifier to match the one associated with the App ID

1. Click on the File Browser icon of the Xcode project
2. Open the Xcode project by clicking on it's triangle
3. Open the **BT\_Config** group
4. Select the **BT\_info.plist** file for editing
5. For the **CFBundleIdentifier** row, double-click it's value and change it to match the value given when creating the App ID

( Do not use any of the numbers preceding the *com.CompanyName.AppName* )

### 3. Save the changes



Because we modified the CFBundleIdentifier to match our Company's name and App name, we should save the BT\_info.plist file

1. Open the **File** menu
2. Select the **Save** menu item



# Use Xcode to load App onto iPhone

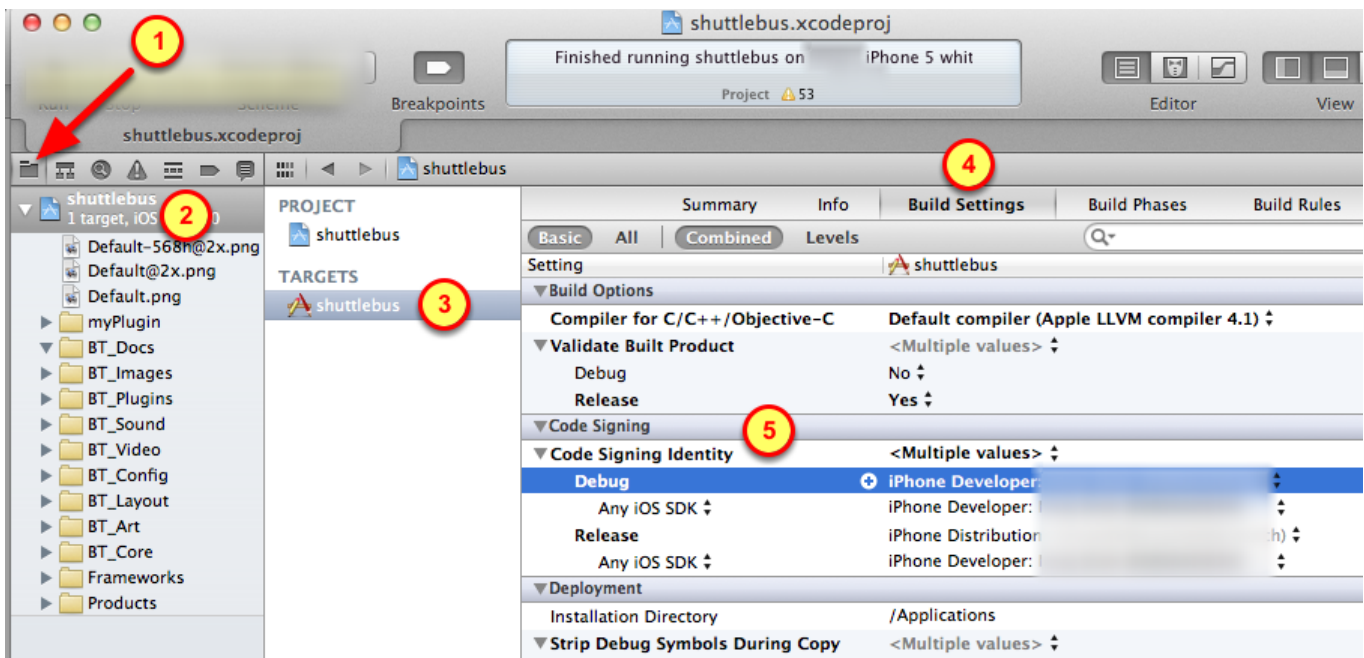
## Setup Xcode Project to use new Provisioning Profile

When compiling the App, we want the new Provisioning Profile to be used. That profile is associated with an App ID that is configured for Push Notifications.

Also, remember that the Developer is registered with the new Provisioning Profile.

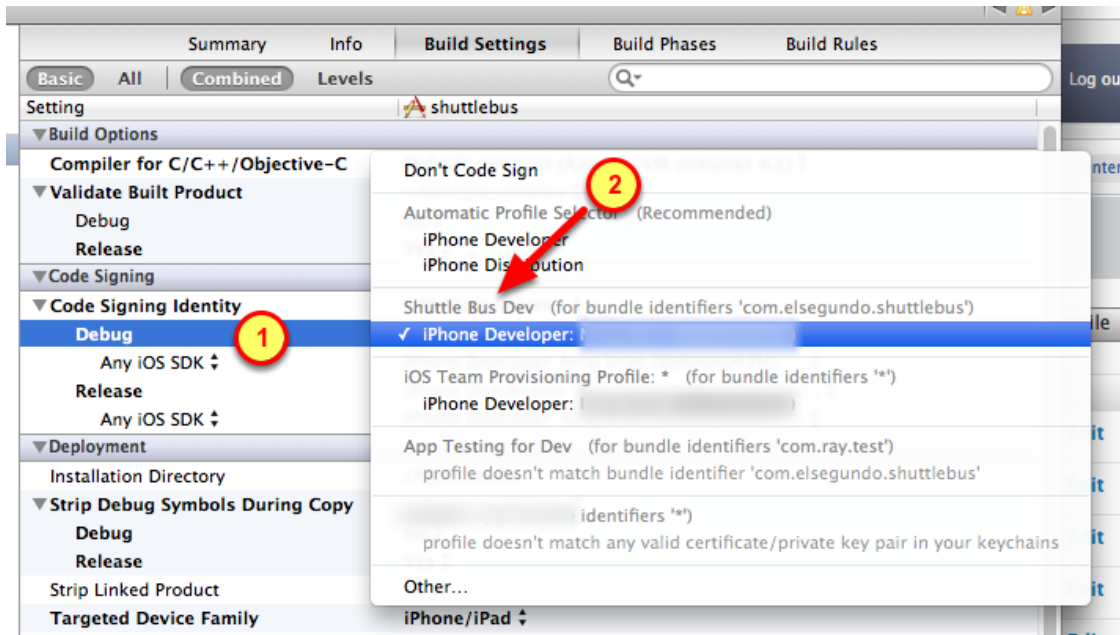
Thus, we have to specify the new Provisioning Profile to be used when the App is compiled.

### 1. Go into Build Settings



1. Open the **File Browser**
2. Select the **Project**
3. Select the **Target**
4. Select the **Build Settings** tab
4. Open the **Code Signing Identity** by clicking on it's triangle

## 2. Change code signing identity



Change the code signing identity to match the Provisioning Profile associated with the App ID that has Push Notifications enabled

1. Click on the **Debug** (or Release) row
2. Find and select the App's **Provisioning Profile** (notice where it says "Shuttle Bus Dev" in this example)

## *Run app on iPhone for testing of Push Notifications*

---

Since Push Notifications do not work on Simulators, the App must be tested on an actual device.

Connect your device (iPhone) to the Mac. Use Xcode to "sideload" the App onto the iPhone.

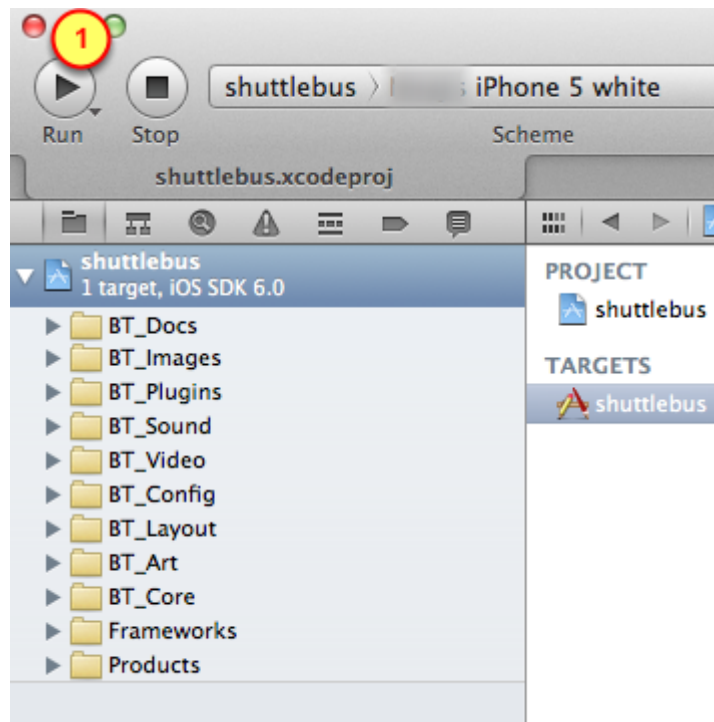
Then testing of Push Notifications for the App can begin!

### **1. Change from Simulator to iPhone**



1. Click on the previous setting, such as "iPhone 6.0 Simulator"
2. Change the device type to your iPhone

## 2. Run the app to install onto iPhone

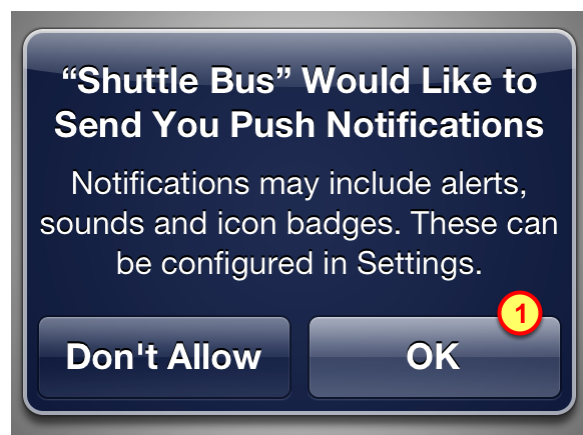


Since the App has run fine on the Simulator, it is ready to be loaded and ran on the iPhone

1. Click on the Run button to install and run the app

( Be patient, it takes a bit longer than it would for the Simulator )

## 3. Accept notifications for the App



To receive notifications from the BuzzTouch Control Panel, the User must allow the App to receive notifications on his iPhone.

1. Tap on "**OK**" to enable notifications for this App on this iPhone

# **Send a notification from BuzzTouch Control Panel**

## Send Push Notifications

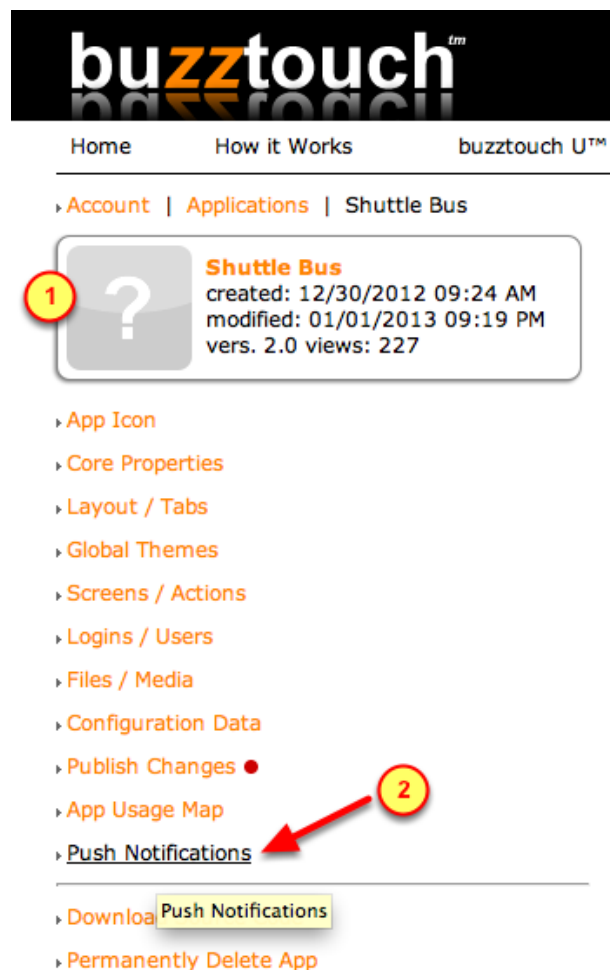
---

When the User accepts the prompt asking for permission to send him push notifications for the App, the iPhone then sends a token to Apple's Push Notification Server. That essentially registers the Device to receive notifications for the App.

(We had previously configured the Core settings for Push Notification to prompt the User for permission)

After a device has been registered to receive Push Notifications, we can test by sending actual messages to the Development devices!

### 1. Push Notification Menu

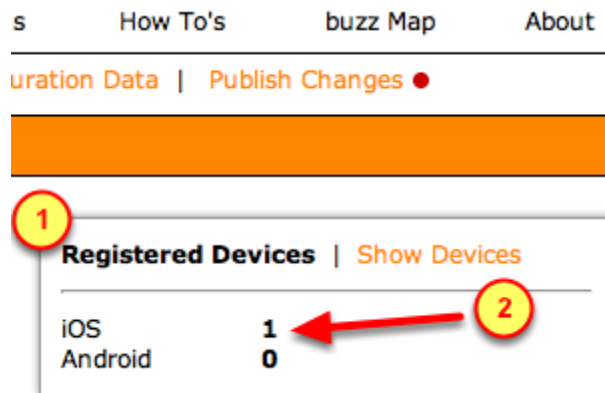


1. Ensure you are in the correct **Application's** Control Panel.

2. Click on the "**Push Notifications**" menu



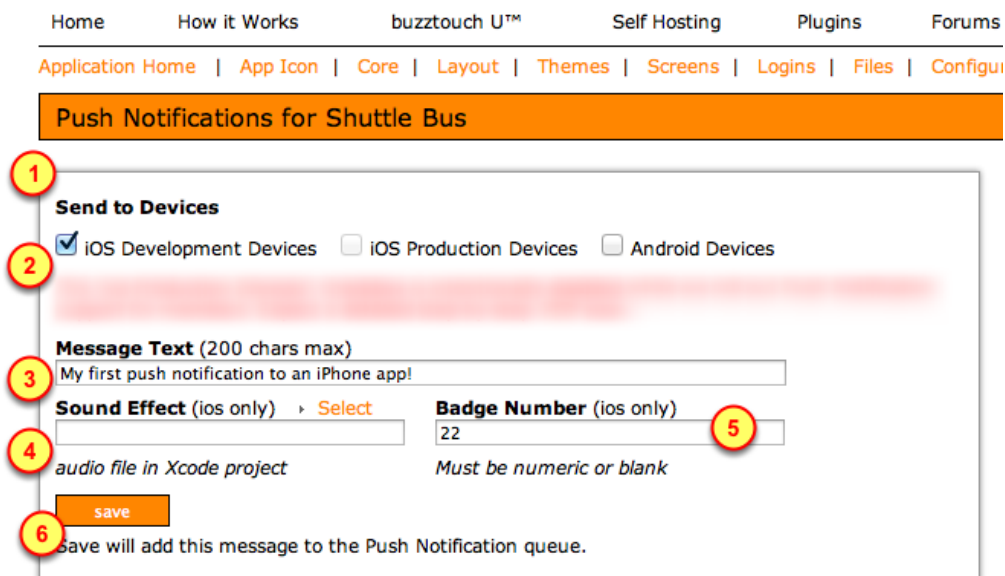
## 2. Ensure device is registered for Push Notifications



During the first-running of the App on the iPhone, you were asked if Push Notifications were allowed (for that app on that device). By saying "yes", that registered the iPhone with Apple and the BuzzTouch server.

1. Look at the **Registered Devices** section on the page
2. Verify the **quantity** of registered devices for iOS is correct

## 3. Add a Push Notification into the Queue



1. Look at the **Send to Devices** section of the screen
2. Enable the sending to **iOS Development Devices** (or iOS Production Devices)

3. Type something into the **Message Text** field
4. (optional) Select a previously uploaded **Sound Effect**
5. (optional) Specify a numeric value for the **Badge Number** (red circle with that number will be displayed on the icon of the App)
6. Click the **save** button

## 4. Begin sending from queue

Home    How it Works    buzztouch U™    Self Hosting    Plugins    Forums

Application Home | App Icon | Core | Layout | Themes | Screens | Logins | Files | Configur

### Push Notifications for Shuttle Bus

**Send to Devices**

iOS Development Devices     iOS Production Devices     Android Devices

**Message Text** (200 chars max)

**Sound Effect** (ios only)    [Select](#)    **Badge Number** (ios only)

*audio file in Xcode project*    *Must be numeric or blank*

**save**

Save will add this message to the Push Notification queue.

---

**Push Notification Queue**

Created: 01/27/13 01:12 AM    [begin sending](#) | [remove from queue](#)

Payload: {"apps":{"alert":"My first push notification to an iPhone app!", "badge":22, "sound":"default"}}}

Send To: 1 iOS devices 0 Android devices

1

1. Click on **begin sending** the notifications from the queue

(You will get a chance to Cancel or to Confirm in the next step)

## 5. Confirm the sending from the queue

Home    How it Works    buzztouch U™    Self Hosting    Plugins    Forums

[Application Home](#) | [App Icon](#) | [Core](#) | [Layout](#) | [Themes](#) | [Screens](#) | [Logins](#) | [Files](#) | [Configur](#)

### Push Notifications for Shuttle Bus

#### Send to Devices

iOS Development Devices     iOS Production Devices     Android Devices

**Message Text** (200 chars max)

**Sound Effect** (ios only)    [Select](#)    **Badge Number** (ios only)

*audio file in Xcode project*    *Must be numeric or blank*

[save](#)


Save will add this message to the Push Notification queue.

#### Push Notification Queue

Created: 01/27/13 01:12 AM    [begin sending](#) | [remove from queue](#)

Payload: {"apps":{"alert":"My first push notification to an iPhone app!", "badge":22, "sound":"default"}}    [confirm](#) | [cancel](#)

Send To: **1** iOS devices **0** Android devices



1. Click on **confirm** to send all messages that are in the queue

## 6. Verify the notifications were sent

Home    How it Works    buzztouch U™    Self Hosting    Plugins    Forums

[Application Home](#) | [App Icon](#) | [Core](#) | [Layout](#) | [Themes](#) | [Screens](#) | [Logins](#) | [Files](#) | [Configur](#)

### Push Notifications for Shuttle Bus

#### Send to Devices

iOS Development Devices     iOS Production Devices     Android Devices

**Message Text** (200 chars max)

**Sound Effect** (ios only)    [Select](#)    **Badge Number** (ios only)

*audio file in Xcode project*    *Must be numeric or blank*

[save](#)


Save will add this message to the Push Notification queue.

#### Push Notification Queue

Created: 01/27/13 01:12 AM

Payload: {"apps":{"alert":"My first push notification to an iPhone app!", "badge":22, "sound":"default"}}}

Send To: **1** iOS devices **0** Android devices

 **Notifications Sent**

1. Verify the notifications were **sent** out to the devices from the queue

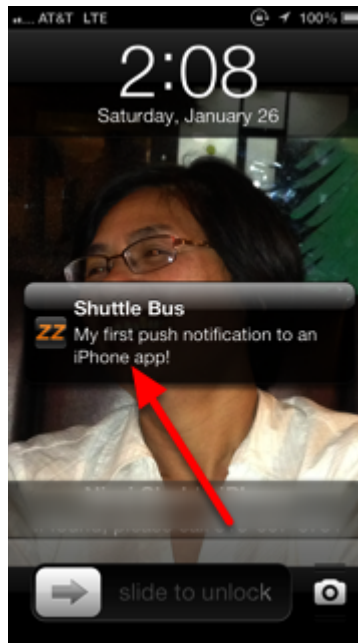
## *Check the iPhone for a push notification*

---

After sending the queue of notifications from the BuzzTouch Control Panel, the messages are then distributed to the Devices by Apple's Push Notification System (APNS).

Check your devices to verify the receipt of the push notifications, the sound and the badge.

### **1. Notification appears on lock screen**

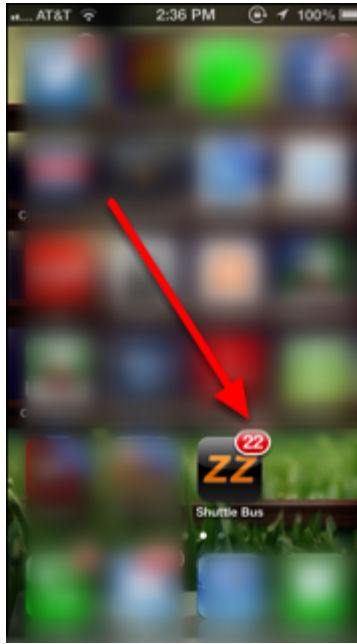


A wee bit after you confirmed with the BuzzTouch Control Panel, the notification arrived on the registered devices.

In this case, the device was sleeping, only showing its lock screen.

Upon receipt of the push notification, the iPhone was awoken to briefly display the push message that was entered into the BuzzTouch Control Panel.

## 2. Badge on App's icon



Along with receiving a push message, the App's icon can also be configured to display a "badge". The purpose of a Badge is to inform the User that the App has something new that may be of interest. The Badge is a means of encouraging the User to launch the App.

The badge number was entered into the BuzzTouch Control Panel and sent along with the push message. Upon receipt of the Push Notification, the badge number is then displayed on the icon of the App.

# Production usage of Push Notifications

## Production Push Notifications

If you diligently followed all the steps in this tutorial, then a hearty congratulations to you! It feels great to have such a powerful way of engaging Users with your App.

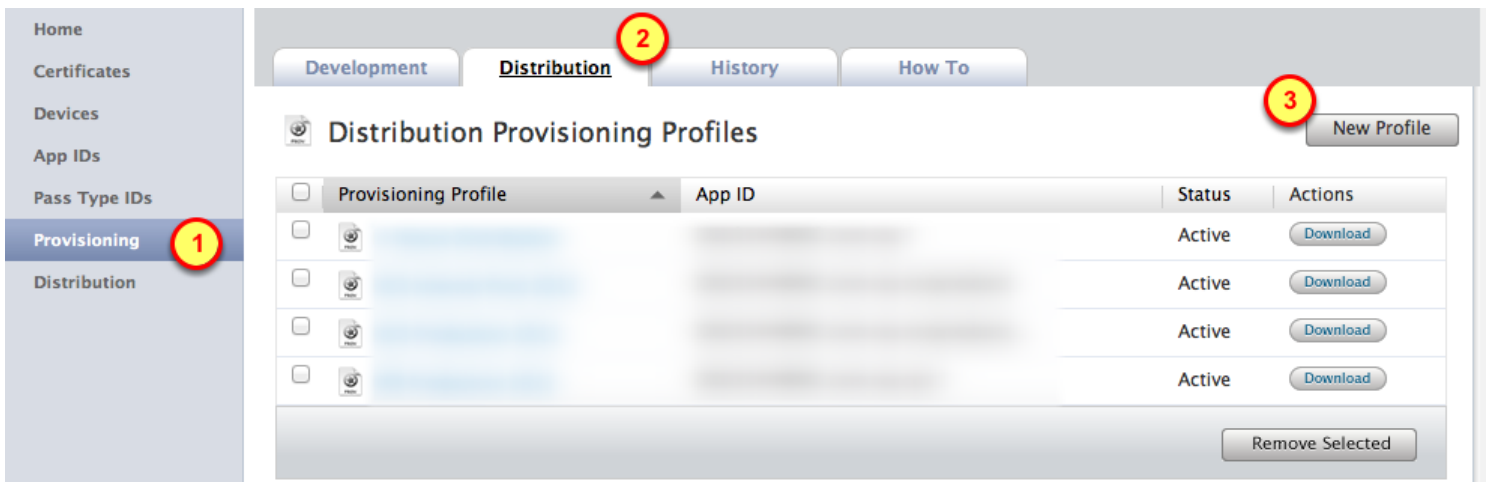
Now that you've tested the Push Notifications with your Test Devices, you're ready to do it for real on with Apps published on Apple's App Store.

To be in production mode, repeat this Tutorial's steps with these specific changes:

1. Use the Distribution section of Apple's Provisioning Portal to create a production Provisioning Profile
2. Upload a Production .PEM Certificate into the BuzzTouch Control Panel
3. Enable the Production Devices in the Push Notifications section of the BuzzTouch Control Panel
4. Send push notifications to Users of your Apps

Here is a brief set of steps that help to jog your memory on the differences between Development mode and Production mode. Bear in mind that you'll still have to re-execute many of the steps documented elsewhere in this tutorial.

### 1. Create a Distribution Provisioning Profile



Go into Apple's Provisioning Portal (follow previous steps in this Tutorial)

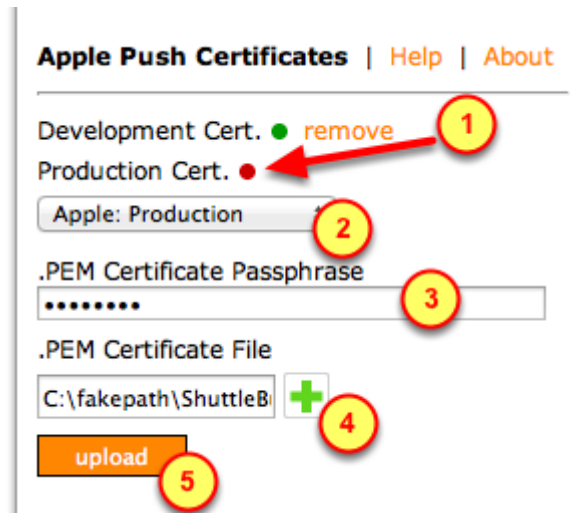
1. Select the **Provisioning** menu item
2. Select the **Distribution** folder tab



3. Click on **"New Profile"**

Then follow previous steps in this Tutorial for obtaining a Certificate.

## 2. Upload production PEM certificate into BuzzTouch server



Navigate to the **Apple Push Certificates** section by going to the App's Control Panel, then click on Push Notifications.

1. Due to the **red-dot**, notice the Production certificate has not yet been uploaded.
2. Select **"Production"** as the type of certificate to be uploaded
3. Type in a random **passphrase**, you won't have to remember it's value
4. Click on the Plus-sign to invoke the file browser (to point it to the .PEM certificate file)
5. Click on the **upload** button to transfer the .PEM certificate to the BuzzTouch server

### 3. Add a Push Notification into the Queue

Home    How it Works    buzztouch U™    Self Hosting    Plugins    Forums

[Application Home](#) | [App Icon](#) | [Core](#) | [Layout](#) | [Themes](#) | [Screens](#) | [Logins](#) | [Files](#) | [Configur](#)

#### Push Notifications for Shuttle Bus

**1** **Send to Devices**

iOS Development Devices **2**  iOS Production Devices  Android Devices

**3** **Message Text** (200 chars max)  
My first push notification to an iPhone app!

**4** **Sound Effect** (ios only) [Select](#) **Badge Number** (ios only) **5**  
 Must be numeric or blank

**6**

save will add this message to the Push Notification queue.

Navigate to the **Apple Push Certificates** section by going to the App's Control Panel, then click on Push Notifications.

1. Look at the **Send to Devices** section of the screen
2. Enable the sending to **iOS Production Devices** (and iOS Development Devices)
3. Type something into the **Message Text** field
4. (optional) Select a previously uploaded **Sound Effect**
5. (optional) Specify a numeric value for the **Badge Number** (red circle with that number will be displayed on the icon of the App)
6. Click the **save** button

## 4. Send notifications from the queue

Home    How it Works    buzztouch U™    Self Hosting    Plugins    Forums

Application Home | App Icon | Core | Layout | Themes | Screens | Logins | Files | Configur

### Push Notifications for Shuttle Bus

#### Send to Devices

iOS Development Devices     iOS Production Devices     Android Devices

**Message Text** (200 chars max)

**Sound Effect** (ios only) ▶ [Select](#)    **Badge Number** (ios only)

*audio file in Xcode project*    *Must be numeric or blank*

[save](#)

Save will add this message to the Push Notification queue.

#### Push Notification Queue

Created: 01/27/13 01:12 AM    ▶ [begin sending](#) | [remove from queue](#)

Payload: {"apps":{"alert":"My first push notification to an iPhone app!", "badge":22, "sound":"default"}}}

Send To: **1** iOS devices **0** Android devices

**1**

Navigate to the **Apple Push Certificates** section by going to the App's Control Panel, then click on Push Notifications.

1. Click on **begin sending** the notifications from the queue

(You will get a chance to Cancel or to Confirm in the next step)