# 提包文档手动篇

## 工作流程：

提包也就是把ios应用包提交到苹果应用商店，供用户去下载，我们现在提包的总体流程为：**原始小游戏 => 交给美术做换皮资源 => 把原始游戏利用对应的换皮资源做换皮 => 打包ipa =>提审 app store => 热更新 => 上架**

## 准备工作-了解提包进度表-总：

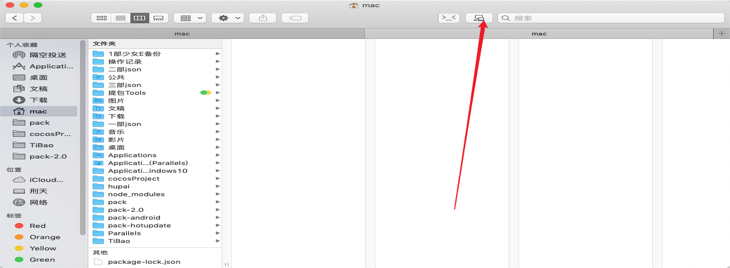
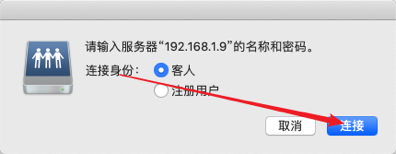
**提包进度表-总**是非常重要使用频率非常高的表格，用了查看自己提包的信息，下面进行举例说明：



* 包名：换皮结束后，需要把根据换皮结束后的游戏内容为游戏起一个名字和游戏的描述，游戏描述要和游戏内容相关联，起好之后发送名字和描述到**提包-技术**群中，如没有问题把游戏名字填写到包名处，有问题进行更改后填写到包名处
* 技术支持网址：根据游戏名称、描述、美宣等信息制作技术支持网址
* 平台：过包后，要加热更新的平台，目前分为：一部、二部、测试部
* 苹果账号：进行提包使用的账号
* 苹果密码：进行提包使用账号的密码
* 邮箱密码：提包后需要进行绑定邮箱到测试机上面，以便接收审核通知
* AppID：填写提包完成后的appID，后面具体说明
* 套装ID：为该游戏起一个唯一的bundleID, 填写到此处格式一般为：com.xxx.xxx, (xxx可以使用任意字符代替)
* n森林partym: 换皮资源名，去**smb://192.168.1.9**服务器里面的**提包中心/换皮资源/第n批**中去找到对应的资源
* 备注：需要注意的内容及需求
* 云主机：打勾的代表使用云主机去提包，不打勾代表使用vpn提包
* 198.11.178.203：如果是云主机提包此处为云主机的ip，否则为VPN的地区
* Mini设备：这里是提包的设备名称
* Mini密码：这里是提包账号锁绑定的电话号码

## 准备工作-1.9服务器和git：

常用的远端存储一般为1.9服务器和git，下面分别进行详解：

* 1.9服务器：为本地局域网服务器，ip地主为：**smb://192.168.1.9,** 用来存放提包常用的工具、资源、文档等，如连接方法如下图所示：  
* Git：git库，所使用的客户端工具为tower。用来存放热更新包、原始游戏包、已过游戏包、json数据等，常用操作为：拉取、上传、还原、拷贝远程分支到本地，详见谷歌知识库中**Tower的使用文档**

## 换皮：

查看提包进度表-总，找到需要进行换皮的游戏及资源，资源在1.9服务器上面，游戏在git上面，参考资源中效果图进行换皮，如缺少资源，向**提包中心-工作群**中找美术大佬进行反馈索要缺失的资源，换皮结束后需要对换皮后的项目进行处理方便过审，目前对换皮项目的处理如下

* 换皮之前把资源全部拖进一个文件夹中，在按照对应的资源的位置进行换皮（这样的是为了生成新的.meta文件）
* 在git上面拉取最新的操作记录添加到项目中去，具体方法：
  + 去1.9git上面拉取最新的操作记录到本地并备份
  + 切换回提包项目分支把操作记录项目中assets/caozuojilu文件夹直接拖到项目中
  + 把caozuojilu/start/start.ts直接拖到层级管理器中
  + 在项目的所有可能进行的操作加上操作记录：
    - 文件头部引用 import {printLog} from “./caozuojilu/common/uyil”
    - 需要加操作记录的位置：printLog(“记操作记录内容”)
* 换皮结束后要改变目录的结构，常见操作如下：
  + 改变资源和文件夹的名称
  + 图片资源目录添加自动图集并设置为webp格式
  + 直接在资源父目录添加一张大的自动图片并设置为webp格式，这样子文件中所有图片都会打进图集中去，并且不会有子文件夹
  + 音乐等资源要改变名称
  + resources文件尽量不存放资源，如果有尽量改进。如进入代码改变资源名称或打webp图集
* 检查项目的packages目录或全局packages目录(~/.CocosCreator/packages)中存不存在hot-update-tools插件并且插件文件夹内部有md5.sh文件
* 运行changeuuid.py来改变项目资源的uuid。使用方法如下：
  + 关闭项目并提交本地git
  + 先进行构建一次
  + 打开终端运行 python changeuuid.py 项目的assets目录
  + 打开项目运行观察有没有报错或资源引用丢失，如果有进行处理
* 检查CocosCreator/偏好设置/原生开发环境是否选择了最新的自定义引擎底层
* 以上都处理且无任何报错后进行构建。打开构建面板，进行如下图的设置：



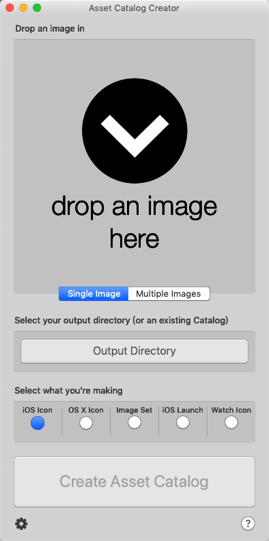
* + 游戏名称填写游戏的名称不用写中文，**必须要进行手动修改，禁止使用默认的pack**
  + 初始场景选择第一个场景
  + **加密密钥1.0的时候必须修改**。可以手动更改也可删除settings/ builder.json后重新打开cocos creator进行自动更改。
  + 对照上面检查一切无误后进行点击构建按钮进行构建。
  + 在构建过程结束会自动运行hot-update-tools插件执行md5.sh来修改所有资源的md5值，观察构建后的结果：
    - 如提示shelljs不存在，则打开终端运行：npm install shelljs后重新构建
      * 如提示npm不存在，则打开终端运行：brew install node
      * 安装成功后重新运行npm install shelljs后重新构建
    - 如提示packages://hot-update-tools/md5.sh--- 不存在 ---检查目或全局packages/hot-update-tools/md5.sh中是否存在，如全局存在项目不存在可以把全局hot-update-tools拷贝到项目中
    - 成功会提示[热更新] 合并'main.js'文件完成并且---- stdout: 处理json文件展开会有处理资源的结果
  + 构建结束后进行修改自定义引擎的类名，使用方法如下：
    - 在项目/build/jsb-default/frameworks/cocos2d-x下面打开终端并运行 python ModifyClass.py 脚本。成功后会自动修改类名及更新Xcode引用。会提示Generating javascript bindings succeeds.
    - 如失败查看/build/jsb-default/frameworks/cocos2d-x/README.md进行修复。修复后删除build后重新进行构建
  + 打开Xcode项目，路径为：/build/jsb-default/frameworks/runtime-src/proj.ios\_mac/\*.xcodeproj
  + 运行项目检查是否报错，如无报错可进行下一步

## 提审：

提审到app store 商店，无论是vpn方式还是云主机的方式，大体流程都可总结为：

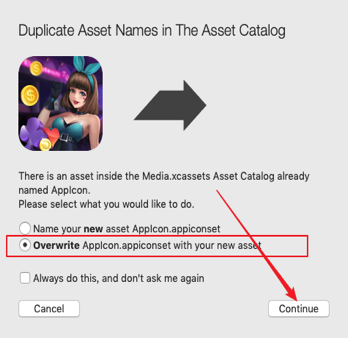
**更换icon及启动图等=>制作证书、描述文件及bundleID=>项目构建=>导出ipa包=>上传至app store => 添写元数据信息 =>提交审核=>提包进度表更新**，下面进行分别讲解。

* 更换icon及启动图：目前有两种方法，现在来分别介绍一下
  + 使用mac内置工具手动替换：
    - Icon: 使用Asset Catalog Creator工具创建，见下图：

点击 drop an image here 选择icon，或直接拖拽此处，

点击Output Directory选择输出路径，路径选择：“/build/jsb-default/frameworks/runtime-src/proj.ios\_mac/ios/Images.xcassets”

这个文件夹就是ios存放icon的路径，点击 Create Asset Catalog 生成icon



选择覆盖后点击继续成功成功icon

* + - 启动图片：打开/build/jsb-default/frameworks/runtime-src/proj.ios\_mac/ios/路径，找到名字为LaunchScreenBackground.png的图片就是启动图，找到合适的图片修改为相同的名字直接进行替换，但是这个可能会报错需要进行一下修复：

打开终端cd到项目工程目录下，执行以下命令：

rm -rf /Users/mac/Library/Developer/Xcode/DerivedData/\*

xattr -rc .

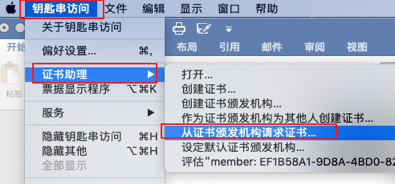
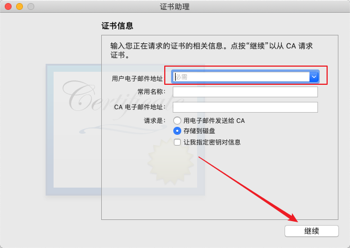
* + 使用tibaoTool.sh 进行更换, 使用 “sh tibaoTool.sh auto 项目根路径 icon的路径”下面进行举例：

下面会提示健康公告路径找不到，拖入一张启动图片即可，会自动打开Xcode

* + 制作美宣图片：先在游戏中每个界面分别截一张图并放到一个文件夹中，使用tibaoTool.sh工具，meix命令具体见下方图片：

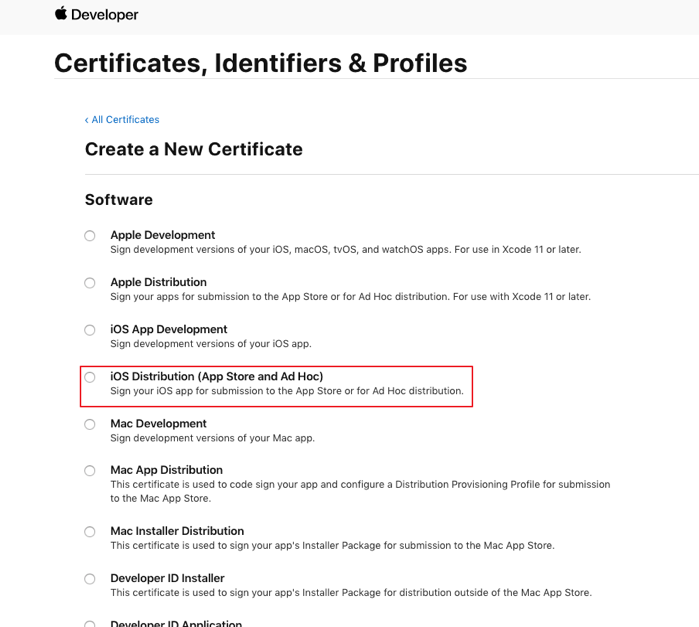


* 使用VPN制作证书及描述文件及导出p12：
  + 在mac上面打开钥匙串访问

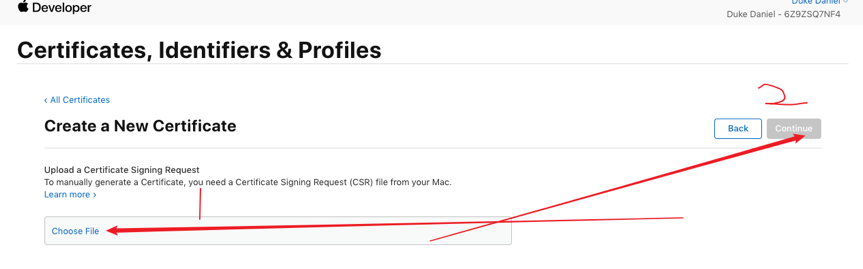
 

电子邮件处填写提包的账号，然后继续，选择一个位置进行存储，这样就创建了一个.certSigningRequest文件，这个是证书请求文件，是mac提包创建证书必须用到的，会使用这个文件创建证书文件

* + 打开<https://developer.apple.com/account/resources/certificates/list>, 然后点击加号

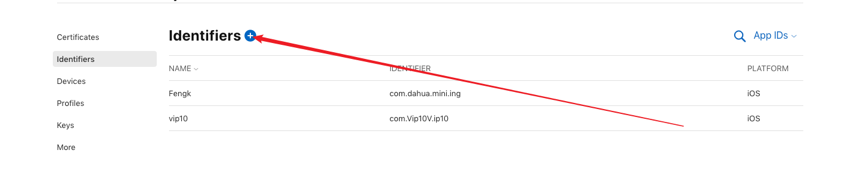


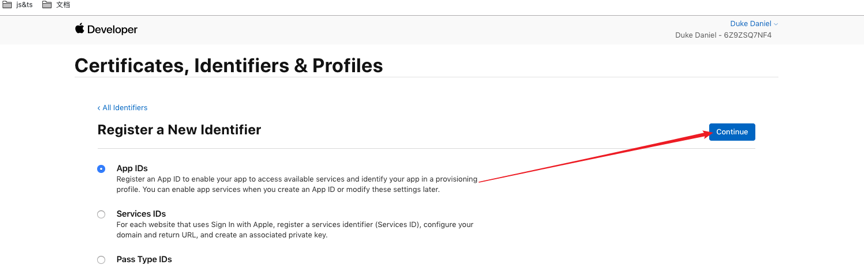
选择图中的这里进行创建证书

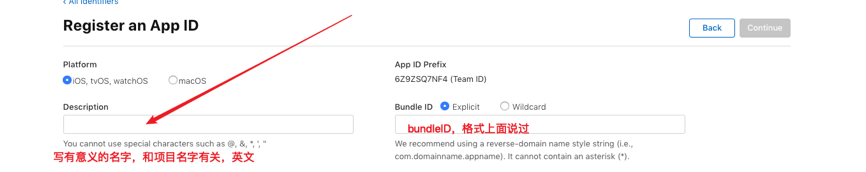


选择更改创建好的.certSigningRequest文件，点击继续，然后下载，会默认下载到Mac的下载目录中，请自行重新选择一个目录进行存储，

* + 创建bundleID：看图

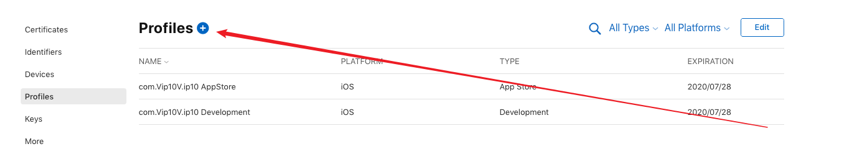


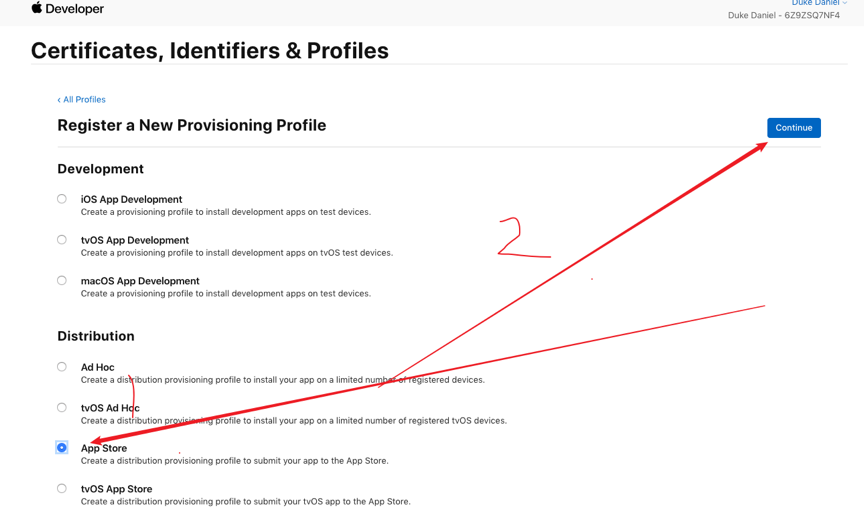


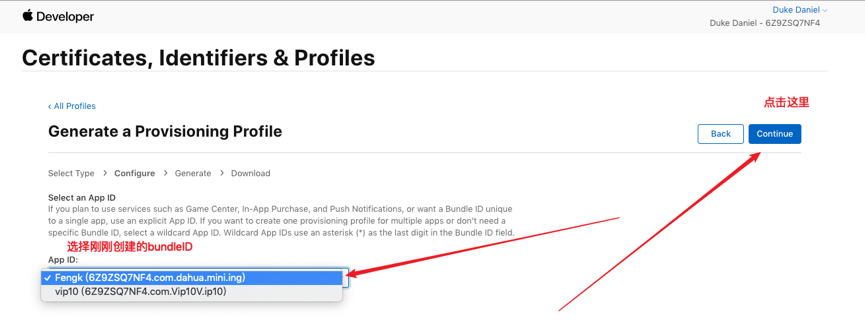


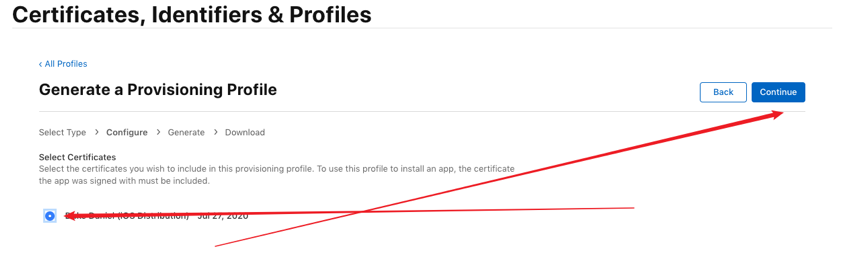
填写好后点击继续，创建完毕

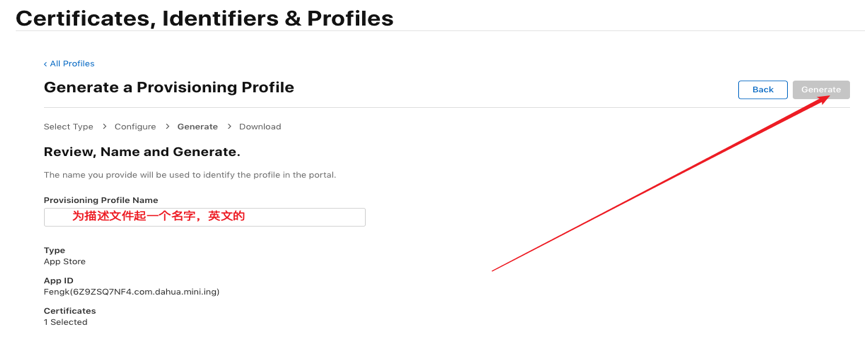
* + 描述文件：看图：





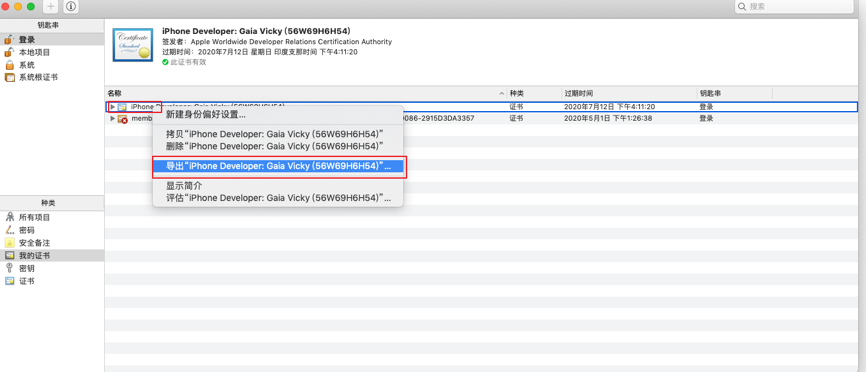




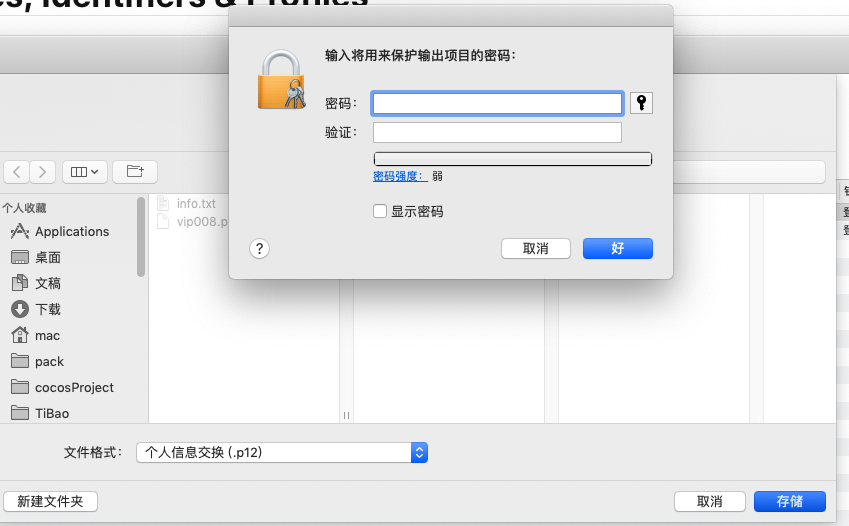


生成后下载，和证书放到一个文件夹中

* + 安装证书：打开装有证书和描述文件的文件夹，挨个双击进行安装
  + 导出p12文件：不导出在其他电脑证书和描述文件无法使用，同样打开钥匙串然后看图：

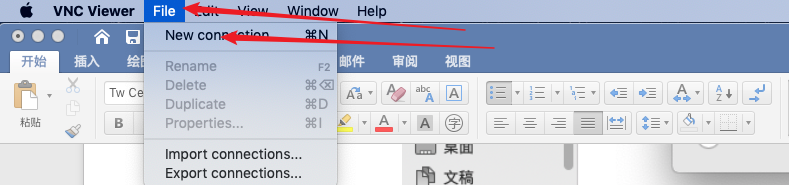


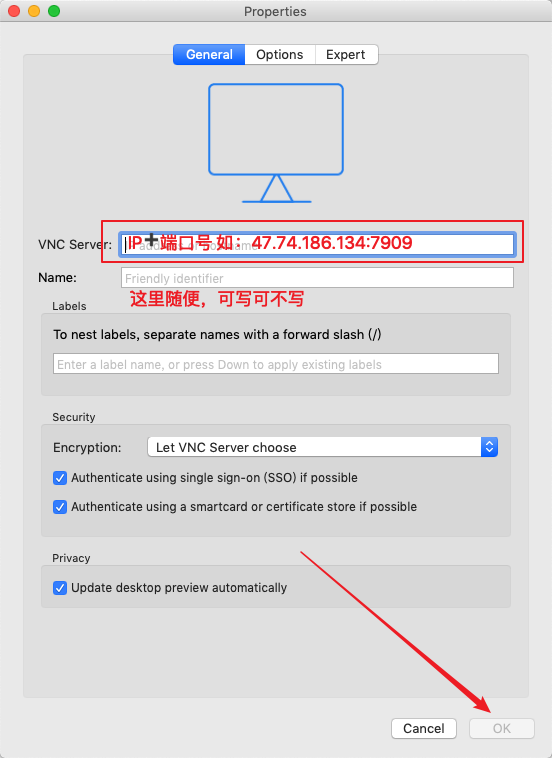
选择刚刚安装的证书，右键导出，选择和证书、描述文件同样的文件夹进行存储

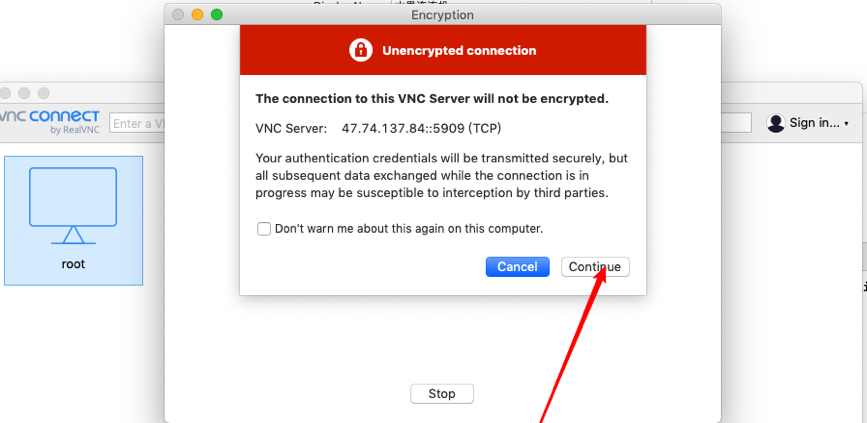


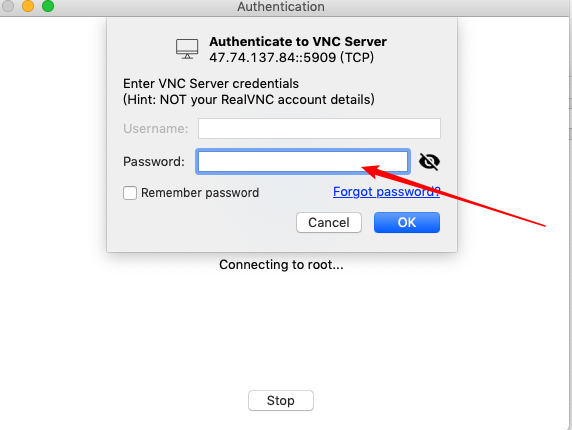
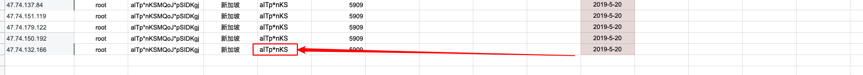
输入密码：123然后点击好，p12导出完毕.

* 使用云主机制作证书及描述文件及导出p12：
  + Linux云主机，vnc viewer简单使用方法
    - 去1.9服务器上面把所需工具下载到本地，1.9路径为：提包中心/提包Tools/ Linux云主机提包工具
    - 打开vnc viewer软件，创建一个新连接，有两种方法，自选：
      * 第一种方式，看图

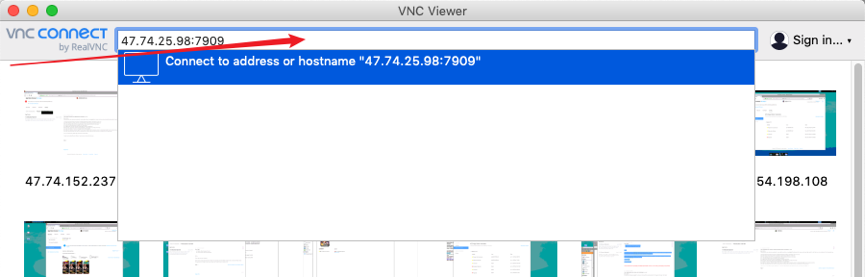






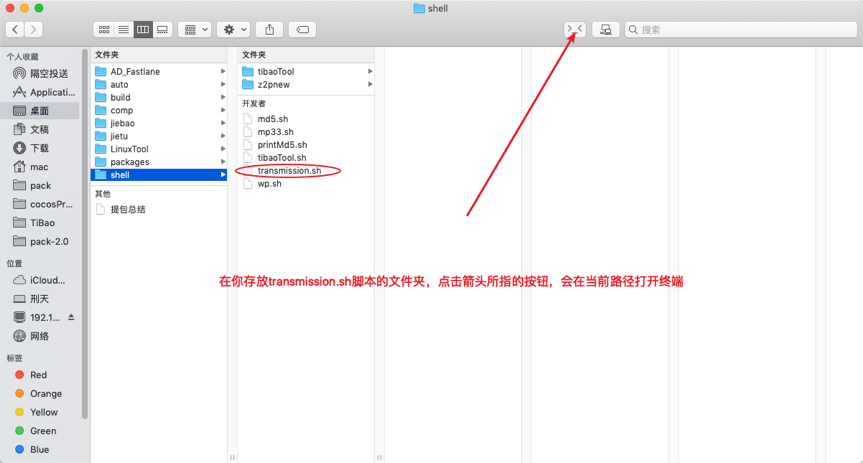
此处填写vnc的连接密码在表中寻找

* + - * 第二种方式，看图

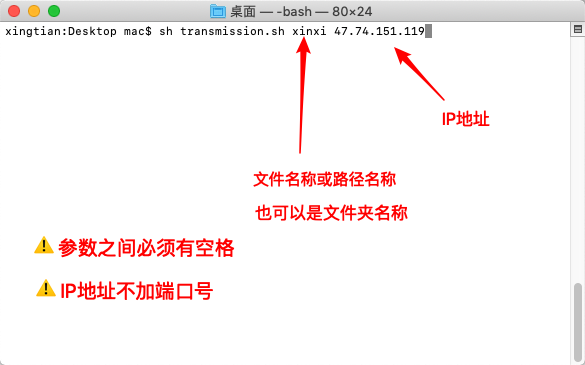


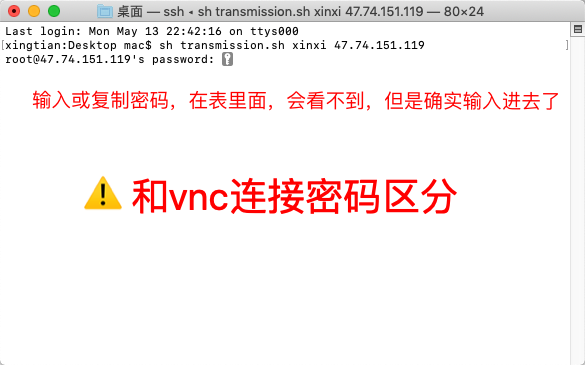
直接在搜索框内输入IP+端口号后回车，简单快捷，也可以使用这种方法搜索云主机

* + 准备文件：因为Linux云主机不支持中文，需要准备一些信息。把美宣图片、名称、描述、宣传文本、技术支持、隐私协议、appuploader等放到一个文件夹中准备传输。
  + 上传文件：目的是为了上传准备文件及打好的ipa包。使用transmission.sh脚本进行操作，transmission.sh 有两个参数，第一个是需要传输的文件， 第二个是ip地址。下面举例说明：



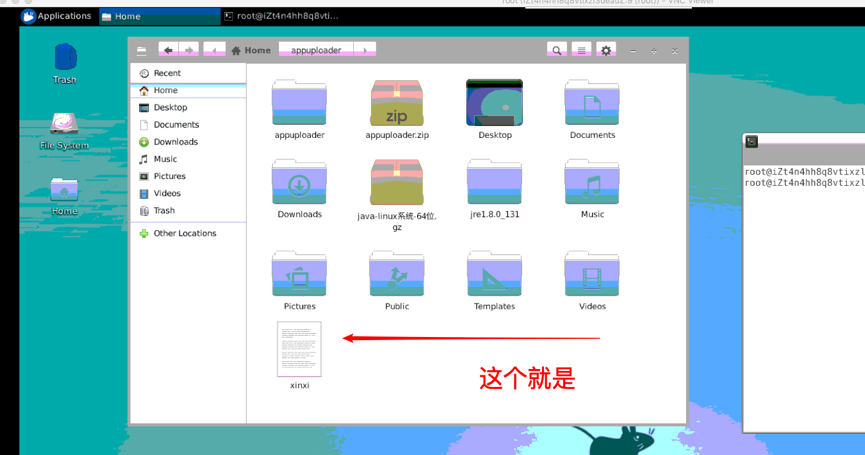








打开云主机确认一下



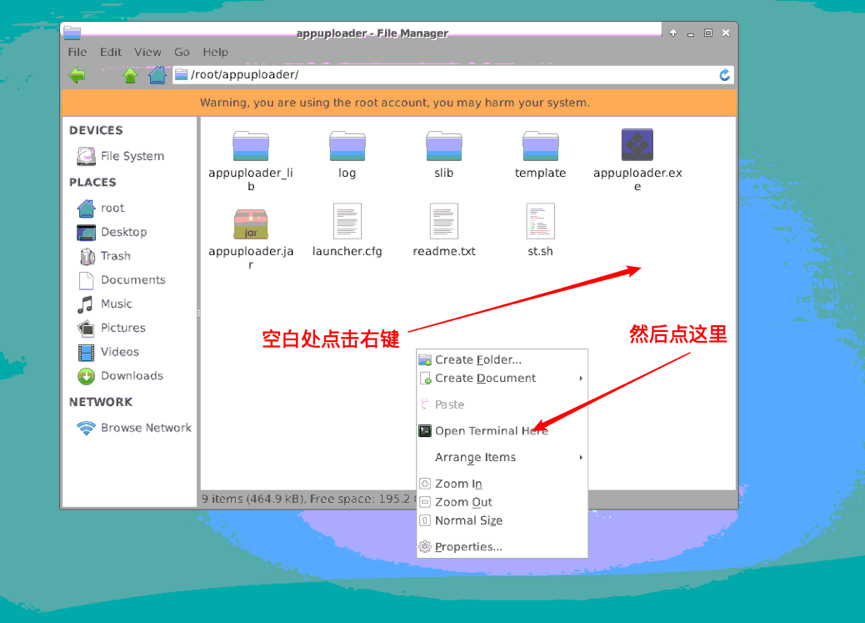
* + 下载文件：目的是为了下载制作好的证书



操作和上面基本相同，就是多了个参数 “-r”

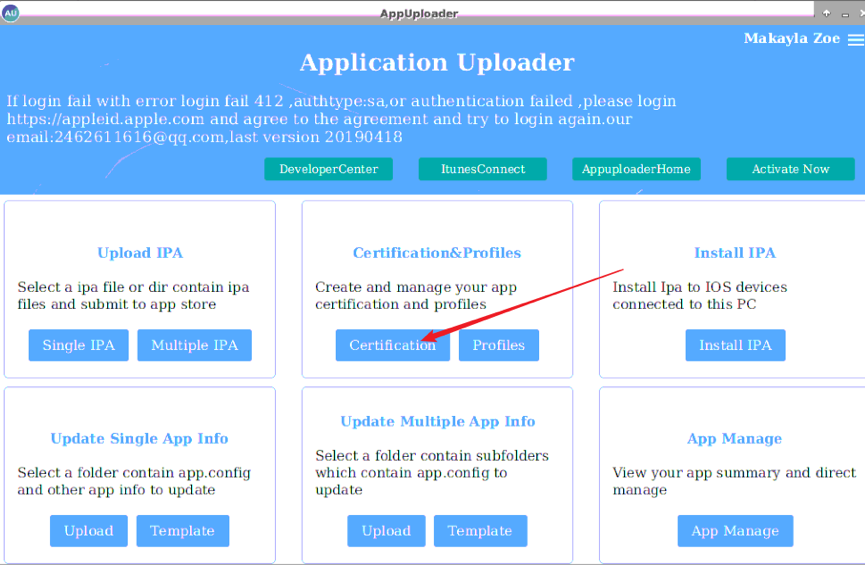
注意的是 这里上传或下载都是基于桌面上的 home目录

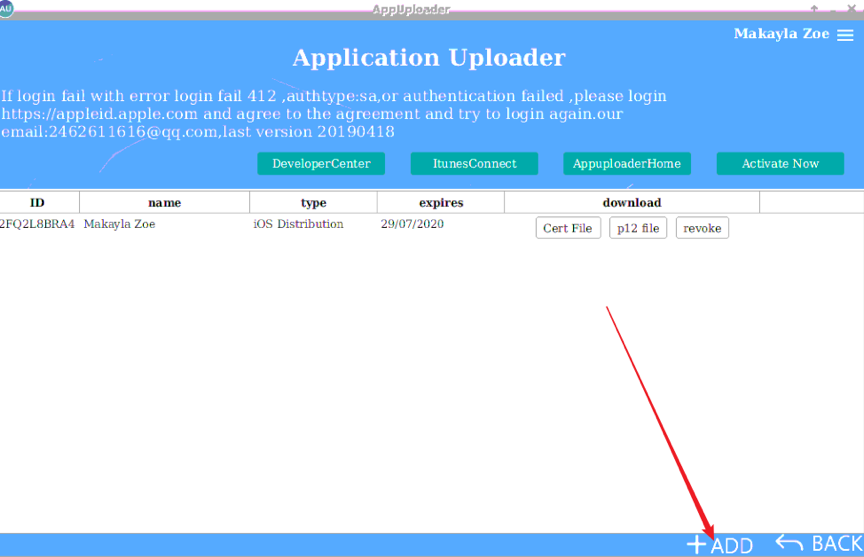
* + 打开软件：先进行解压 appuploader.zip 然后进入到appuploader 目录中去

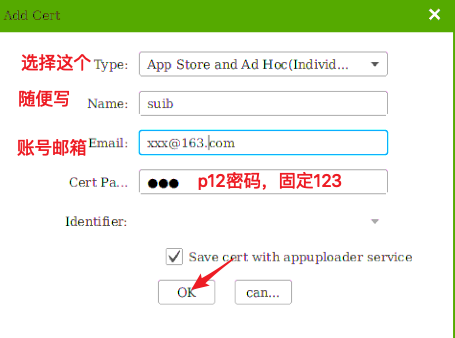


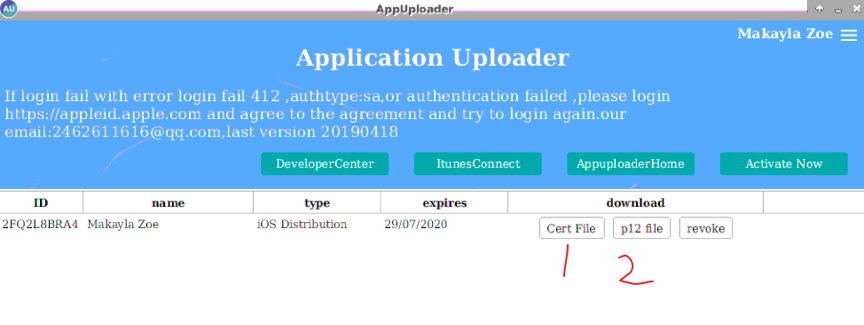
打开后进行登陆，这里需要注意的是在appuploader中输入账号密码验证码的时候不要使用小键盘会出错

* + 创建证书，看图：



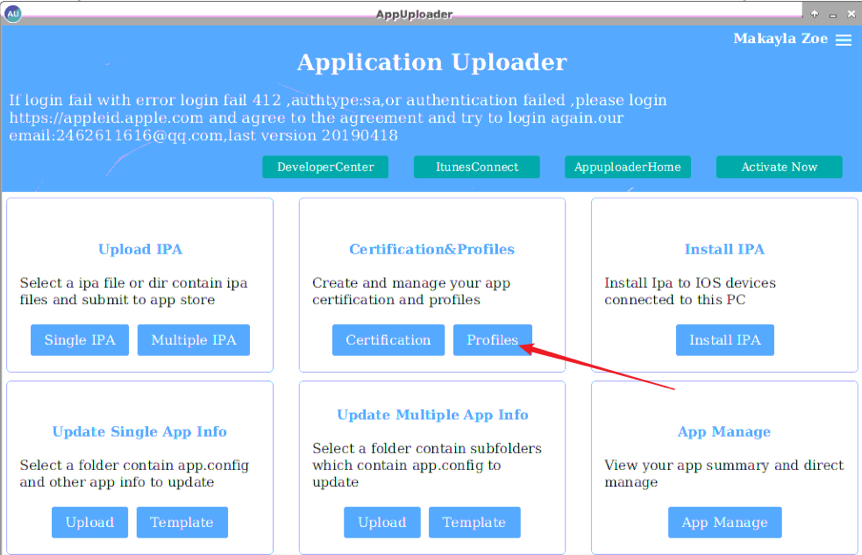


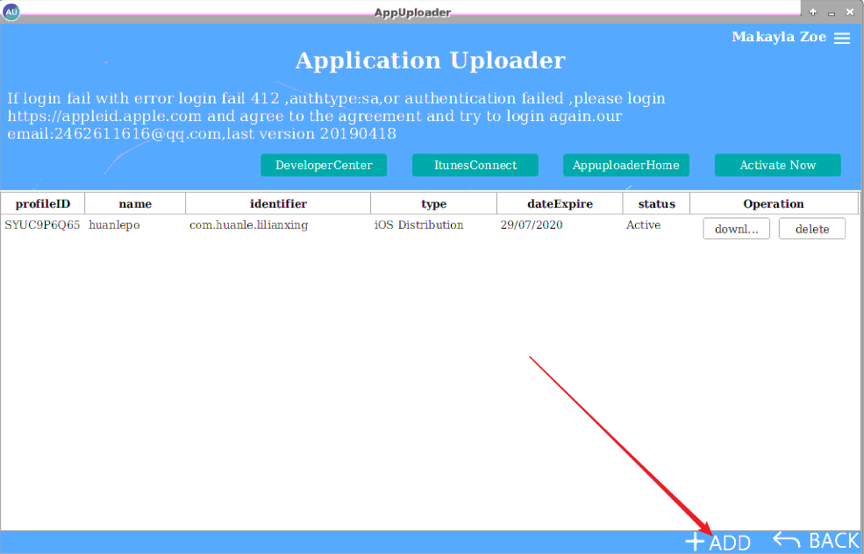


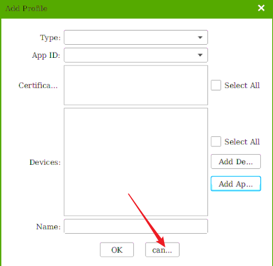
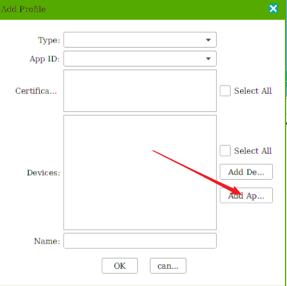


点击1和2分别下载证书和p12

* + 创建bundleID：

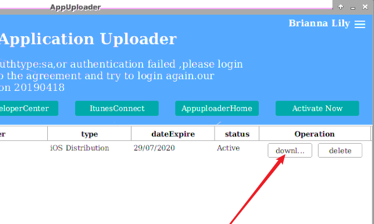






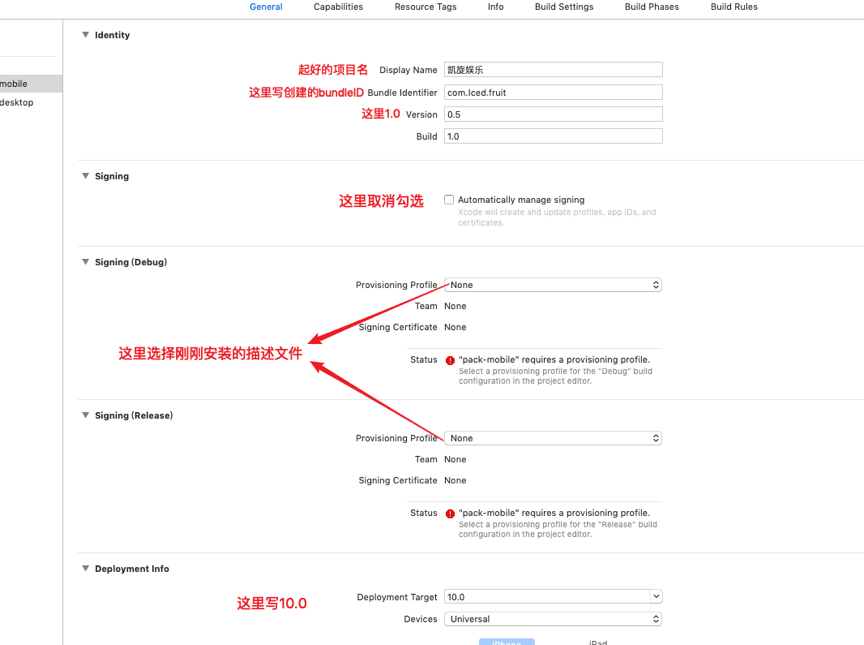
填写bundleID和名字，点击OK后返回

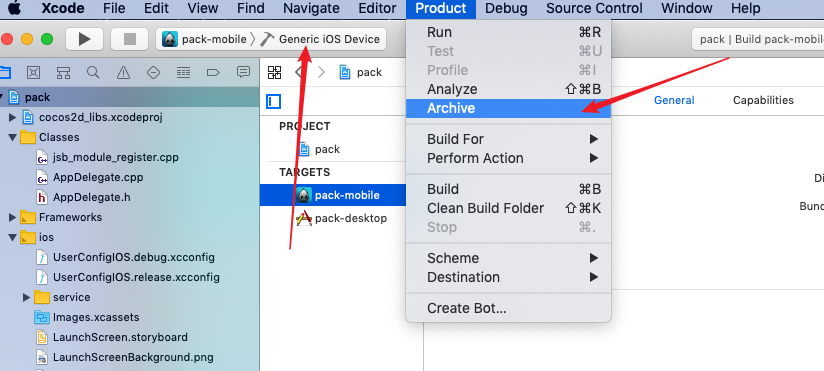
* + 创建描述文件：看图：



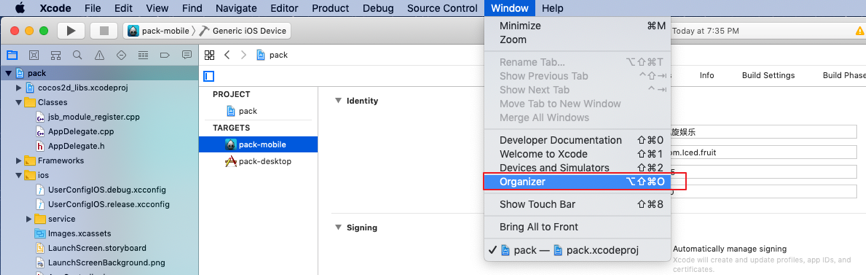
点击下载和证书描述文件放到一起。然后按照上面👆下载文件的方法使用transmission.sh 工具进行下载到本地并进行安装。

* 上传ipa包到商店
  + 打ipa包：使用vpn创建的证书及描述文件，就直接安装，云主机创建的就下载出来后安装，然后打开Xcode，按照如图所示进行设置：

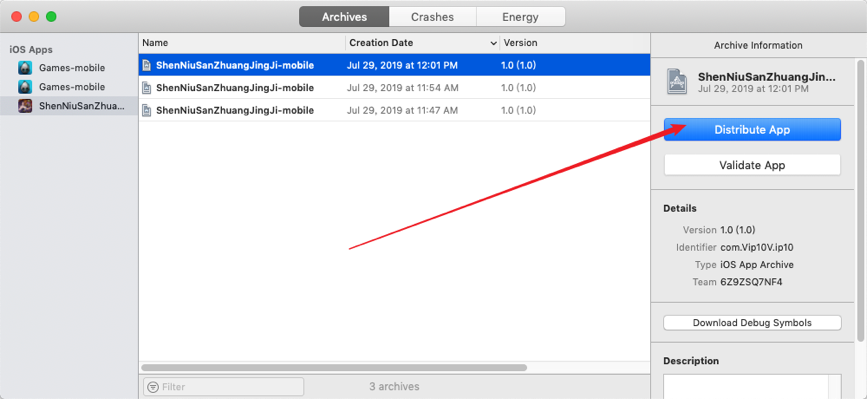


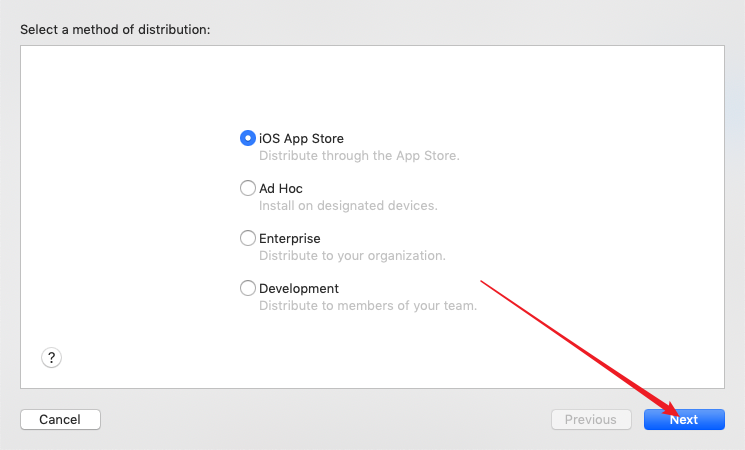


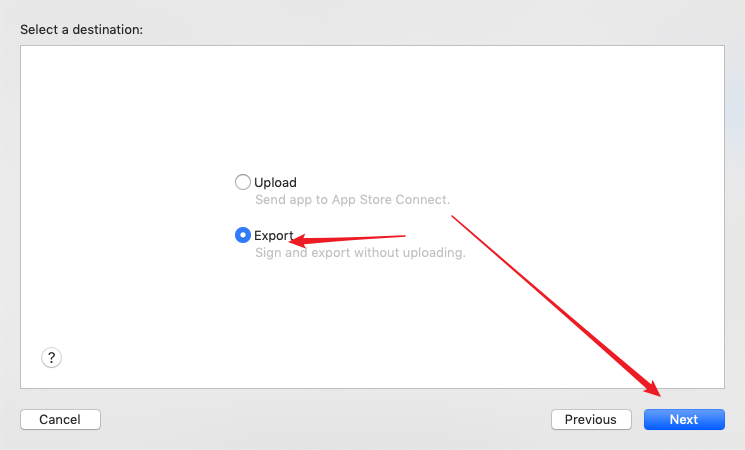
点击Archive会进行构建、打包及签名

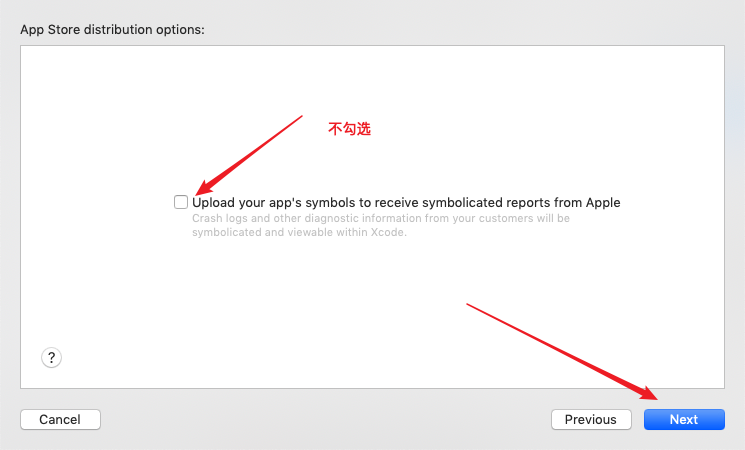


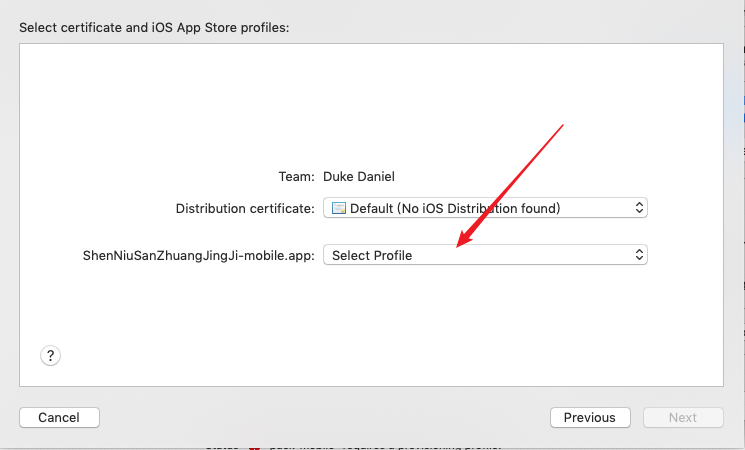
点击这里导出ipa包





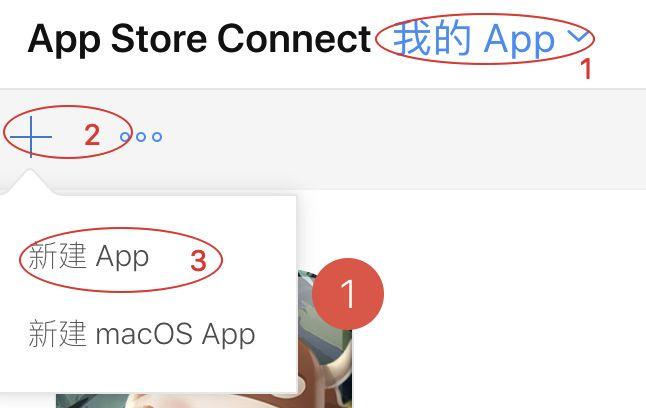






选择当前的描述文件，点击Next选择目录进行导出

* 在浏览器(vpn电脑里浏览器或云主机浏览器)中打开<https://appleid.apple.com> 进行登陆，点击生成密码，把专属密码记录下来
* 创建应用：在浏览器(vpn电脑里浏览器或云主机浏览器)中打开<https://appstoreconnect.apple.com/login> 创建应用，具体见图：

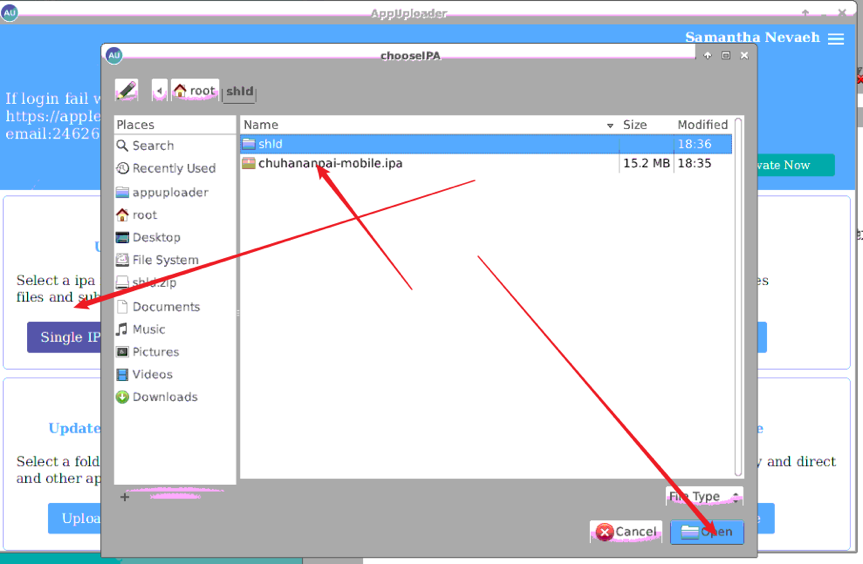


* 上传ipa，分云主机和vpn两种，下面分别说一下
  + Vpn，打开Application Loader.app软件，使用提包账号和专属密码进行登录，登陆后会显示下面这样：



点击选取，选择打包好的ipa，进行上传

* + 云主机上传：

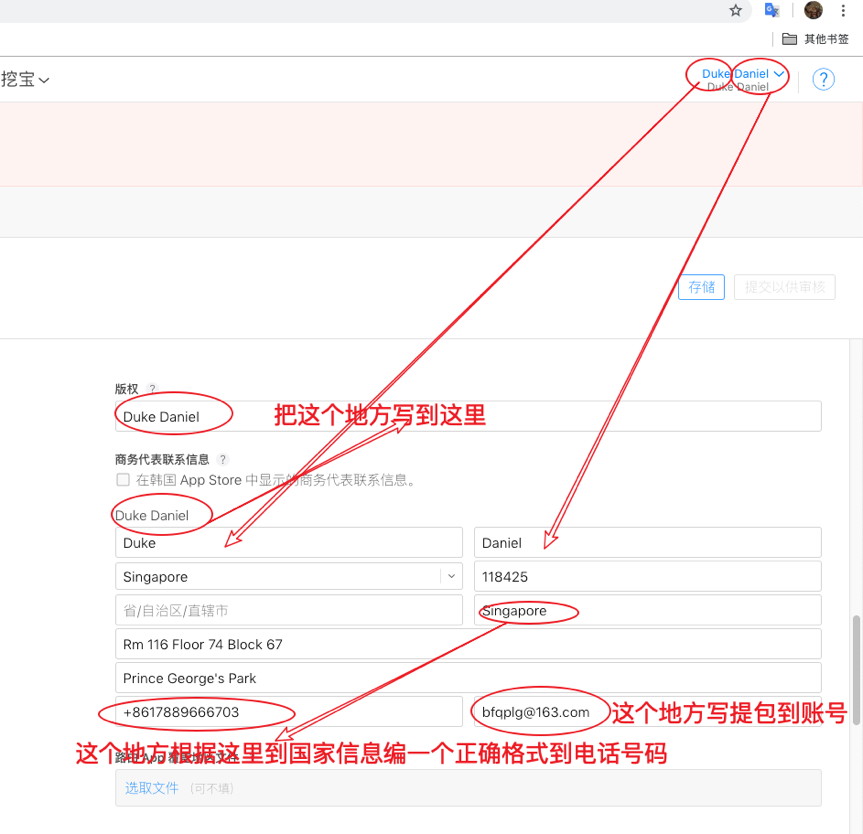


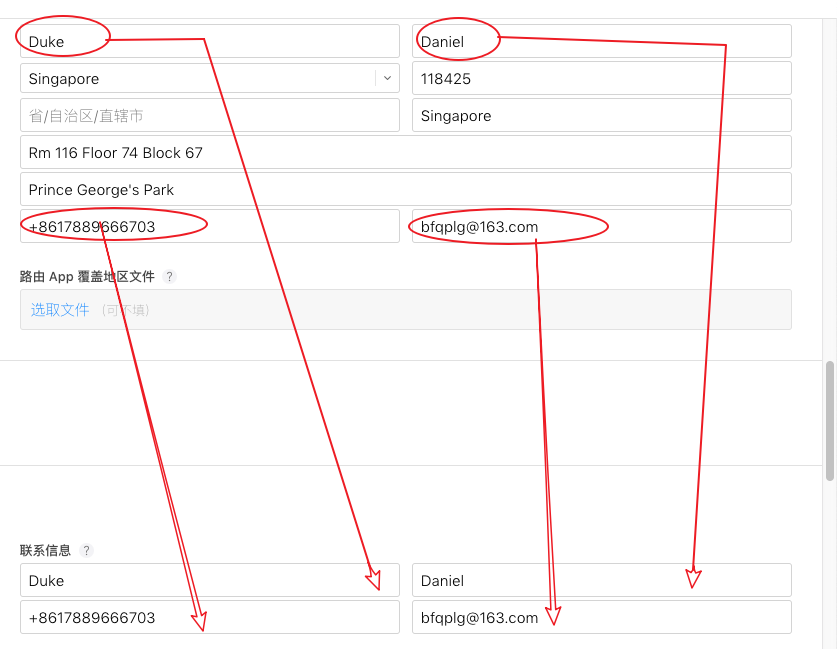
* 填写元数据：看图





****

****

****

****

****

**上传完毕后返回继续点击加号选择app**

****

****

****

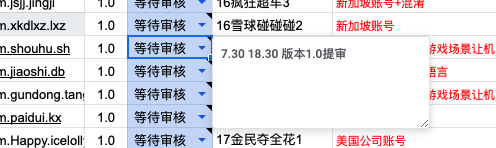
****

提审之后前两个选项不勾选，只勾选最后一个广告标志符，具体见下图：

****

## 提审之后

到此为止应用已经提交到苹果 App Store，提审之后要记录一下App信息里面的Apple ID，把它填写要提包进度表中，并把审核状态更改为等改审核，在等待审核上面鼠标右键选择插入备注填写提审时间及版本下面截图举例：



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

，有结果后使用提包设备及提包网络进入查看结果，一般状态为：**等待审核=>正在审核=> 等待开发者发布/二进制文件被拒绝/元数据被拒绝**。元数据或二进制文件被拒绝要根据情况进行修改，等待开发者发布说明过审，等待添加热更新，并把审核信息发送到**提包-技术**群中。

## 注意事项：

* **不要使用自己自己的工作电脑进行提包，要使用专用电脑连接专用网络进行提包**
* **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​**
* **工作电脑只能连接tb2-2.4G或tb2-5GWi-Fi，严禁连接其他任何Wi-Fi**
* **Wi-Fi的IP地址使用固定ip，不允许私自更换IP**
* **提包完成后上传至本地git库，过审后推到远程git库**
* **每次提包的证书、描述文件、p12、专属密码自己保存好，过审之后上传至1.9服务器**