AppStore提包规范

第一章 提包注意事项

1. 提包所需信息
2. 游戏名称+副标题
3. 描述信息+宣传文本+关键词
4. icon+美宣图
5. 隐私协议+技术支持
6. 设备及网络环境
7. 开发者账号只能在专用的提包设备（Mac mini）、网络环境（使用VPN）下登录，登录后填写对应的iTC信息。

\*\* 提包使用的开发者账号不允许在自己开发设备、未使用VPN的网络下登录

1. 生成开发证书、发布证书选择手动生成，并且也只能在专用环境下生成。不可与自己的开发设备混淆。
2. 打包上传后在当前提包设备的钥匙串中删除掉对应的开发证书、发布证书。
3. 查看包的审核状态，安装网易邮箱大师绑定开发者账号对应邮箱，通过邮件查看状态变化。

**新版热更+压缩注意点**

**begin.ts**

let resource = "abc" // 文件名，保证文件名不同

let suffix = "png" // 后缀格式，用来伪装文件。可用 png\gif\jpg\pdf\doc\ppt\ 等任意格式

最终分割的文件名应为： abc.00.png

存放路径： encryptstr = enc\_s("res/") ；可选择放在res路径下的其他目录，只要能确保找到该目录。也可以自定义目录，即在res里新建一个目录（”res/zipPath/“）

**tool.ts**

// 随机挑选几组数字修改

let ENCRYPT\_KEY = [100, 101, 102, 103, 104, 111, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99];

// 随机挑选几个字符修改

let ENCRYPT\_IV = "poiuytreqwekjhgf";

运行后复制加密后的字符串到脚本中对应位置

**压缩包**

1. 使用脚本生成压缩包（abc.zip），注意：压缩密码、文件名 确保不重复

2. 压缩包分割:压缩包分割使用随机大小，生成的碎文件没有规律

3. 压缩包 添加可识别格式的目的是为了伪装文件，故 后缀格式最好为可识别的文件格式

**jsClass**

1~~. 分割了jsClass 类的实现到三个不同的类，添加完后使用脚本修改这三个类的类名~~

2. DJZipTool 每次都要修改该类名，确保不重复

3. 所有准备工作完成后测试解压、热更，正常启动游戏，测试游戏正常，测试官方代充功能正常

4. 添加混淆代码调用到 jsClass

**基础包**

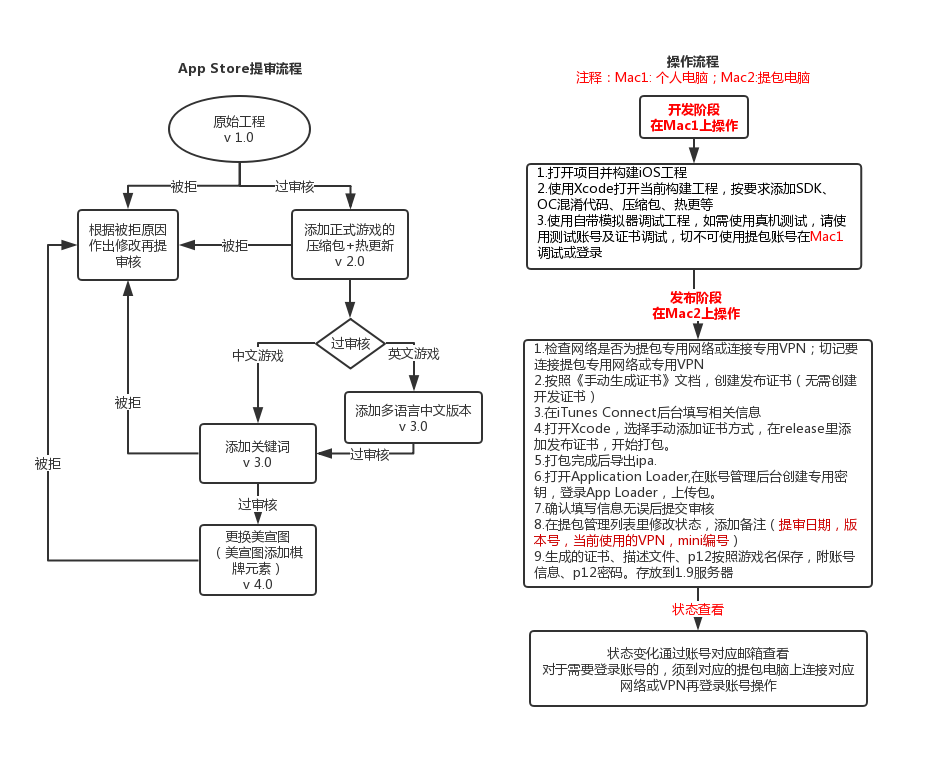
新资源留一半左右不打到图集里，不打图集的不要使用编号形式的名字，一定要改成其他名字。

**更新包**

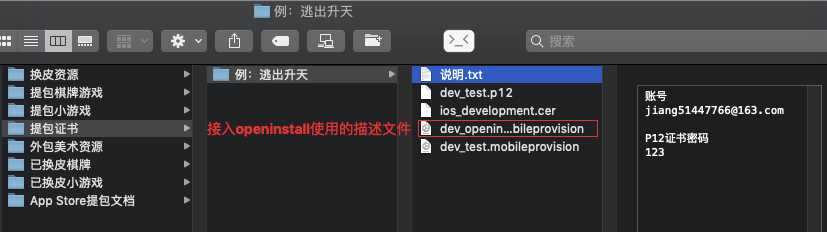
1. 把原本没有打到图集里的放一部分到图集，留一部分不打图集的用作藏压缩包

2. 更改jsclass 类 .h定义的变量名；.m定义的常量、变量、成员函数（不影响外部调用的函数都可以修改）

第二章 提包流程



**生成的证书、描述文件、p12统一备份到 smb://192.168.1.9 服务器上，格式如下**

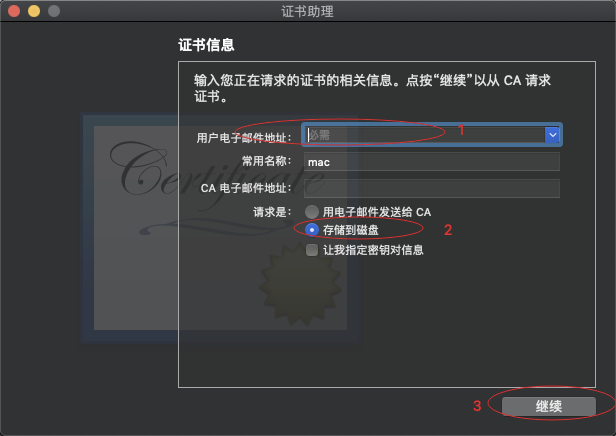


第三章 手动生成证书+描述文件

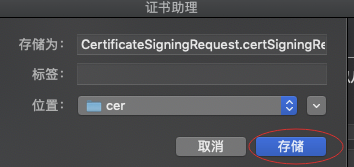
1. 在本机生成证书



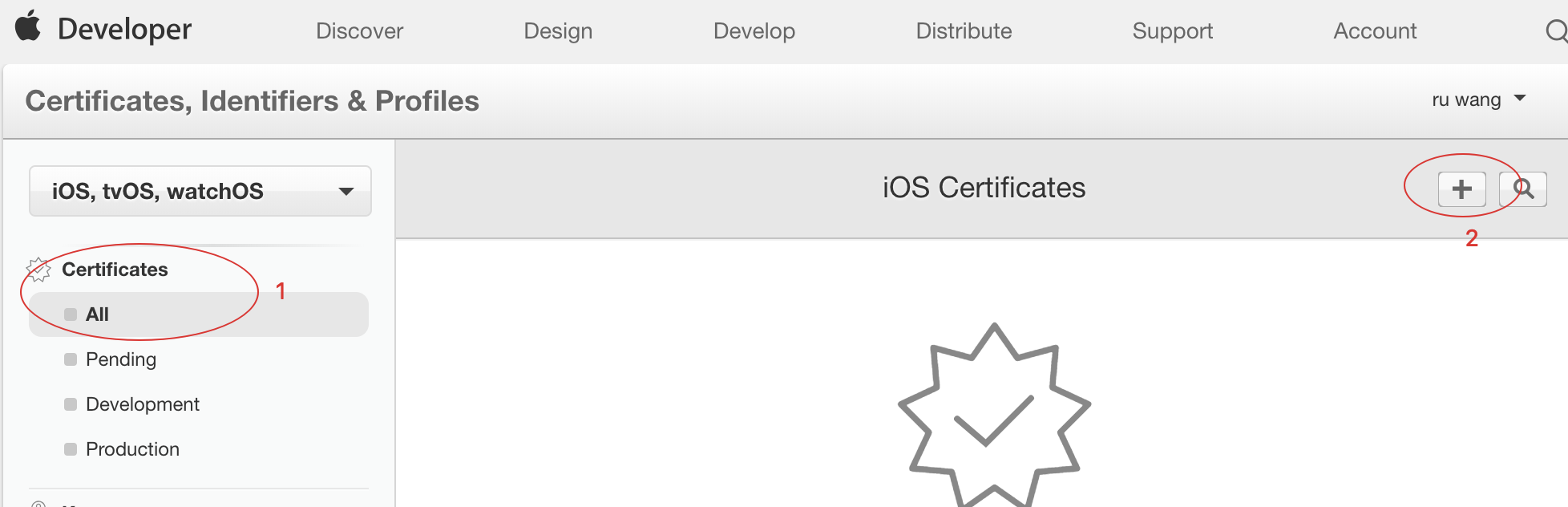


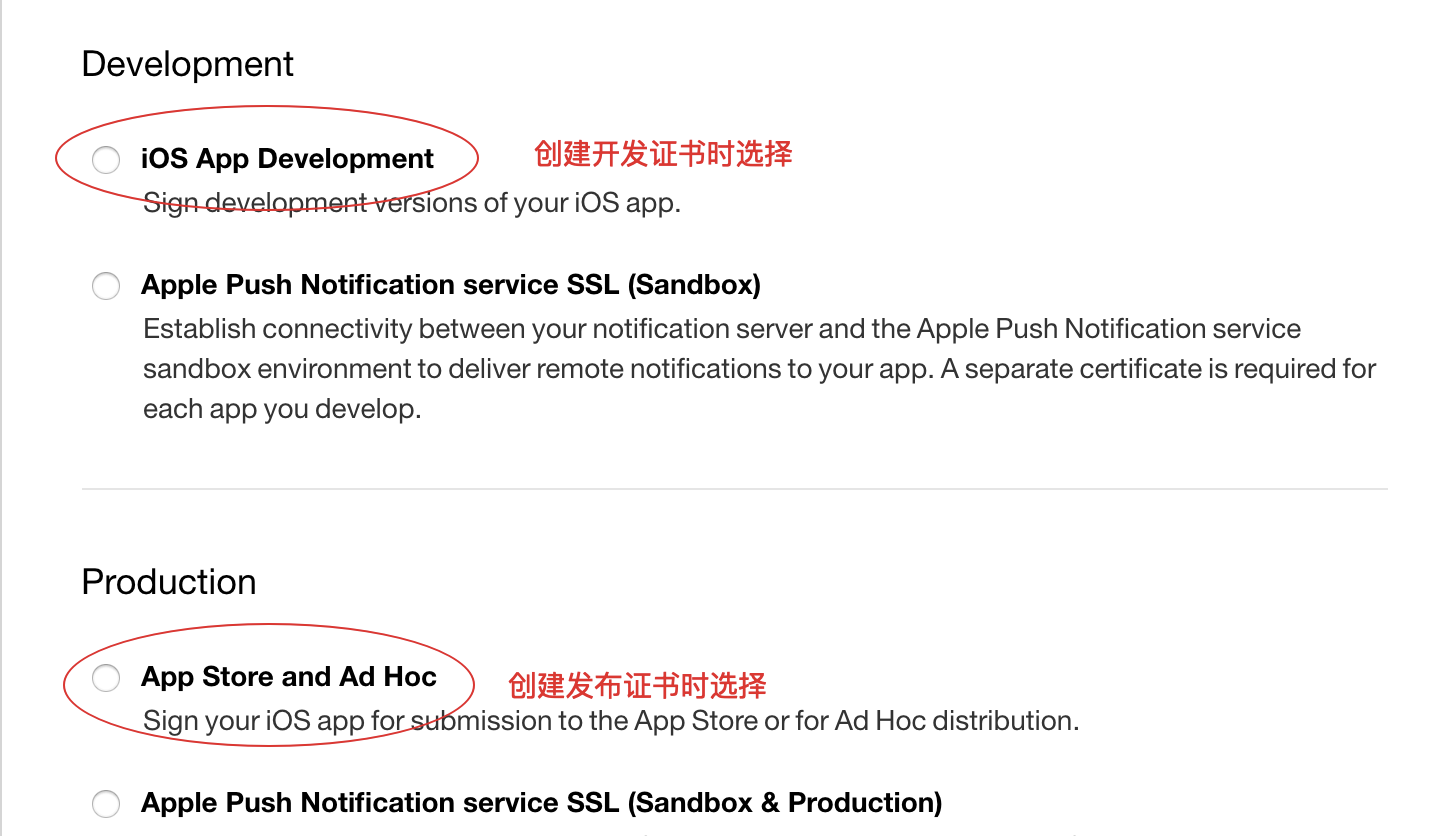


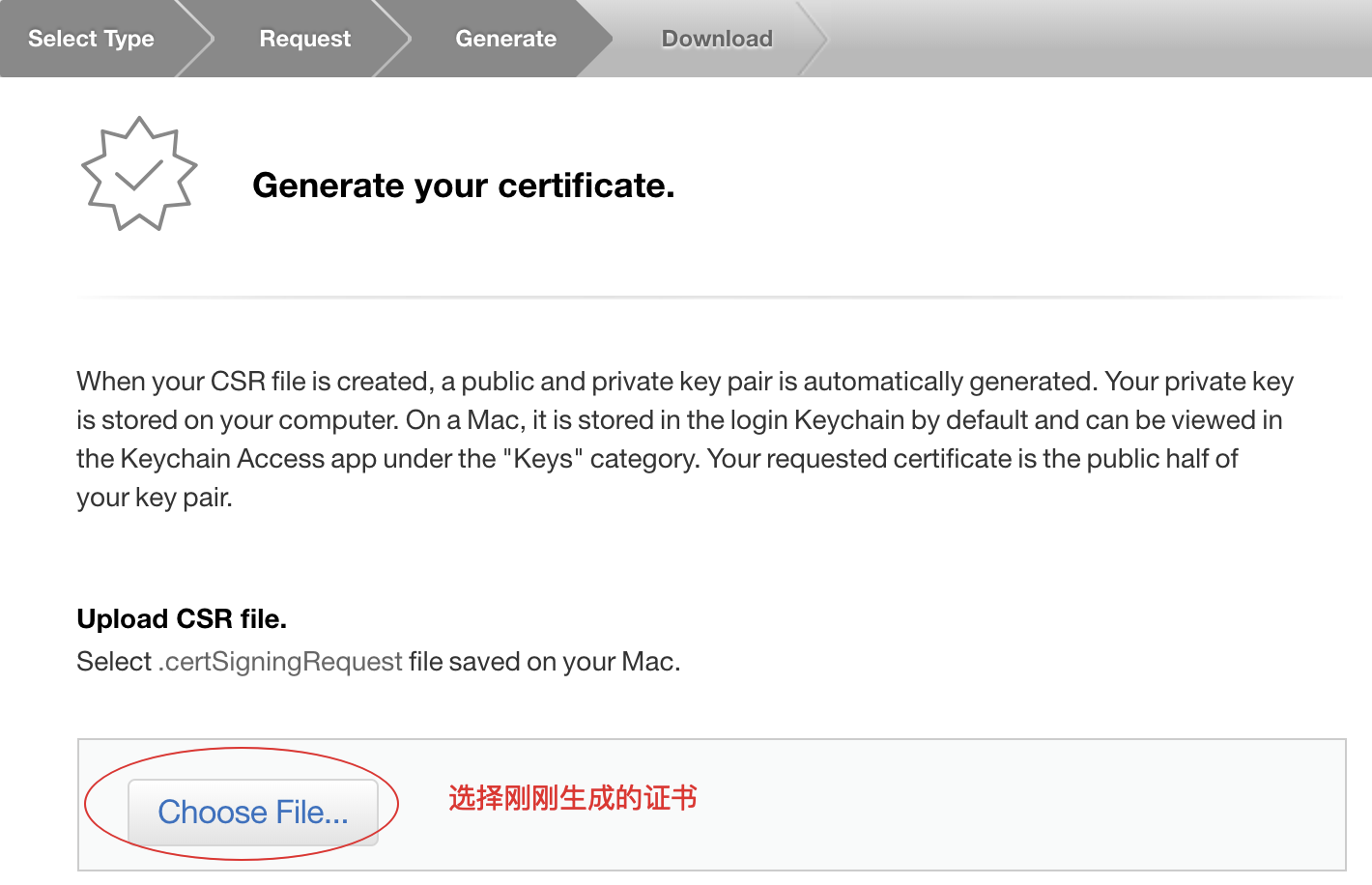
用户电子邮件地址填入当前账号，选择存储到磁盘，点继续。选择一个存储位置存储（在桌面新建目录《游戏名称》来存储）



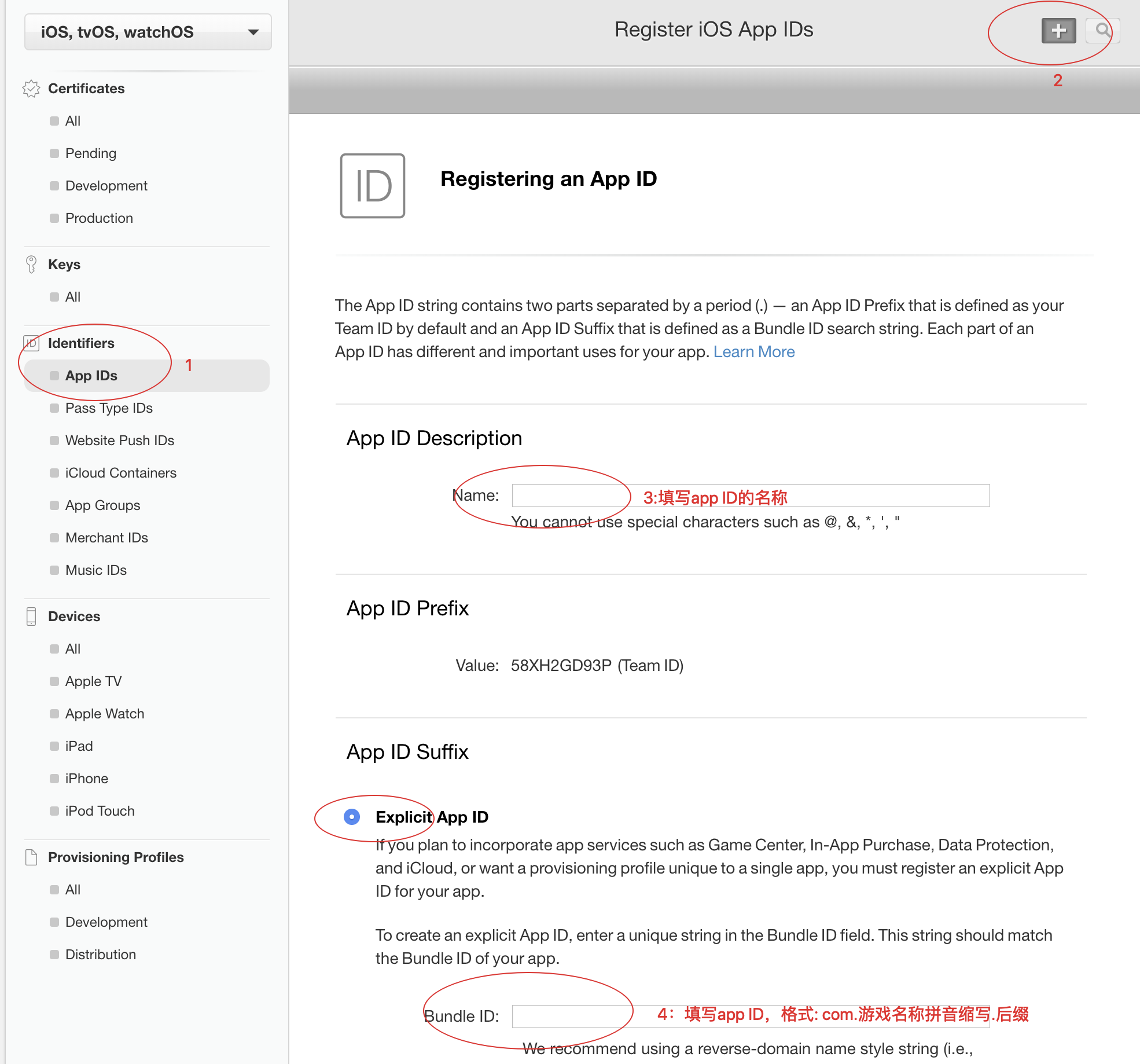
1. 在开发者后台生成发布证书、描述文件
2. 登录开发者后台：<https://developer.apple.com/account/ios/certificate/>
3. 创建发布证书：选择 **Certificates → ➕号 → App Store and Ad Hoc → Continue → Continue → 选择刚刚保存到本机的证书 → 点继续后下载保存到本地。**

****

****

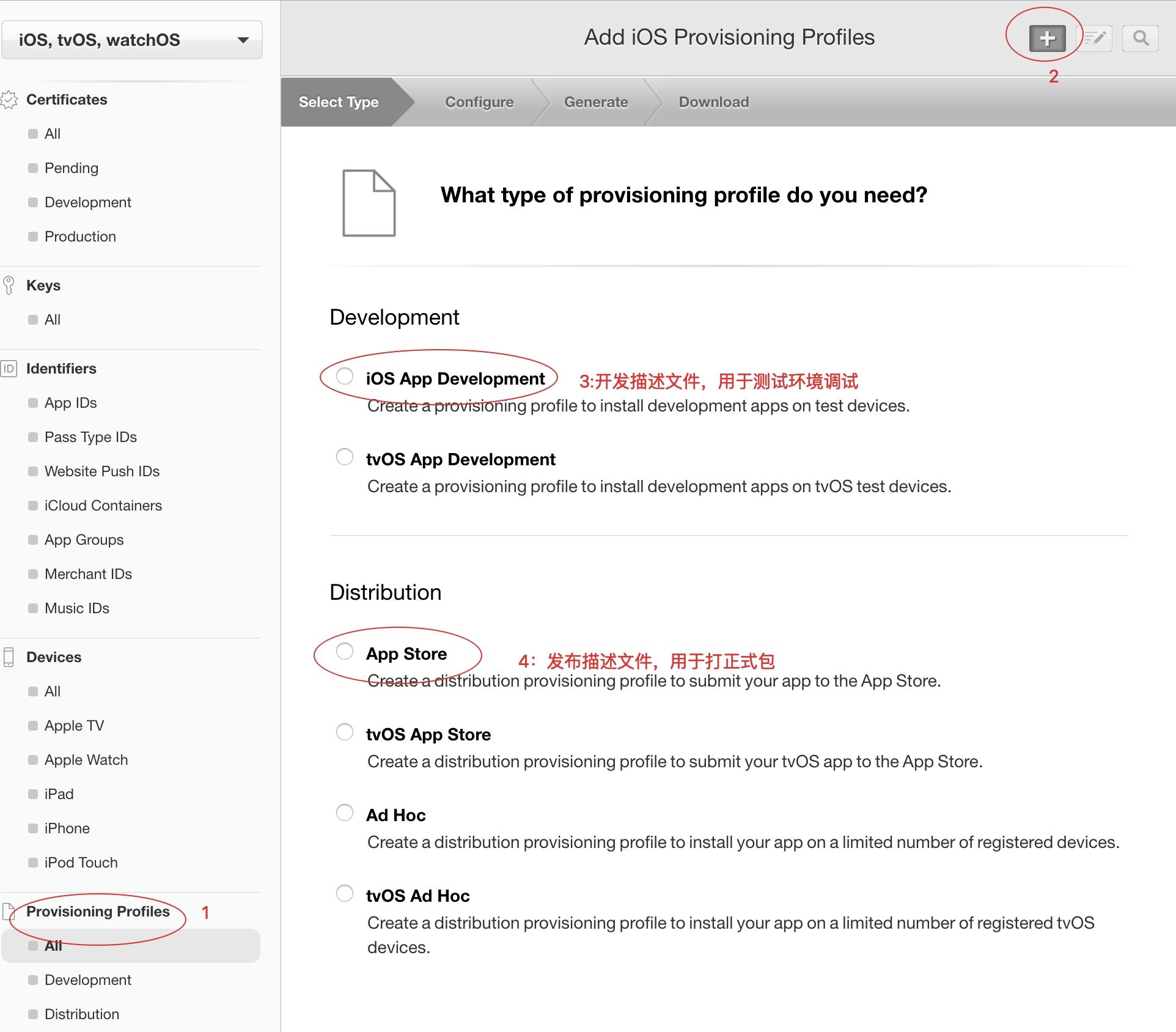
****

1. 生成App ID：选择 **Identifiers → App IDs → ➕ → 填写app ID昵称 → app ID → 继续 → 完成注册。**

****

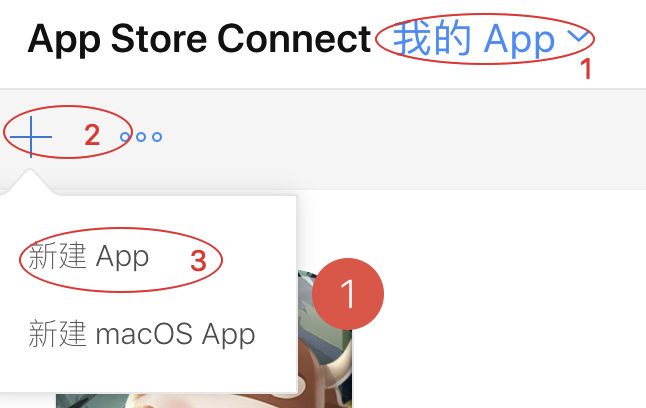
1. 生成开发描述文件、发布描述文件：

选择 **Provisioning Profiles → ➕ → 根据需要选择开发描述文件/发布描述文件 → 点继续后选择创建的app ID → 点继续后选择创建的证书，开发环境、正式环境一样对应 → 选择添加的设备ID → 完成注册后下载保存到本机。后续使用时需要对应安装开发证书-开发描述文件，发布证书-发布描述文件（双击即可安装）**

****

第四章 填写iTC信息、打包、上传App connect

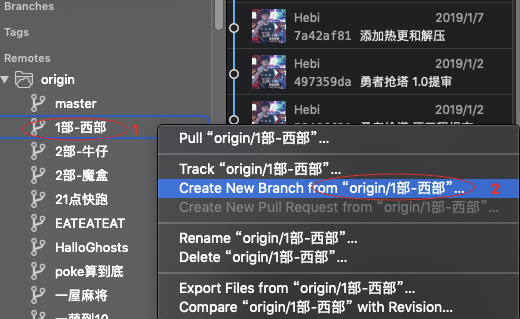
1. 登录iTC后台，填写对应信息
2. 登录 <https://appstoreconnect.apple.com/login>
3. 选择 我的App → ➕ → 新建App → 按照提示填写对应信息后点击创建
4. 根据提示填写完对应信息即可
5. 有内购的游戏需要填写内购所需信息

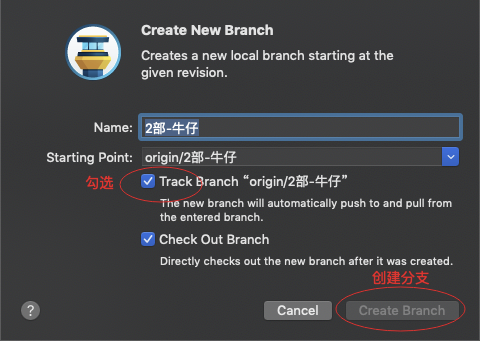


1. 打包
2. 在准备的打包电脑上安装生成的发布证书、发布描述文件（双击即可），证书选择使用手动方式，选择安装的描述文件，确认证书、描述文件没有问题后打包并导出ipa
3. 上传
4. 打开Application Loader登录，密码需要登录账号管理后台生成专用密钥<https://appleid.apple.com/#!&page=signin>
5. 选择导出的IPA上传即可

第五章 添加游戏压缩包、热更新

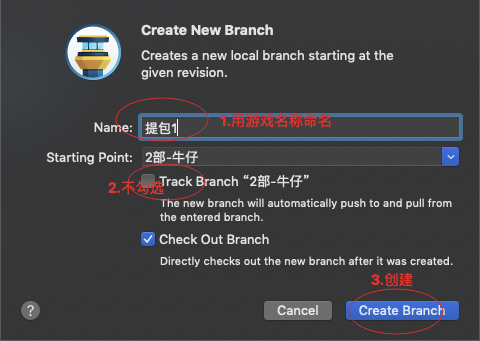
1.在提包git库拉取需要添加的平台分支到本地，找到该库的对应目录备份到本地（例：拷贝game-client-pack目录的内容粘贴到Desktop/xibu\_backup/）



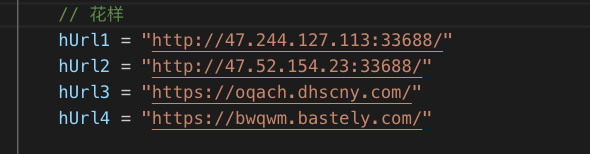


2.在库中找对应游戏分支。如果未找到，则根据本地平台分支创建一个游戏分支，然后打开对应目录，删除掉所有内容，拷贝游戏的内容到该目录；如果已有游戏分支，则直接切换到对应游戏分支。

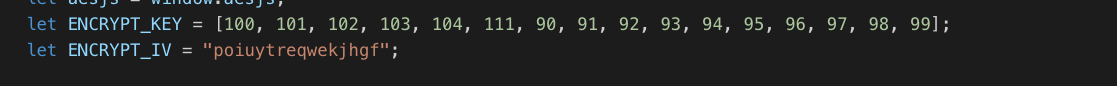
从备份的xibu\_backup中拷贝aes.d.ts、creator.d.ts、jsb.d.ts、md5.d.ts到游戏根目录，将xibu\_backup/packages、xibu\_backup/build、xibu\_backup/assets目录拷贝到游戏对应目录。



注 ：删除分支工程将小游戏工程替换进来，打开小游戏运行是否有错误，如果有错误修复完错误后，再进行后续操作。

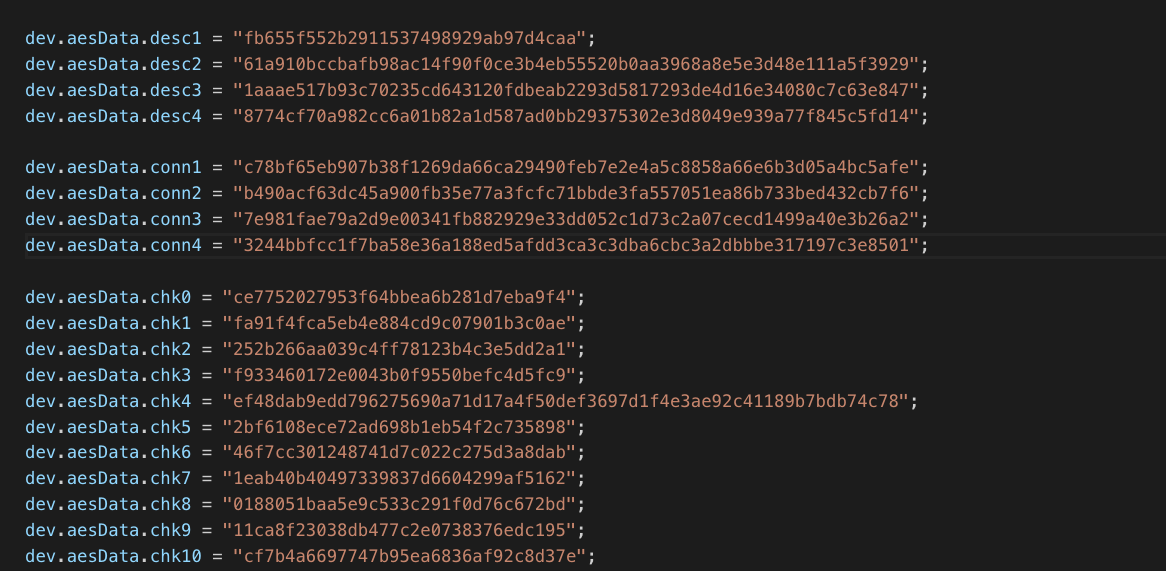


查看ip与域名是否正确， 将ip设为1，2，域名设为3，4



打开tools.ts找到ENCRYPT\_KEY，ENCRYPT\_IV，随机挑选5-10个数字及字符进行替换；保存后运行工程，在控制台查看输出内容，拷贝对应的MD5输出到代码中



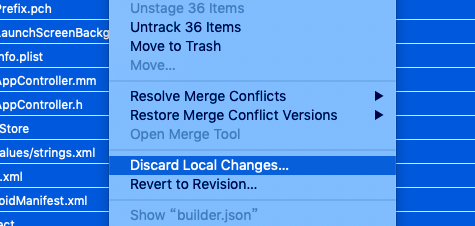


替换到相对位置，完成后注释掉begin.ts里的输出日志

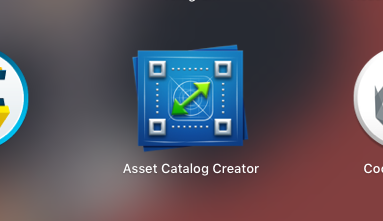
构建小游戏，选择ios平台，设置构建路径为build，设置模版为default，勾选加密脚本，设置密钥，选择zip压缩



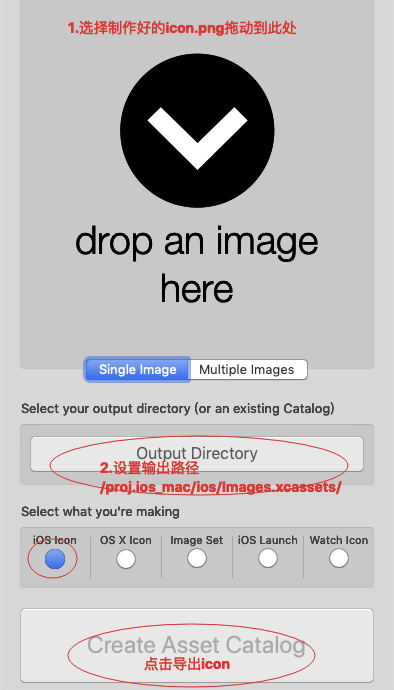
构建完成后，打开分支会发现build目录下的部分文件被移除，选择build/jsb-default/frameworks/runtime-src/proj.ios\_mac/、build/jsb-default/frameworks/cocos2d-x/下的所有文件还原

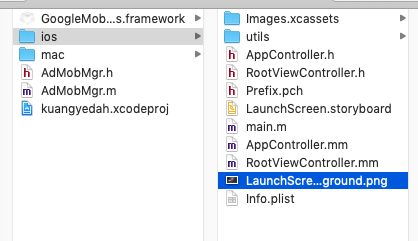


打开工程重新构建。

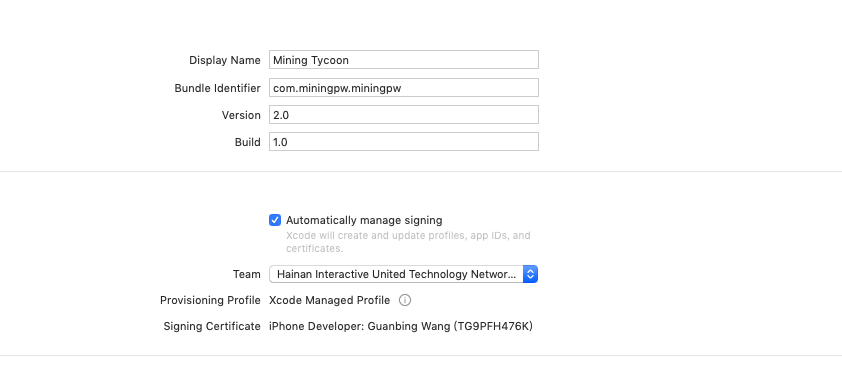


打开asset catalog creator





将启动图命名为LaunchScreenBackground.png替换掉默认到启动图

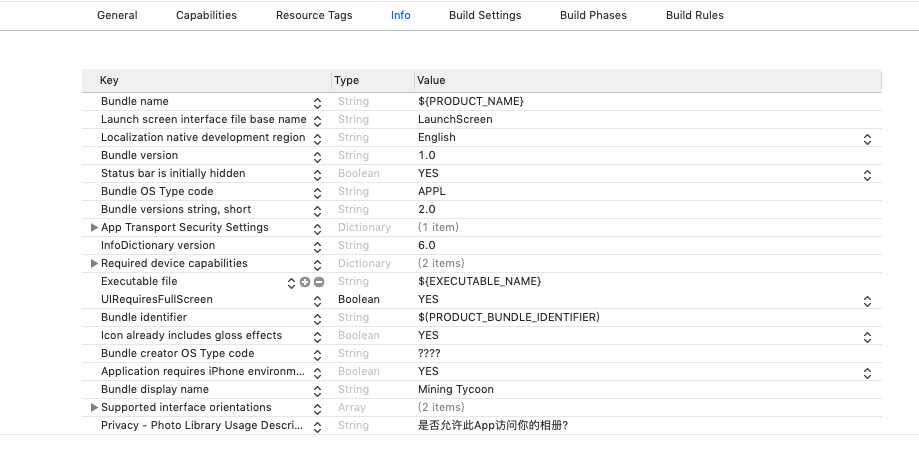


打开xcode，依次填写Display Name，Bundle Identifier，Version，Build

开发调试阶段勾选自动生成证书**Automatically，**选择模拟器或真机调试；打包发布阶段去掉自动管理证书，必须使用手动生成证书，只需要配置release发布证书即可



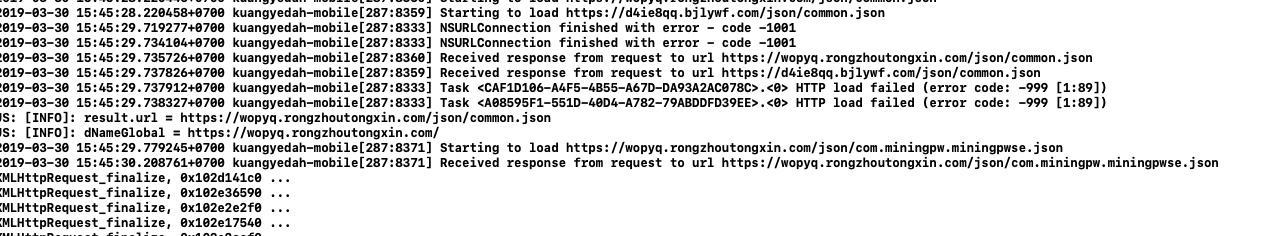
设置deployment target版本为10.0



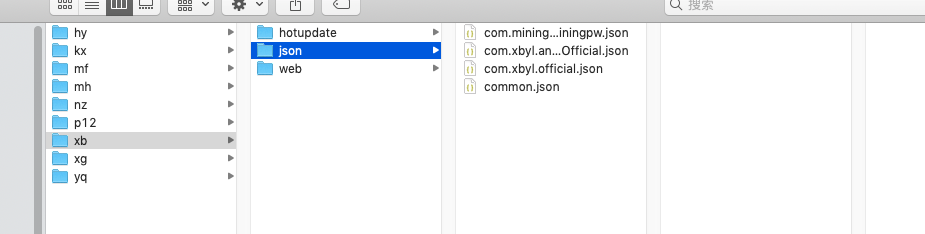
查看info是否添加访问相册到权限

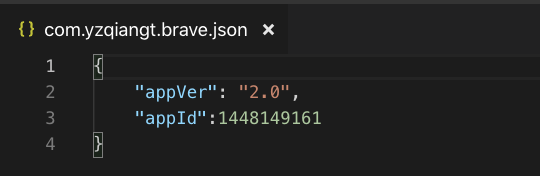
所有配置信息填写完成了，Version设为1.0，在模拟器运行检测是否进入原游戏。成功后开始测试热更新、解压缩逻辑

如果卡在启动节目，且日志输入如下



是因为没有添加json文件

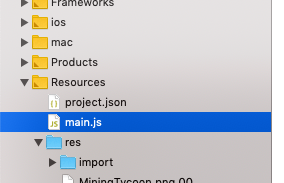


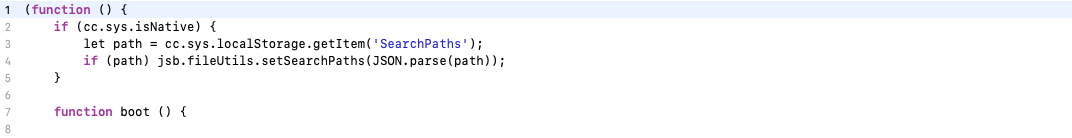
选择平台下的json文件夹，创建一个新的json文件，

文件名称，是bundie, appver是版本号，appid是创建应用的id

创建完成后上传git，并告诉运维，让他们上传到服务器中

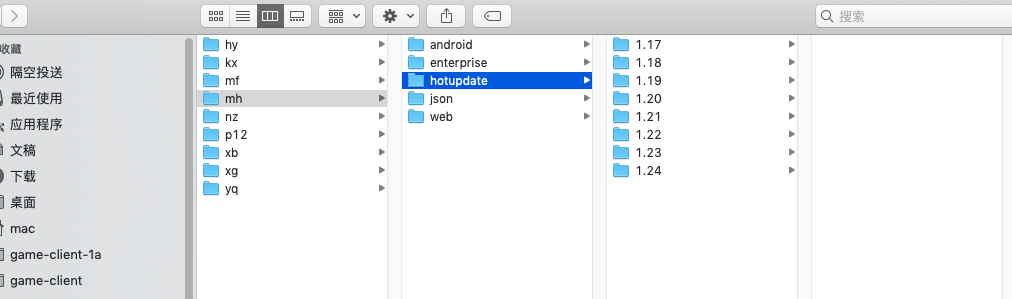
如果出现更新完成后，没有重启，继续更新的情况



查看main.js有没有添加

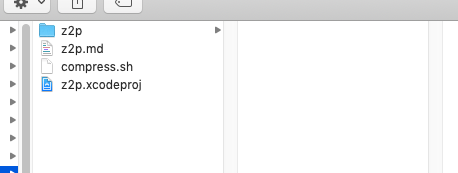
这两行代码，没有加上，从新打包测试，

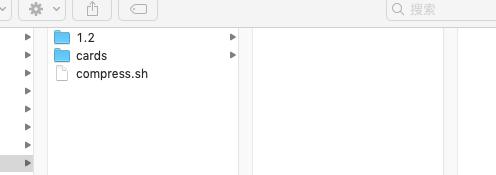
注 : 完成以上所有步骤，且没有问题的情况下，再进行后面的操作，

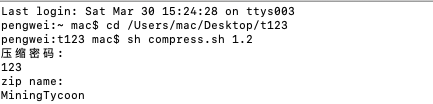


拷贝分支中最新到更新资源到桌面，



创建一个t123的文件夹，并将更新资源拖进去，

将压缩工程下的compress.sh复制一份到t123文件夹中

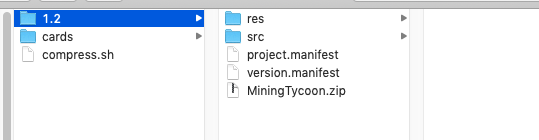


在终端打开t123路径

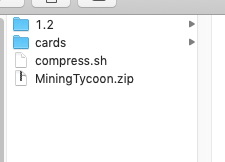
执行.sh文件，后面跟上资源文件夹名称

输入压缩密码，和zip名称

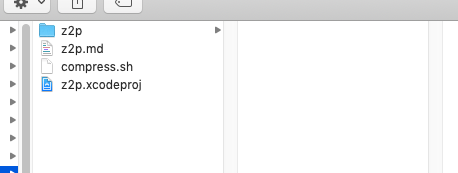


zip名称必须与脚本设置到名称相同，

执行完成后在资源文件夹下，会生成zip压缩包，



将zip拖到同级目录中，



打开z2p， xcode工程，

设置路径，第一个是导出到png压缩包路径，

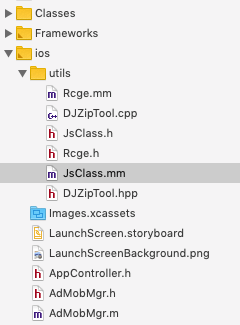
第二个是获取zip压缩包路径，



导出的png名称要与脚本设置的相同，

运行xcode，

将生成的*byte*



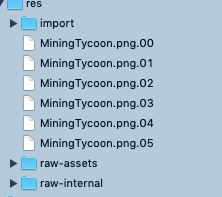
替换到jsclass脚本的



mergefilepath方法中的相对位置，



查看导出的png压缩包名称是否与设置中的一样，



将压缩包，拖到xcode工程中，位置

要与脚本设置的相同，

设置解压密码，在jsclass脚本中的decpwd变量

运行测试解压

如果没有执行资源解压中,此过程不消耗流量...直接进行了更新，查看输出日志



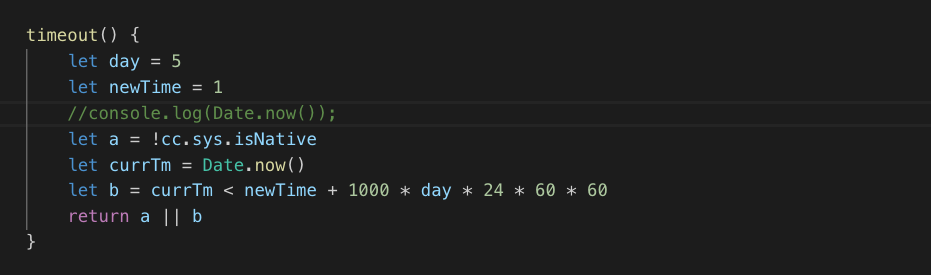
zip isfileexist 是否为 true，

不等于true检查路径，压缩包名称是否与脚本设置的相同，

等与true

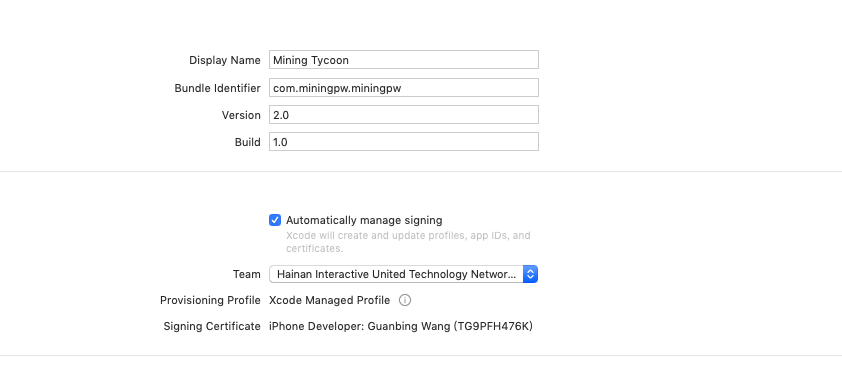
检查jsclass脚本mergefilepath方法序号后拼接出的名字与压缩名字相同，

注：以上步骤没有问题再进行以下操作，



在begin.ts,中将timeout的newtime的临时变量设置为当前时间戳，第一次提交小游戏。可以将day设置为5，后续更新设置为3

重新构建，检查选项



打开构建工程，检查填写的资料

没有问题后提交ios审核包