## 苹果程序打包及推送问题说明

### 一、概念介绍

#### 1、Certificate

证书是用来给应用程序签名的，只有经过签名的应用程序才能保证他的来源是可信任的，并且代码是完整的， 未经修改的。在Xcode Build Setting的Code Signing Identity中，可以设置用于为代码签名的证书。

证书主要分为两类：Development和Production，Development证书用来开发和调试应用程序，Production主要用来分发应用程序（根据证书种类有不同作用），下面是证书的分类信息：（括号内为证书有效期）

1)、Development

App Development (1年)：用来开发和真机调试应用程序。

Push Development (1年)：用来调试Apple Push Notification

2)、Production

Ad Hoc ：用来发布AdHoc的应用程序。

App Store ：用来发布提交App Store的应用程序。

Push Production (1年)：用来在发布版本中使用Apple Push Notification。

#### 2、App ID

App ID用于标识一个或者一组App，App ID应该是和Xcode中的Bundle ID是一致的或者匹配的。App ID主要有以下两种：

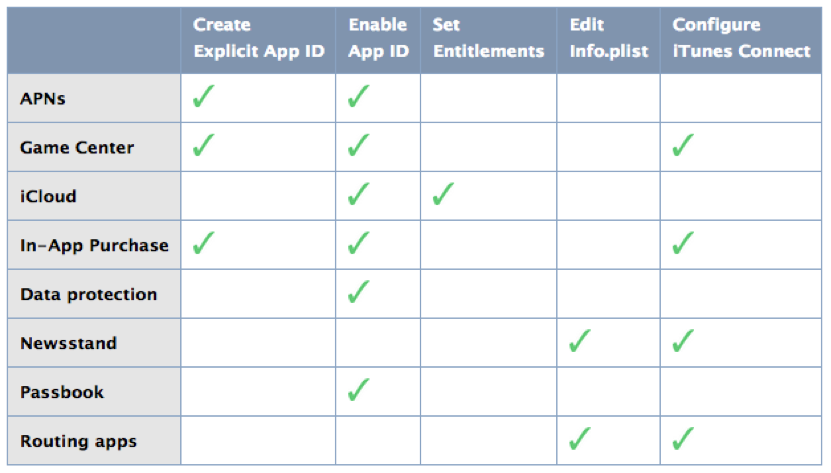
1)、Explicit App ID：唯一的App ID，这种App ID用于唯一标识一个应用程序，例如com.wzhi.demo，标识Bundle ID为com.wzhi.demo的程序。

2)、Wildcard App ID：通配符App ID，用于标识一组应用程序。例如\*

可以表示所有应用程序，而com.wzhi.\*可以表示以com.wzhi开头的所有应

用程序。

  每创建一个App ID，我们都可以设置该App ID所使用的APP Services。每种服务都有着不同的要求，例如，如果要使用Apple Push Notification Services，则必须是一个explicit App ID，以便能唯一标识一个应用程序。下面是目前所有可选的服务和相应的配置要求。



如果你的App使用上述的任何一种service，就要按照要求去配置。

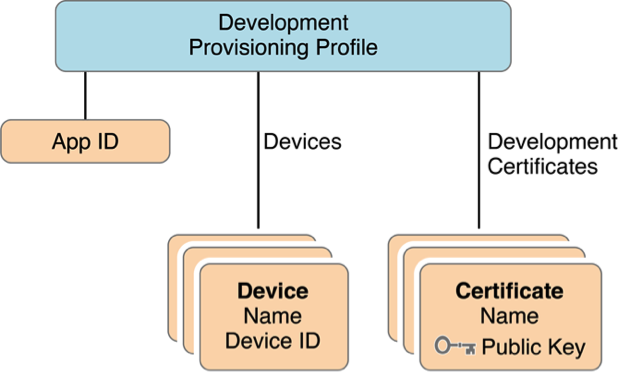
#### 3、Device

Devices中包含了该账户中所有可用于开发和测试的设备。 每台设备使用UDID来唯一标识。每个账户中的设备数量限制是100个。

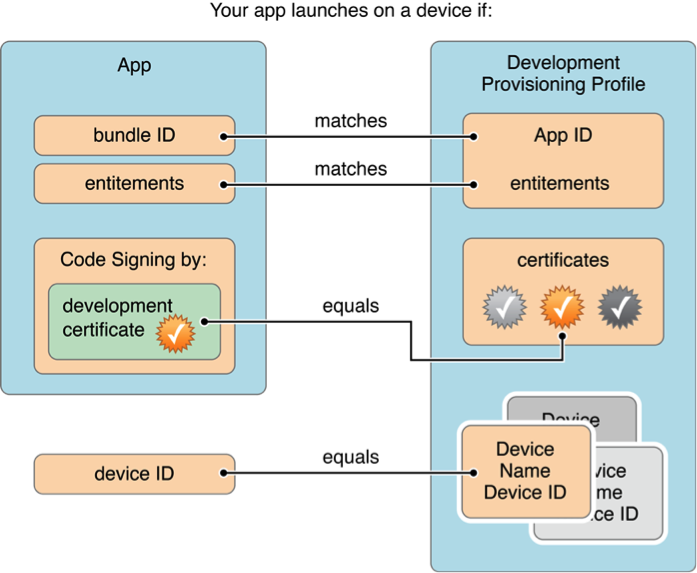
#### 4、Provisioning Profile

一个Provisioning Profile文件包含了上述的所有内容：证书、App ID、设备。要打包或者在真机上运行一个应用程序，首先需要证书来进行签名，用来标识这个应用程序是合法的、安全的、完整的；然后需要指明它的App ID，并且验证Bundle ID是否与其一致；再次，如果是真机调试，需要确认这台设备能否用来运行程序。Provisioning Profile就把这些信息全部打包在一起，方便在调试和发布程序打包时使用，这样只要在不同的情况下选择不同的profile文件就可以了。而且Provisioning Profile文件会在打包时嵌入.ipa的包里。

例如，如下图所示，一个用于Development的Provisioning Profile中包含了该Provisioning Profile对应的App ID，可使用的证书和设备。这意味着使用这个Provisioning Profile打包程序必须拥有相应的证书，并且将App ID对应的程序运行到Devices中包含的设备上去。



如上所述，在一台设备上运行应用程序的过程如下：



与证书一样，Provisioning Profile也分为Development和Distribution两种：

1)、Development (1年)

2)、Distribution (1年)

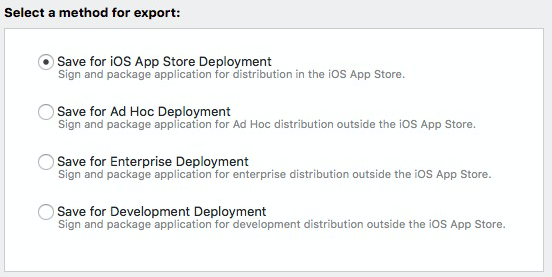
Ad Hoc

App Store

Ad Hoc的包只能运行在该账户内已登记的可用设备上，显然是有最多100个设备的数量限制。所以这两种Provisioning Profile文件的区别就在于其中的设备限制不一样而已，而他们所使用的Certificate是相同的。

### 二、程序打包介绍

#### 1、苹果程序打包方式介绍



1)、Save for iOS App Store Deployment

保存到本地 准备上传App Store 或者在越狱的iOS设备上使用

2)、Save for Ad Hoc Deployment

保存到本地 准备在账号添加的可使用设备上使用（具体为在开发者账户下添加可用设备的udid），该app包是发布证书编译的.

3)、Save for Enterprise Deployment

这种主要针对企业级账户下 准备本地服务器分发的app

4)、Save for Development Deployment

针对内部测试使用，主要给开发者的设备(具体也为在开发者账户下添加可用设备的udid)。该app包是开发证书编译的

我们现在提供给测试打包，测试环境和HK环境都使用第三种方式打包(Save for Enterprise Deployment)，这样程序安装包可以安装在任意一台iPhone手机上，方便测试发给市场等公司内部同事安装体验；ME环境使用第二种方式打包(Save for Ad Hoc Deployment)，第二种方式使用的发布证书与正式提交AppStore的版本使用的证书是一样的，因此他们可以共用同一推送证书，这样就可以测试ME环境的推送流程是否有问题了,这种方式打包的程序只可以运行在我们已经添加到开发账户对应的devices设备上使用。

#### 2、用户版打包说明

1)、给测试打包

a．HK环境和测试环境的测试包：

使用bundleid：com.weizhi.consumer.enterprise

程序中的AMAP\_API\_KEY的值改为 com.weizhi.consumer.enterprise对应的值 71fa130247ded477c0e63e12316353cd, Provisioning Profile使用 consumer\_enterprise\_distribution，证书使用其对应的发布证书。

WZConst.h文件中选择对应的宏定义：DEMO\_VERSION或TEST\_VERSION

导出包的方式选择： Save for Enterprise Deployment

b. ME环境的测试包

使用bundleid：com.weizhi.consumer

程序中的AMAP\_API\_KEY的值改为 com.weizhi.consumer对应的值 c4fe9f90ae3db10104b2db66f13a6898, Provisioning Profile使用 user\_adhoc，证书使用其对应的发布证书。

WZConst.h文件中选择对应的宏定义：ONLINE\_VERSION

导出包的方式选择：Save for Ad Hoc Deployment

2)、提交Appstore版本打包

使用bundleid:com.weizhi.consumer

程序中的AMAP\_API\_KEY的值改为com.weizhi.consumer对应的值 c4fe9f90ae3db10104b2db66f13a6898，Provisioning Profile使用 user\_distribution，证书使用其对应的发布证书

WZConst.h文件中选择对应的宏定义：ONLINE\_VERSION

导出包的方式选择：Save for iOS App Store Deployment

#### 3、商户版打包说明

1)、给测试打包

a．HK环境和测试环境的测试包：

使用bundleid：com.weizhi.shop.enterprise

程序中百度地图的key值改为 com.weizhi.shop.enterprise对应的值 iGT2G4SakUesu52BkO5HXUBn, Provisioning Profile使用 shop\_enterprise\_distribution，证书使用其对应的发布证书。

TFInterface.h文件中选择对应的宏定义：DEMO\_VERSION或TEST\_VERSION

导出包的方式选择： Save for Enterprise Deployment

b. ME环境的测试包

使用bundleid：com.tuanbao.weizhi

程序中百度地图的key值改为 com.tuanbao.weizhi对应的值 71b796a40afc797d0bacaca6d291f9b1, Provisioning Profile使用 shop\_ad\_hoc，证书使用其对应的发布证书。

TFInterface.h文件中选择对应的宏定义：ONLINE\_VERSION

导出包的方式选择：Save for Ad Hoc Deployment

2)、提交Appstore版本打包

使用bundleid: com.tuanbao.weizhi

程序中百度地图的key值改为 com.tuanbao.weizhi对应的值 71b796a40afc797d0bacaca6d291f9b1, Provisioning Profile使用 shop\_distribution，证书使用其对应的发布证书。

TFInterface.h文件中选择对应的宏定义：ONLINE\_VERSION

导出包的方式选择：Save for iOS App Store Deployment

### 三、备注

#### 1、使用苹果的推送服务，需要注意以下几点：

a. APNS是免费的。只要有开发者账号便可以申请APNS证书。

b. APNS是不可靠的，苹果对信息推送的可靠性不做任何保证。

c. APNS对消息的大小是有限制的，总容量不能超过256字节。

如果需要推送成功，需要保证 程序打包使用的证书 和服务器端推送使用的证书以及推送网址需要保持一致，才可以推送成功。一般对应逻辑如下：

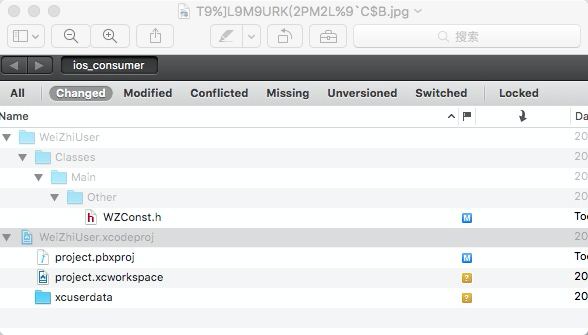
Product证书--->Product推送证书--->Product证书打包的程序对应的devicetoken--->Product推送网址(ssl://gateway.push.apple.com:2195)

Development证书--->Development推送证书--->Development证书调试的程序对应的devicetoken--->Development推送网址(ssl://gateway.sandbox.push.apple.com:2195)。

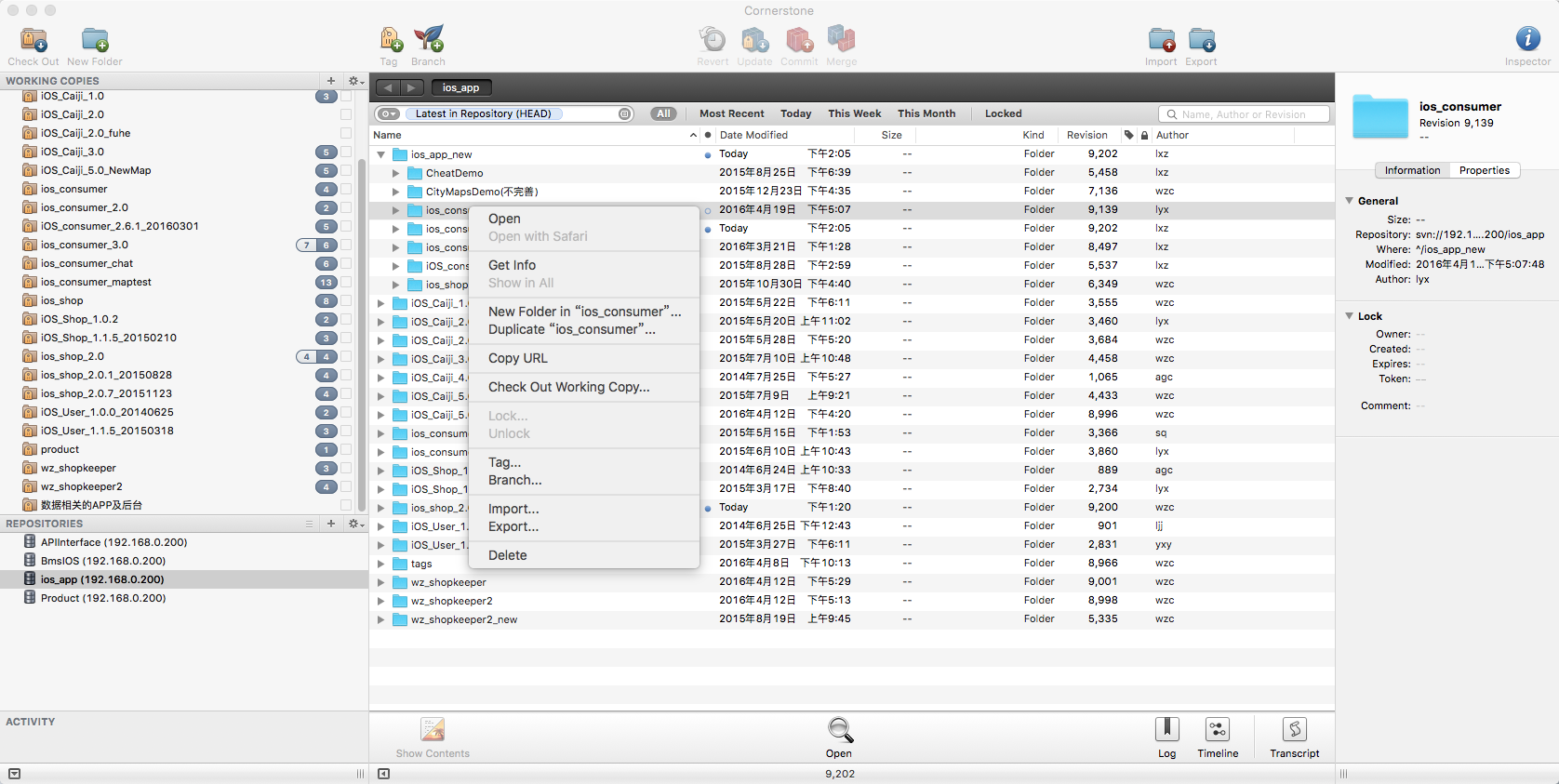
#### 2、打包程序需要注意以下几点：

a. 程序使用的环境要正确；

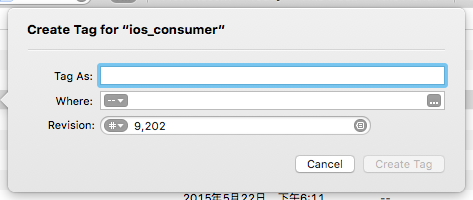
b．打包ME环境的测试包或者提交AppStore的包，一定要检查本地的程序代码是否有不必要的修改，（逐一核对Cornerstone该代码目录对应的changed文件，如下图所示，所有无关的修改都要还原）；



c. 打包完成后要做tag记录，如下图所示：



选择tag后会弹出如下界面：



Tag As:tag的名称，一般为做tag记录的文件夹名称（上图所示的ios\_consumer）\_版本号\_日期

Where：tag对应的路径，一般选择ios\_app/tags

Revision：一般默认的为最后提交的版本

#### 3、参考网址

推送证书的生成：

<http://blog.csdn.net/wzc10101415/article/details/51119416>

php实现苹果推送所需证书文件生成：

<http://blog.csdn.net/wzc10101415/article/details/51119720>

打包方式介绍：

<http://www.ithao123.cn/content-10472652.html>

苹果开发Certificate、Provisioning Profile等概念介绍：

<http://blog.csdn.net/joosonmao/article/details/21172835>