Jenkins自动打包部署—iOS

---20180905

环境准备： OSX+xcode+gitlab+Jenkins

开发者证书、APP描述文件

工程打包类型：

单一工程打包（存在的自动打包的工程有3种，一种是单一的工程打包；一种是集成cocoapod的打包；还有一种是有www文件需要插入到工程里的打包）

阅读对象、要求：普通人均可，要求具备一定的Linux操作、熟悉OSX系统基本操作、具备一定的脚本代码编写能力。

说明： 本文档仅用作指导完成Jenkins自动打包的基本操作，如需深入了解可参考Jenkins官方网站的指导说明。

# Start

## 1准备：

MacBook Pro（苹果电脑）：MacOS High Sierra 10.13.3

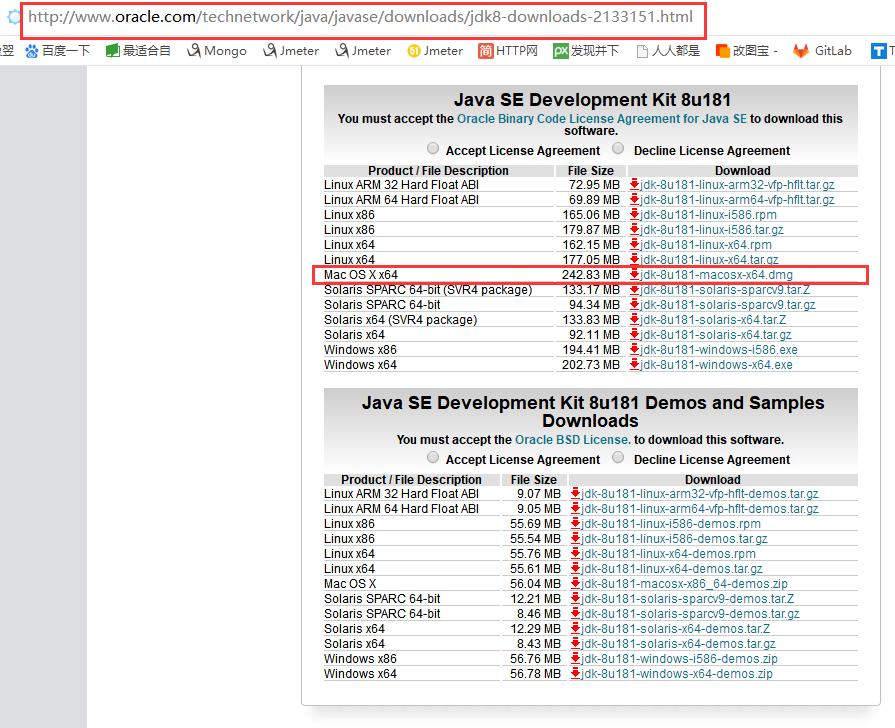
Xcode 9.4.1

Jenkins 2.140

git 1.8.3.1

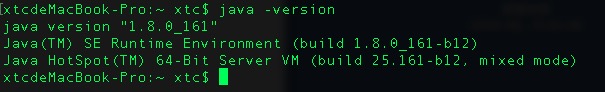
## 2安装java环境：

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>下载安装包，下载完成后，解压放置到意向目录下，配置java环境变量，即可使用。（如已有可跳过）



验证java环境安装完成：

java –version



## 3安装Jenkins：

终端输入命令进行安装

$ brew install Jenkins

进行链接 launchd 配置文件

$ ln -sfv/usr/local/opt/jenkins/\*.plist ~/Library/LaunchAgents

$ launchctl load~/Library/LaunchAgents/homebrew.mxcl.jenkins.plist//如果要其他机器也可以访问，需要进入文件把ip地址改为广播地址:--httpListenAddress=0.0.0.0

## 4运行Jenkins

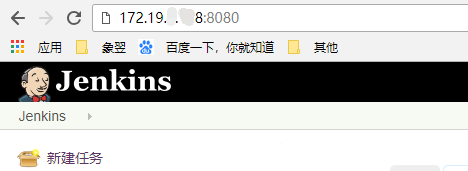
直接终端输入命令：Jenkins，即可启动Jenkins

也可采用以下命令进行启动Jenkins，多用在需要修改部分参数情况

/Library/Java/JavaVirtualMachines/jdk1.8.0\_161.jdk/Contents/Home/bin/java -Dmail.smtp.starttls.enable=true -jar /usr/local/opt/jenkins/libexec/jenkins.war --httpListenAddress=0.0.0.0 --httpPort=8080

## 5进入Jenkins：

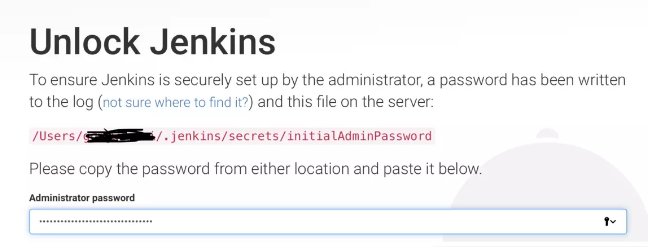
浏览器输入localhost：8080即可进入，或输入IP：端口 进入。



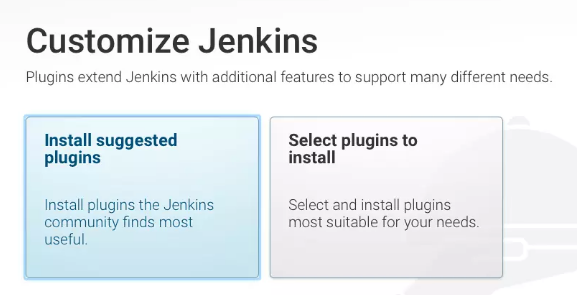
### 5.1登陆Jenkins：

进入到Jenkins首次登陆窗口后，需要填写秘钥登陆（保存在

/Users/用户名/.jenkins/secrets/initialAdminPassword）如提示无权限，需要对文件/文件夹进行授权



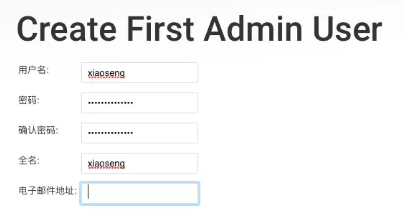
### 5.2安装默认插件：



一段时间后刷新页面（安装完成后（均已有打勾或打×））

插件下载安装完后，页面不会自动刷新，需要手动进行刷新页面，可能存在一些插件没有安装成功，会打×，需要手动去进行安装，后面进行讲解（第6章节第1条既6.1）。

### 5.3设置用户账号：（此处采用默认admin账号）



此时点击右下方的跳过，使用admin账号进行

## 6 配置Jenkins

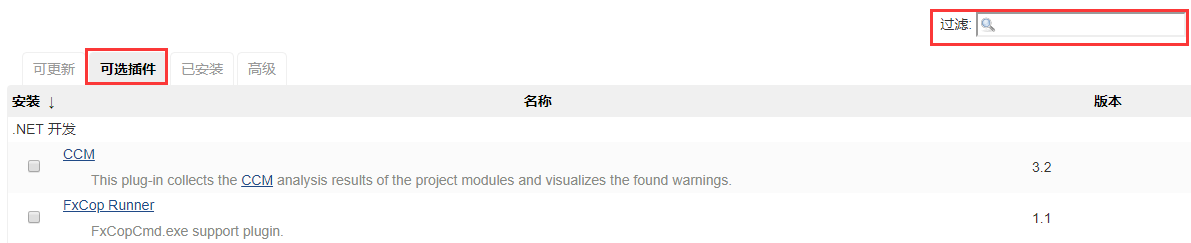
### 6.1插件补充：

默认插件存在有失败或我们需要的插件不在默认里，则需要进行补充安装。（系统） 系统管理->管理插件 这里我们需要添加GitLab Plugin、Gitlab Hook Plugin、 xcode integration、Keychains and Provisioning Profiles Management。





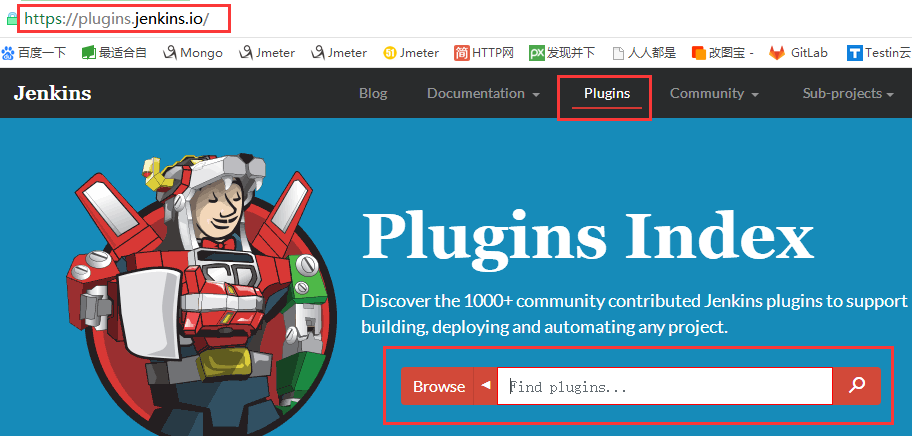
进入到管理插件界面后，点击进入可选插件



然后依次搜索插件进行直接安装，全部补充的插件安装好后，选择重启Jenkins。

### 6.2补充插件安装----上传安装

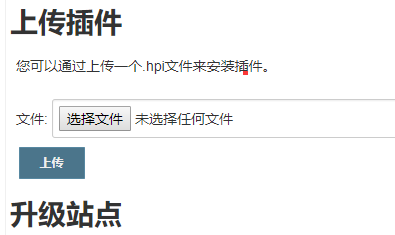
如遇网络问题，或没有自动查找到对应的插件，可到Jenkins.Plug（<https://plugins.jenkins.io/>）进行下载需要的插件。需下载.hpi类型的文件。



下载好插件后，进入 系统管理->管理插件 ，点击高级，拖动页面到“上传插件”

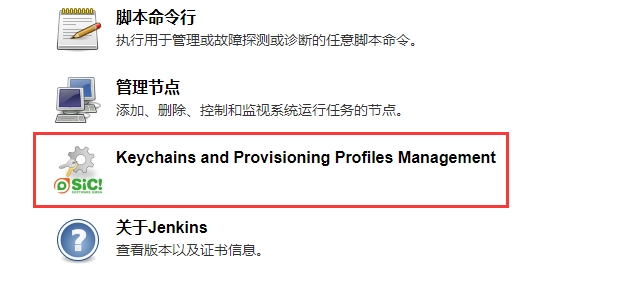


点击选择文件，在弹出的对话框中选择上面下载的插件文件，然后点击上传即可



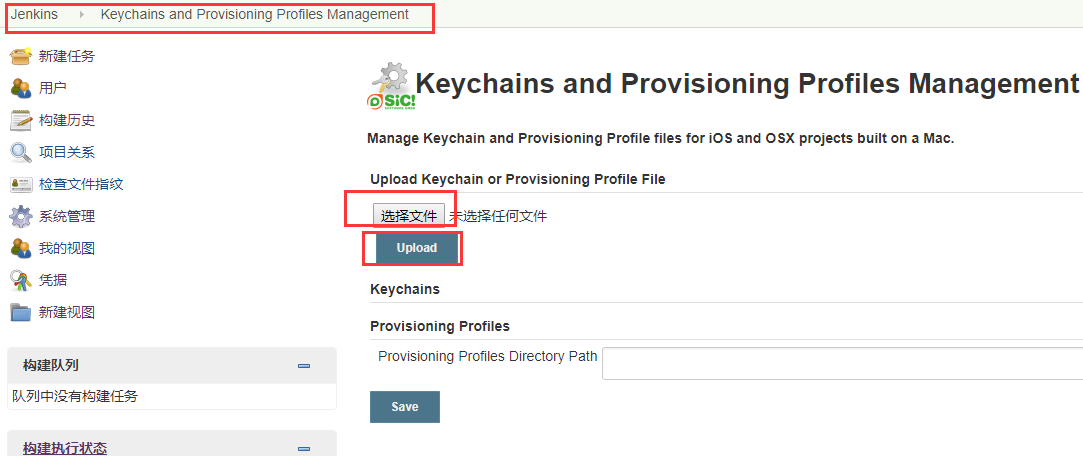
### 6.3添加钥匙串、描述文件

进入 系统管理，下拉找到Keychains and Provisioning Profiles Management插件，点击进入，



**PS：**

事先需要找开发给到开发者证书（看需求给，此处只需要打包发布用的包，既采用的是发布证书），APP描述文件。依次选择下面的选择文件—upload来完成上传操作。（需要提前把开发者证书导入到本地电脑，既双击证书，如是P12文件要求输入给证书的人设置的密码，然后会把证书添加到本地钥匙串中；如直接是证书，点击后则直接会把证书添加到本地钥匙串中）（如需分类，则选择添加到“登录”分类下即可）



打包用证书即login.keychain-db。此时需要注意，上传到Jenkins的只认.keychain后缀和.mobileprovision后缀的文件，其他文件都会上传失败。所以我们要把这个上传的钥匙串后缀的-db去掉，文件名修改成login.keychain，然后再上传。

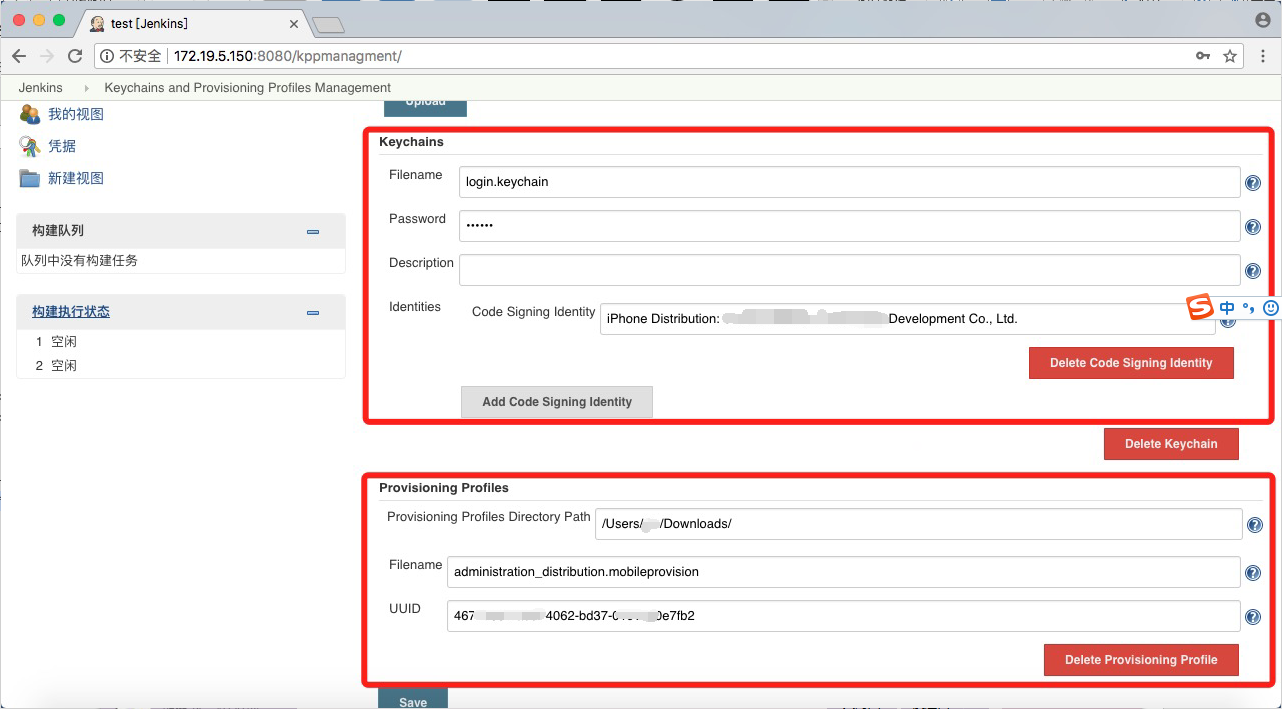
上传成功后Keychains下边会显示Filename为login.keychain，密码什么的都是自动的（P12文件前面输入的密码，直接是证书也会有证书的密码），可以添加打包用的证书。

添加好证书后，如下图，则可进行上传描述文件了，先添加描述文件的导向路径，

/Users/[user name]/Library/MobileDevice/Provisioning Profiles

。描述文件和钥匙串一样的添加上传方式。添加后就可以看到描述文件信息了。

最后点击保存（save）即可。



## 7创建工程任务

点击Jenkins首页，左侧显示选择列表，右侧显示现有工程任务，点击左侧的新建任务，进入新建任务对话框，输入任务名称，点选自由风格，点击页面底部“确定”即可。





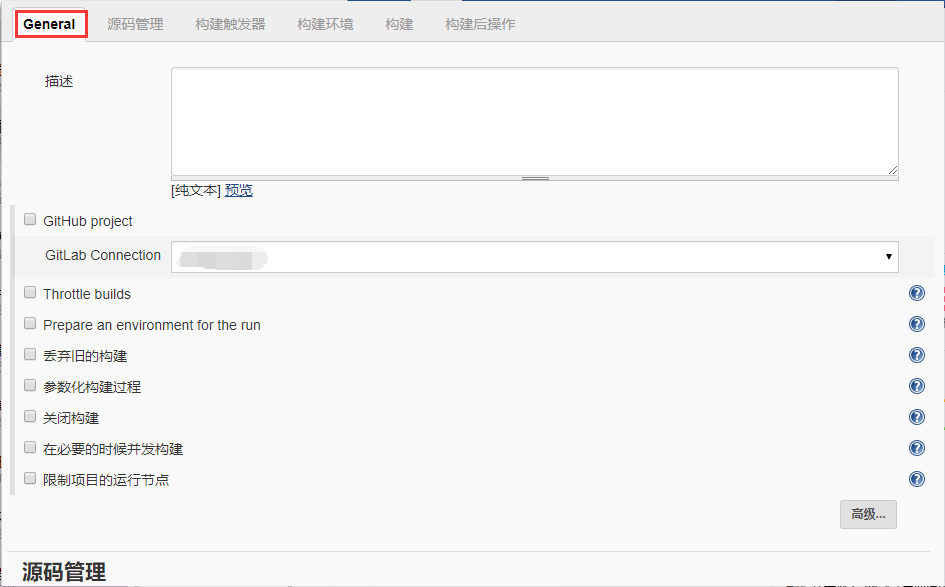
当然，如果我们已经创建了任务，并且其他配置都一样就改个证书什么的，可以下滑看到可以复制一份现有的任务，输入名字就可以选择，选择后点击页面底部“确定”即可。



### 7.1工程配置

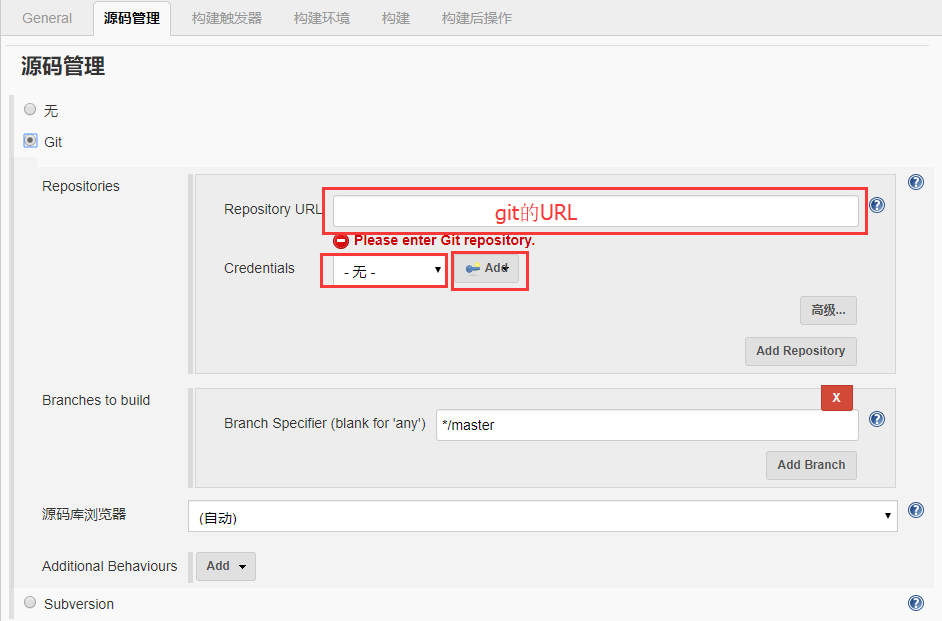
#### 7.1.1 General

上一步点击“确定”后，会直接进入到该工程任务的配置项页面中，在这里，我们可以再描述中输入对该工程任务的一个描述内容。及其他的一些需要的选项。



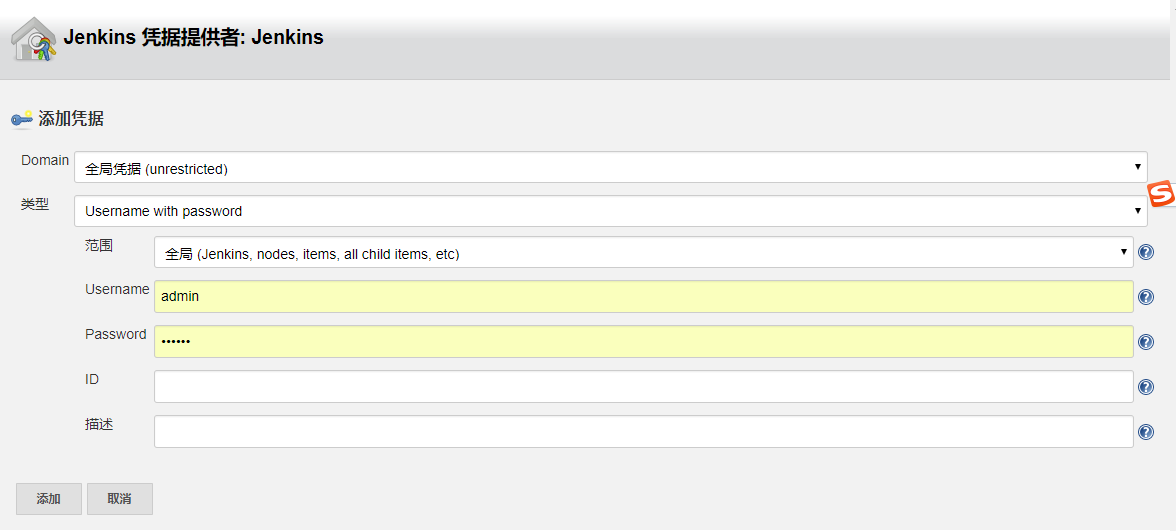
#### **7.1.2 源码管理**

当然，这个General标签下内容可以都不做选择。直接进入“源码管理”标签下的内容，并进行编辑。我们这里则跳过General，如有需要可根据相关内容进行选择。



URL：填写gitlab的项目URL地址，此处需要注意，安装Jenkins的方式不同则此处选择的地址则不一样：如采用Jenkins官网下载的Jenkins.war包进行起的Jenkins服务，则此处需要使用ssh的URL地址，则需要进行秘钥对的匹配。详细可见node项目的文件；如采用的是命令模式brew进行安装启动的服务，则可直接使用用户名和密码的形式进行。

此处我们使用的是命令模式，步骤如下：1、登录Gitlab，前往并进入意向项目，点选http，复制http的URL；2、回到Jenkins，粘贴到URL项里，选择Credentials里的账号，如不存在需要的账号，可点击add，选择Jenkins，进入到添加账户凭证（凭据）对话框，



Domain默认的全局凭据即刻，

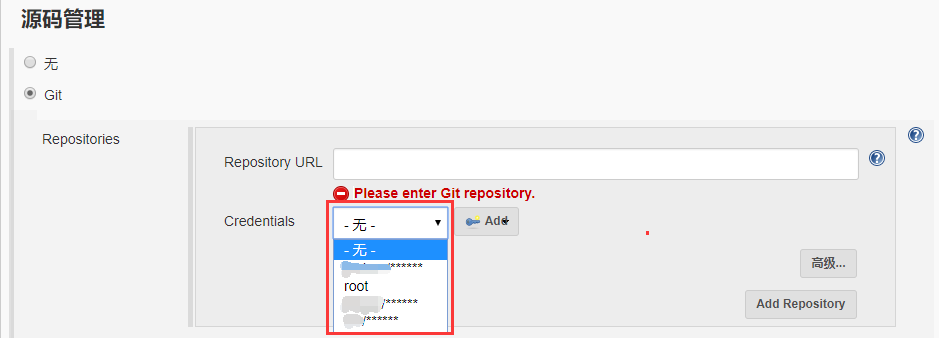
我们这里用的是http形式进行连接git，故类型中选择Username with password即可，

范围选项默认选择全局即可

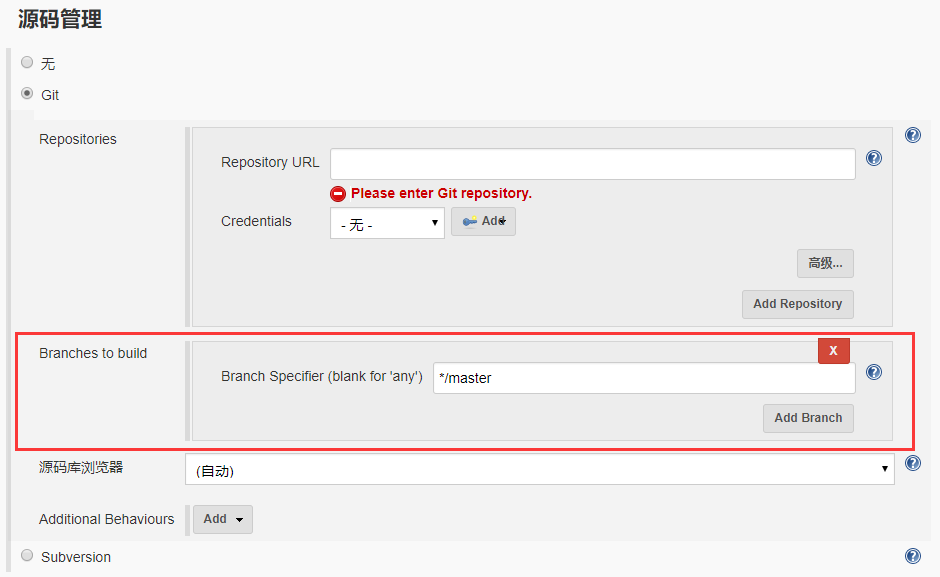
Username 填入登录Git的账号名即可

Password 填入对应Git账号的登录密码即可

完成以上内容后，点击底部的添加，即可完成账户凭证的添加。



返回选择Credentials里刚刚添加的账号，然后选择要进行打包作业的代码的分支，注意前面要有\*/，然后就完成了“源码管理”的内容填写，



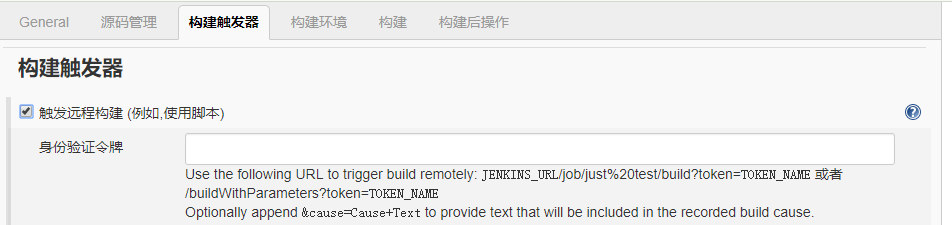
#### 7.1.3 构建触发器

完成“源码管理”的内容填写后，进入“构建触发器”，打包作业，称之为构建，该标签的意思则是要让Jenkins如何在什么条件下开始进行构建打包作业，对常用几项进行区别讲解：



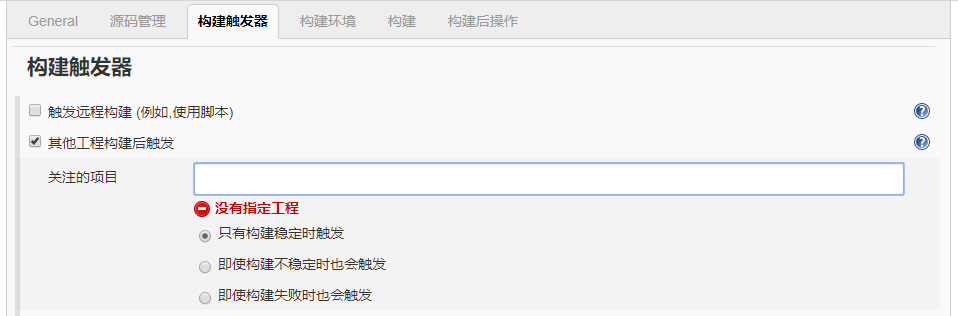
触发远程构建：

勾选此项后，选项下端会有相关提示，通过复制这个URL地址，在别的机器上打开这个URL就相当于给这个jenkins服务器发送了一个构建请求。只不过这个请求是在其他人电脑上发出的。这里重点来看看URL的组成，第一个参数JENKINS\_URL,这里我们写IP地址或者机器hostname，第二个参数TOKEN\_NAME就是你在身份验证令牌文本输入框输入的值。这里我们把令牌设置成123456，然后我就在我机器的另外一个浏览器来模拟远程构建，这个时候我的远程构建地址就是这样的：localhost:8080/job/Test1/build?token=123456，此时在他人的浏览器中输入该远程构建地址，即可开始构建该任务。



其他工程构建后触发：

选择一个其他构建任务，在这个构建任务完成构建后，当前任务会紧接着进行构建，多用于多部件需要分开启动的、带有联动关系的项目。



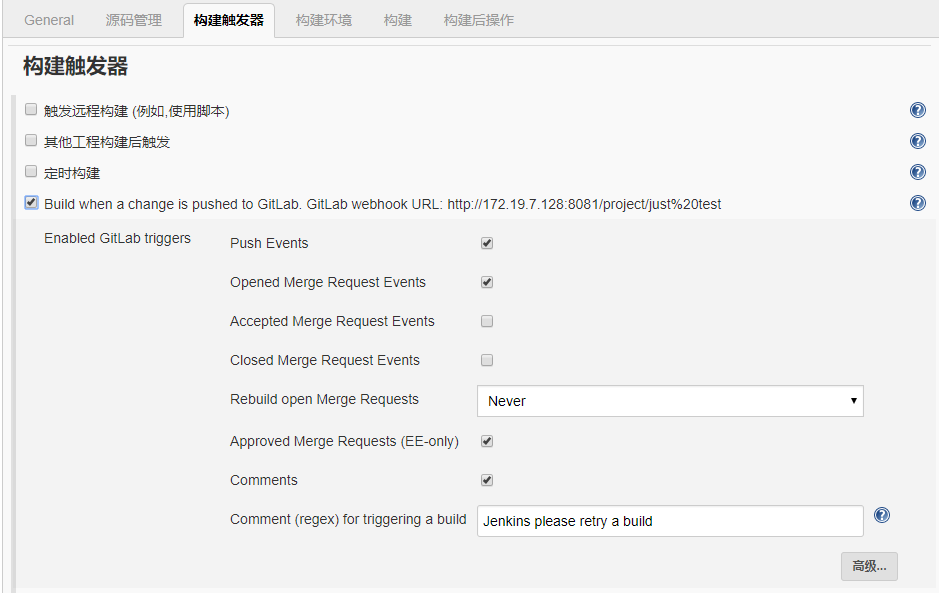
定时构建：

顾名思义，既在输入框内输入指定的时间，到了该时间则自动开始构建，时间的填写有一定的要求规则，详细可网上搜索Jenkins定时构建，既有详细说明，这里不做赘述。



Build when a change is pushed to GitLab.：

既当Gitlab上对应的分支代码上有更改时，Jenkins会立即开始构建该任务。



轮询 SCM：

既在指定的时间内，Jenkins会随机选定其中的某个时间进行查询Gitlab对应分支是否有代码改动，如有，怎进行构建任务。



#### 7.1.4 构建环境

进行编辑构建任务所处的环境，这里是iOS的打包，且使用的是xcodebuild进行，所以我们这里需要填入证书和描述文件。

Keychains and Code Signing Identities

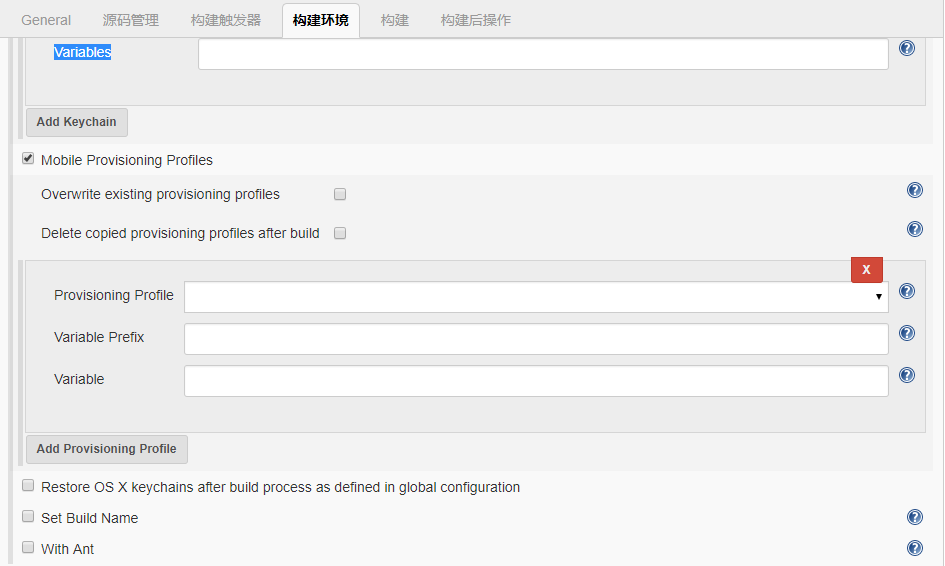


Keychain：填入证书名称，既6.3章节中上传的证书，此处如果6.3上传完成，则可进行选择。

Code Signing Identity：选择对应的签名证书。此处如6.3上传完成，则可进行选择。

Variables：上两项填写好后，该处会自动生成。

Mobile Provisioning Profiles



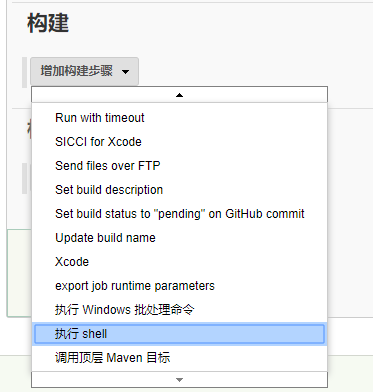
Provisioning Profile：选择描述文件

Variable：选择了描述文件后，此处自动生成

#### 7.1.5 构建

此处进行选择搭建构建的方式、构建步骤。

构建的方式多种多样，组合方式随意，因此此处不做赘述，仅对此次任务使用的进行说明。



#PROJECT\_NAME = "项目名"  
#CURRENT\_SHEME = "Administration"  
#CONFIGURATION = "这里是release "  
#BOUNDLE\_VERSION = "boundle的版本号"  
#ExportOptionsPlistPath = "/Users/用户名/"  
  
xcodebuild clean  -project  Administration.xcodeproj \  
                  -configuration Release \  
                  -target Administration  
                  #-alltargets  
echo "================= Clean一下 Over ================="  
# archive  
echo "============== archive =================="  
xcodebuild archive  -project Administration.xcodeproj \  
                    -scheme Administration \  
                    -configuration Release \  
                    CFBundleVersion=0.8.2.0 \  
                    -destination generic/platform=ios \  
                    -archivePath build/Administration.xcarchive   
                   # CODE\_SIGN\_IDENTITY="${\_\_signingIdentity}"  
                    # export ipa  
echo "+++++++++++++++++ exportArchive +++++++++++++++++"  
xcodebuild -exportArchive -archivePath build/Administration.xcarchive \  
                          -exportPath build/Administration\  
                          -destination generic/platform=ios\  
                          -exportOptionsPlist /Users/。。。/ExportOptions.plist \  
                          -allowProvisioningUpdates

echo "+++++++++++++++++ uploadpgyer +++++++++++++++++"

curl -F "file=@/要上传的文件的路径" -F "uKey=蒲公英上用户秘钥" -F "\_api\_key=蒲公英上应用的接口秘钥" https://www.pgyer.com/apiv1/app/upload

此处我们使用的是shell脚本进行构建，

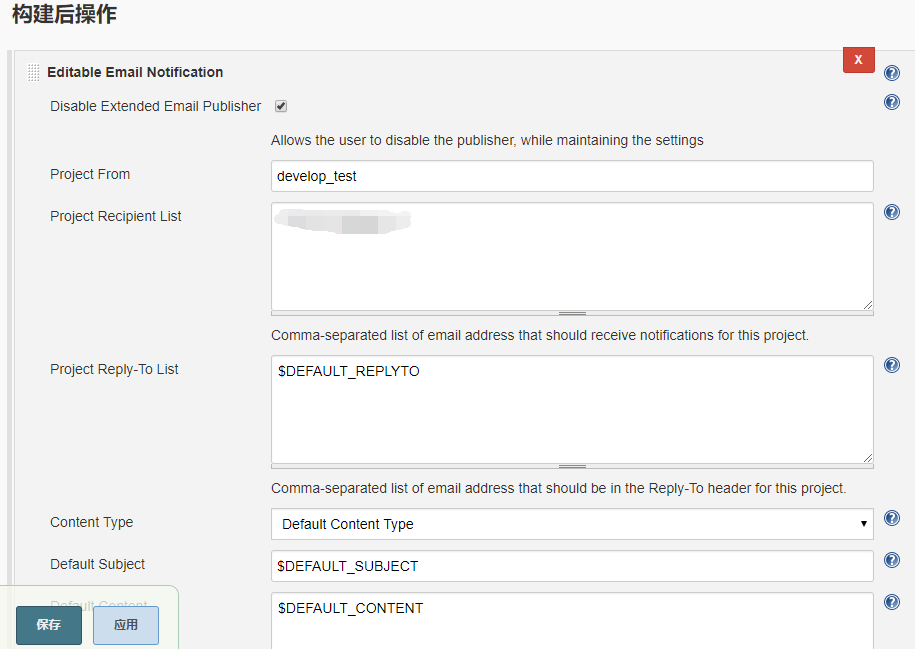
#### 7.1.6构建后操作

构建完成后进行的一些操作，可不选择，这里我选用的**Editable Email Notification**

**勾选**Disable Extended Email Publisher后，构建成功的则不发送邮件提醒。

Project From ：填写项目名称，可随意填写，用于自己识别是哪个构建任务就可以

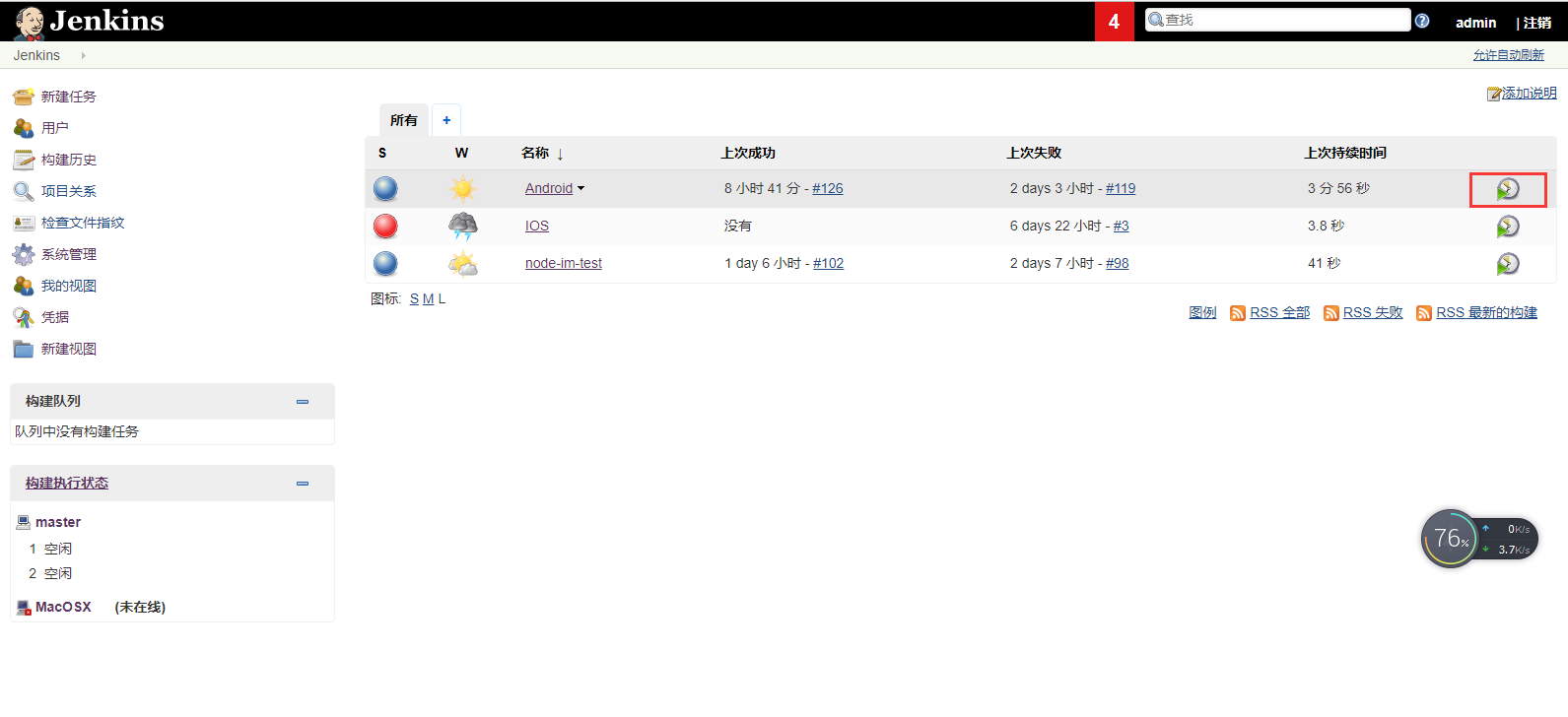
Project Recipient List：填写接收提醒的地址，这里使用一个QQ邮箱做接收



完成后，即可点击保存，则完成了全部的工程任务的配置。

## 8 发起构建

配置好任务后，可进行手动操作进入构建状态，有三个地方可触发，分别为：首页的工程后面的三角符号；右击工程名右边的下拉三角，点击立即构建；点击工程名，进入工程，选择右侧目录的“立即构建”。





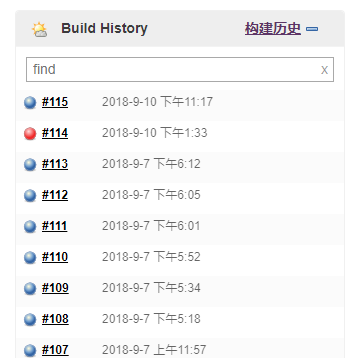


**8.1构建历史查看**

首先是在首页中可以快速查看上一次构建信息：



其次是点击项目名称进入项目后，在左下部分区域将显示历史记录



点击单次构建历史，可进行查看本次构建的详细信息系统：



同时点击上面页面的左侧功能按钮，进入对应的功能

其中：

“变更记录”可进行查看本次构建详细的变更信息

“控制台输出”可查看本次构建产生的日志

“轮询日志”可查看本次构建进行轮询的信息

## 9 补充终端的操作

Jenkins会在终端中生成一个工作空间，用于保存项目任务的的构建，默认路径为/Users/xtc/.jenkins/workspace/构建任务的名称

打成的包默认保存在/Users/xtc/.jenkins/workspace/构建任务的名称/build

里的应用名命名的文件夹里（如有做输出路径的限制，则在限制的路径中）。



终端可能需要安装git命令：brew install git

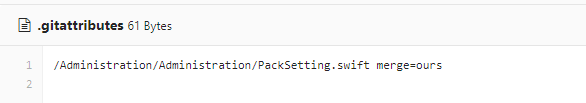
安装好后需要做初始化（your\_name,your\_email,gmail均和自己的匹配即可）

git config --global user.name "your\_name"

git config --global user.email "your\_email@gmail.com"

## 10关于忽略文件

在进行代码同步时，有一些文件的内容，存在一定的个性化设置或本地化生产的内容，这时候我们需要把我们不需要或者说不能要的文件做忽略处理，这里采用的是在工程根目录下生成忽略文件 **.gitattributes** 来进行忽略不需要同步的文件，其中路径建议采用绝对路径来进行编写，格式为：路径 merge=ours。



## 11更多关于

Gitlab的操作：

Linux Git操作：

Shell脚本的编写：

Xcode工具使用：

Xcodebuild插件的使用：

Git常用命令：

参考文献：