# 更新

## ？

1 不同版本目录的服务器版本标识每次更新要求全部更新

2测试用不同的渠道号即可

Apk版本号的最后一位表示渠道号 与Resources下的\_\_DOWNLOAD\_SOURCES\_\_内容对应

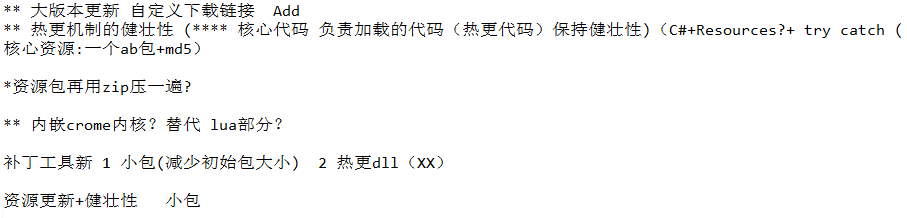
Version的内容跟渠道号没有直接关系， 但第三位一般表示渠道号，热更一般改变中间的数字（但改变第三个数字要求也能热更）

压缩包没用？

整包输出半包输出啥时用？

大版本更新？

之前标准的流程？



## TODO

IOS （IOS改变第三个数字才算热更 relegou(也就是前两个版本改变都是大版本更新)）[OK]

Editor模式处理 [OK]

内网测试环境？[直接用不同的渠道号测试]

拷贝补丁到http目录

渠道号处理[版本号的最后位表示渠道号，要是内容一样，直接复制补丁给其它渠道号]

中间版本号变化测试 [OK]

.apk变成一个网址测试 [OK]

1,跨越n个大版本如何更新 [每次更新都会用最新的服务器版本文件]

2,灰度控制 [用不同的渠道号]

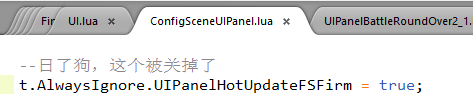
3,编辑器模式处理(跳过更新检测？) [OK]

## Q&A

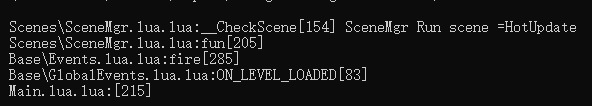
### 安卓手机版SceneMgr.lua function t:\_\_CheckScene(index) 重复调用，导致UIPanelHotUpdateFSFirm被太早关闭？更新流程中断

\_\_CheckScene竟然调用了两次,第二次调用\_\_CheckScene把UIPanelHotUpdateFSFirm关了,

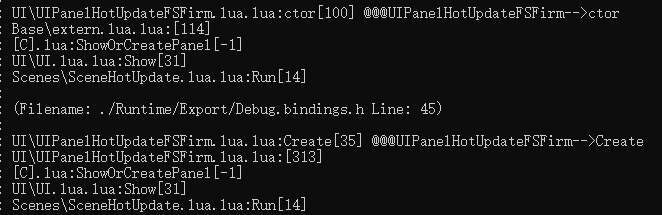
解决方案就是加这个标志：



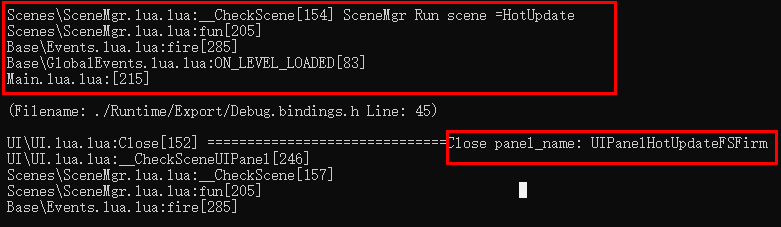
是的在调用\_\_CheckScene时不再关闭UIPanelHotUpdateFSFirm



🡺



🡺

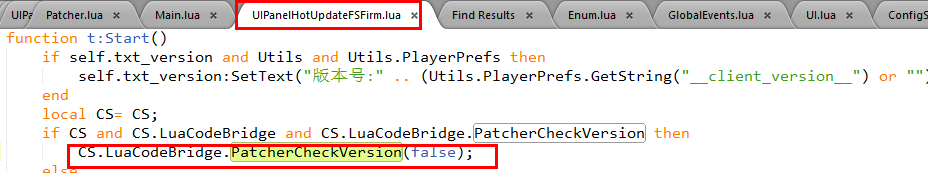


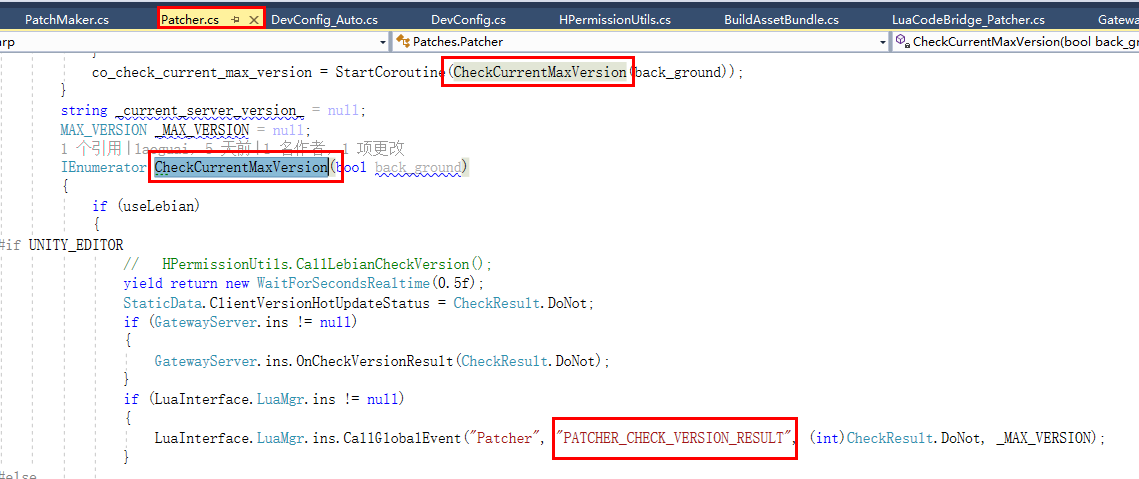
## 代码执行流程

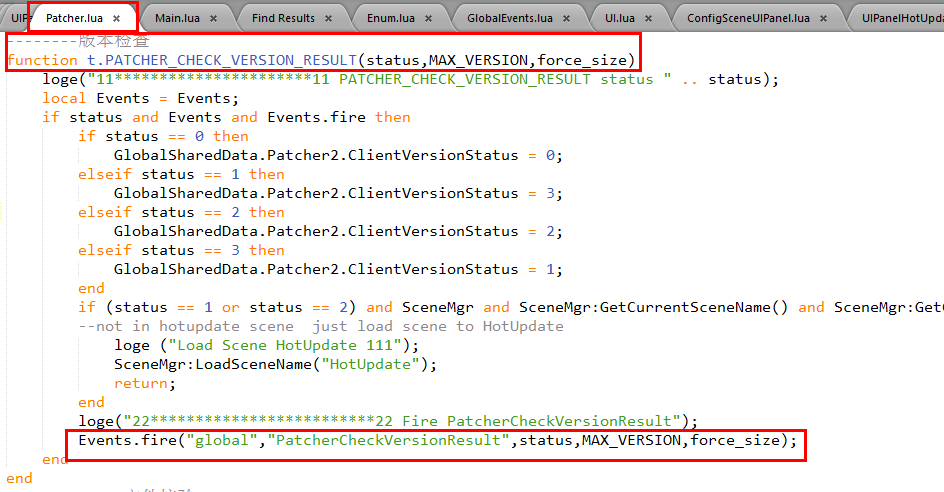
检查版本🡪校验文件

### 检查版本

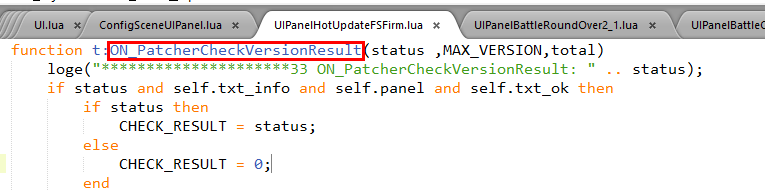
Main.lua🡪hotupdate(scene)🡪SceneHotUpdate.lua🡪UIpanelHotUpdateFSFirm

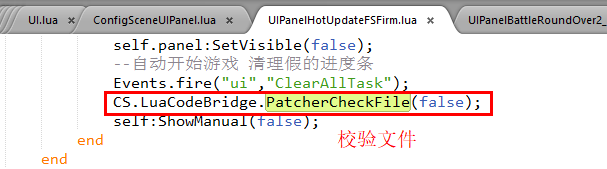






### 校验文件





## 版本制作说明

## 运行流程

FirstScene🡪Patcher.cs🡪Patcher.lu🡪Hotupdate(Scene)🡪 UIPanelHotUpdateFSFirm

🡪不需要更新Hotupdate还是会通过下面流程加载…

Main.lua🡪 SDKInitFinish🡪 RunGame🡪 UI.Show("UIPanelLoadingFSFirm",Enum.UISort.Peek5);

🡪 SPLASH\_FILM\_DONE🡪 Hotupdate(Scene)

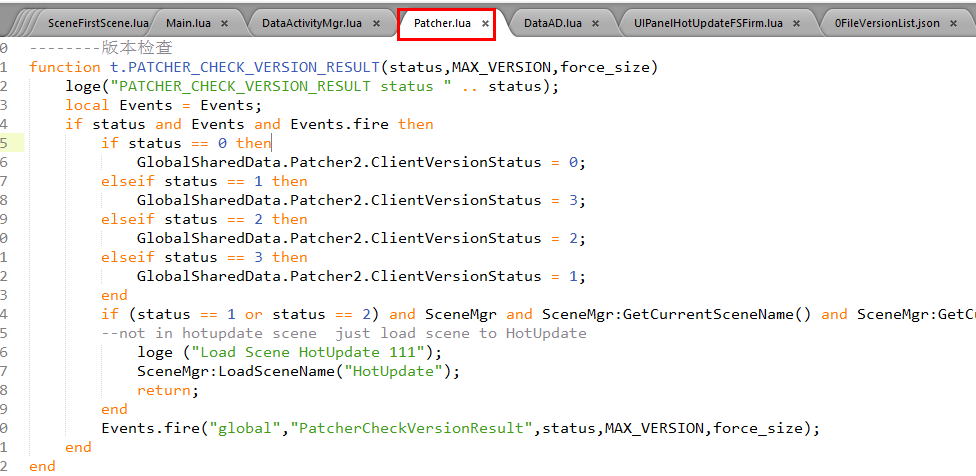
1.拉取MAX\_VERSION.txt服务器最新版本

2.对比版本信息确认是否需要热更或者强更

3.需要版本热更的话 拉取FULL\_VERSIONS.json 做进一步检查具体版本 IOS 是FULL\_VERSION\_IOS.json

4.下载地址会是cdn的patch目录下的android\_1.8.1.zip 或者ios\_1.8.1.zip

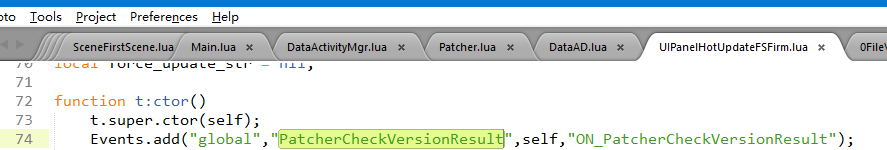
### Patcher.lua 版本一致

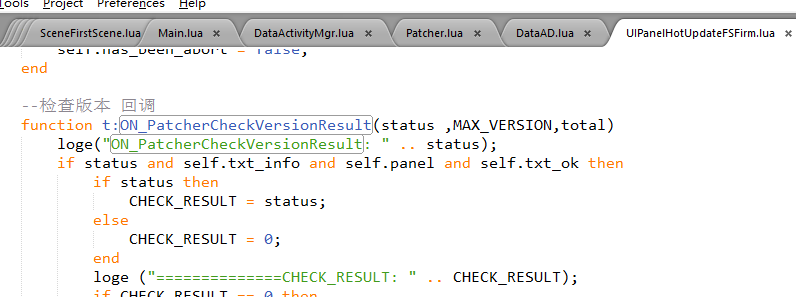


🡪

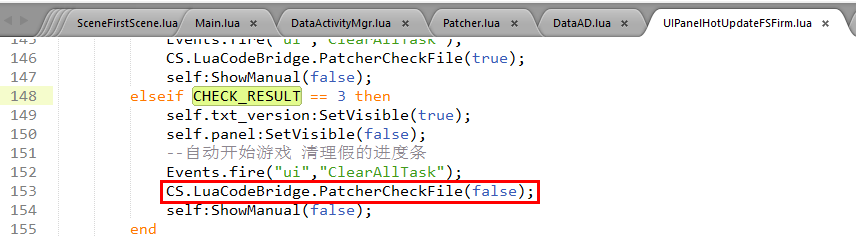


🡺

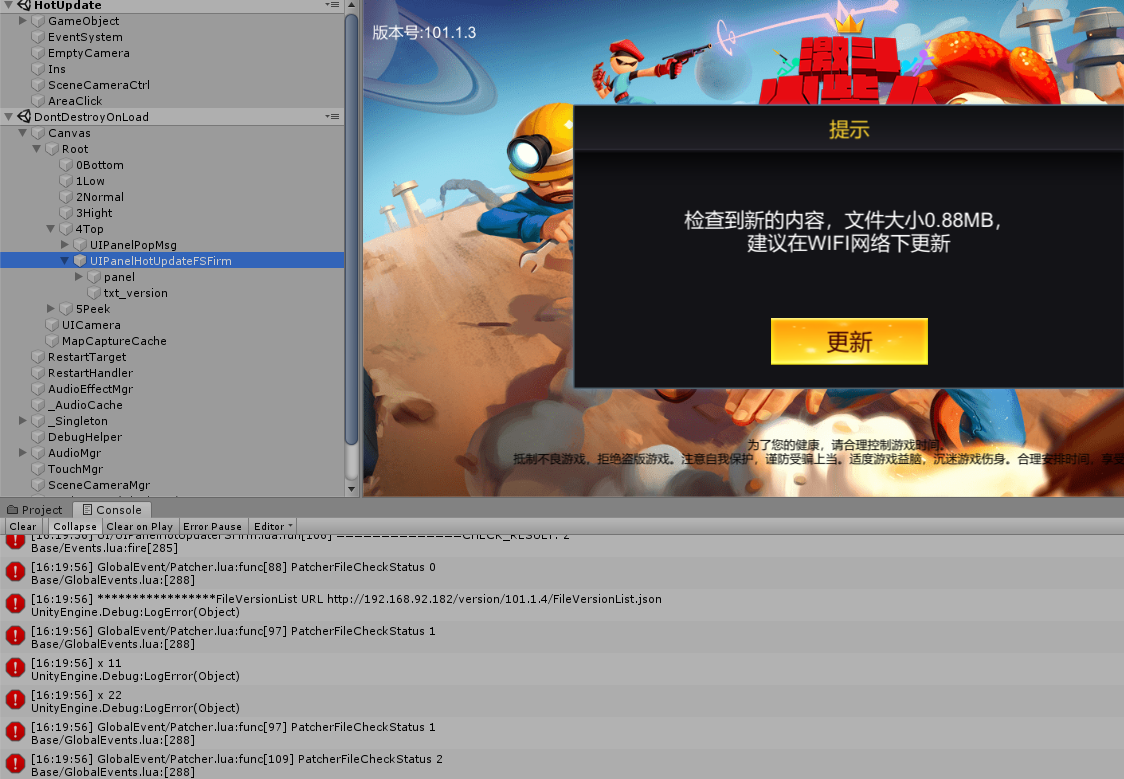




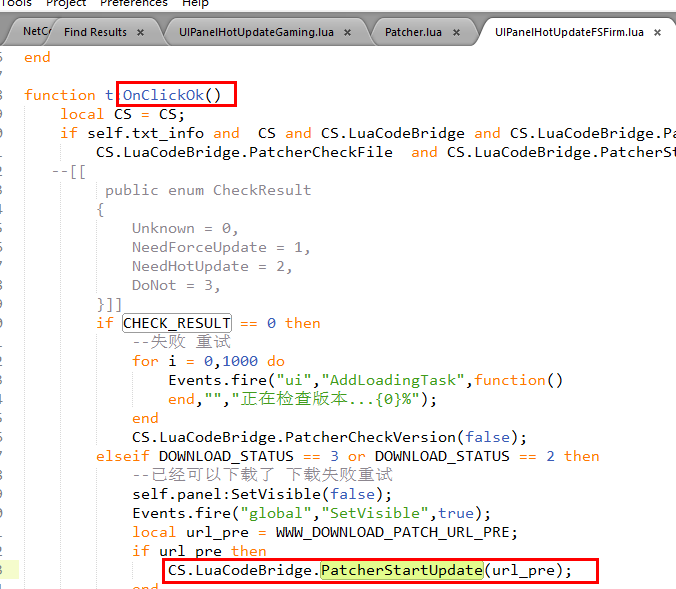
🡺校验



### 热更



开始更新：



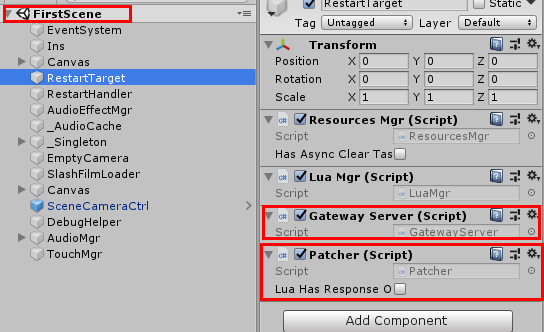


### 检查 版本目录 是否需要清理掉

Patch.CheckVersionFileValidate

### UIPanelHotUpdateFSFirm

### FirstScene



## 打包流程

### 版本流程

### 补丁工具/---1---BuildPatch\_Android Step1 build AssetBundle 这步就拷贝到VersionFiles目录了

TryDeleteDir("AssetBundles");

//SetupABNames();

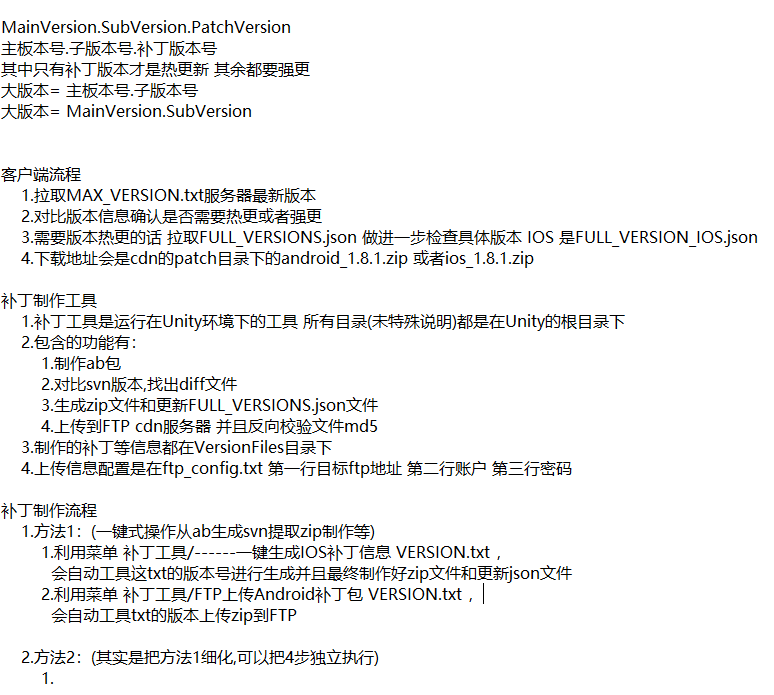
\_InnerBuildAllBundle(BuildTarget.Android);

CopyDir("AssetBundles", "VersionFiles/AssetBundles/Android/AssetBundles");

Ref完整输出AB包

\_InnerBuildAllBundle(EditorUserBuildSettings.activeBuildTarget);

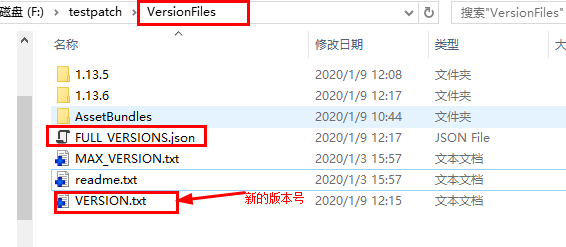
===============================================================================

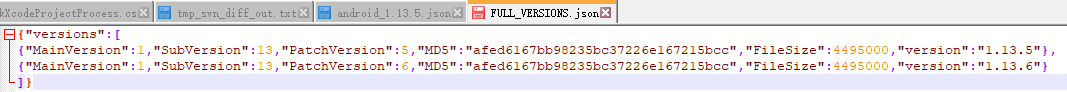




## 文件OR菜单说明

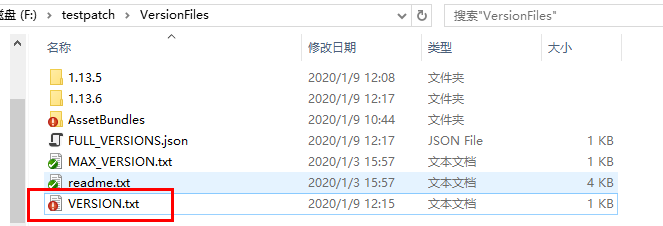
### FFULL\_VERSIONS.json

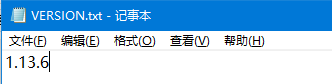






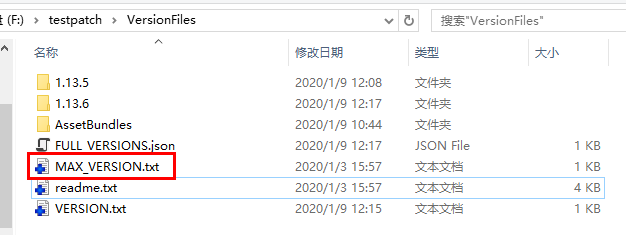
### VersionFiles/VERSION.txt

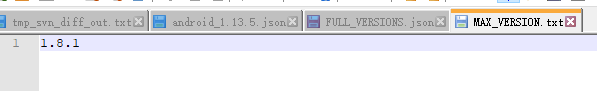


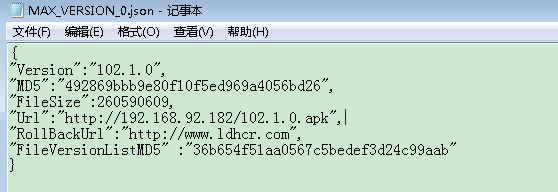


### MAX\_VERSION.json

服务器上的版本号文件

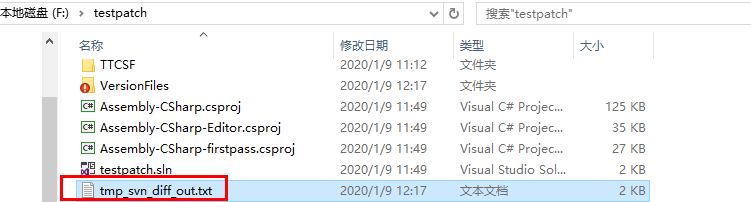


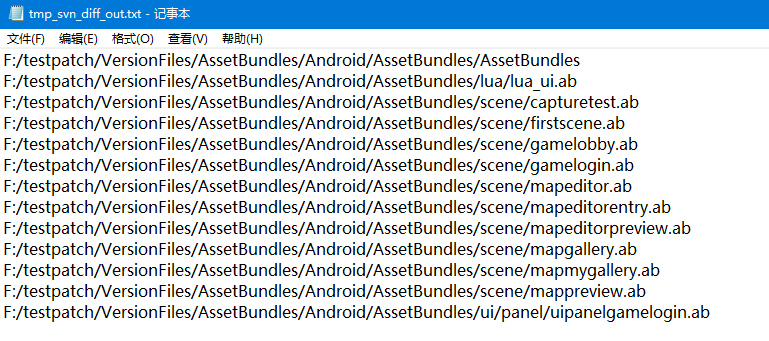




md5必须是小写。。

### tmp\_svn\_diff\_out





### ftp\_config.txt (要自己建)

上传信息配置是在ftp\_config.txt 第一行目标ftp地址 第二行账户 第三行密码

### VersionListCache.json

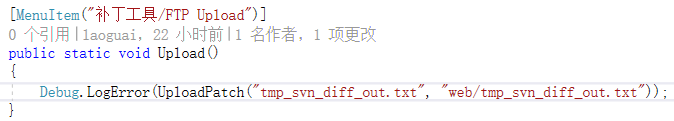
缓存的列表？

会把最新版本的FileVersionList下载下来放在文件VersionListCache.json里

### UIPanelHotUpdateFSFirm



### 补丁工具/FTP Upload

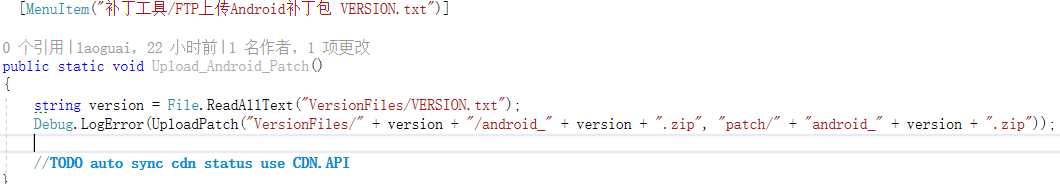


上传SVN差异文件

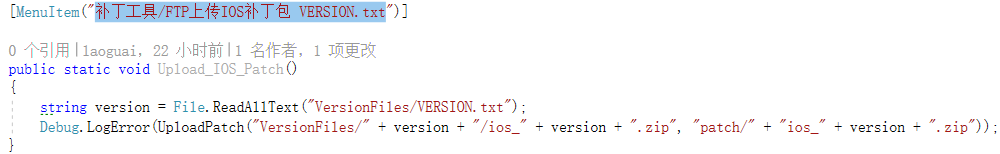
### 补丁工具/FTP上传Android补丁包 VERSION.txt 压缩包版本号由VERSION.txt内容决定

以VERSION.txt内容为1.13.6为例:

上传VersionFiles/1.13.6/android\_1.13.6.zip 🡪到远程的 patch/android\_1.13.6.zip位置



### 补丁工具/FTP上传IOS补丁包 VERSION.txt



同上传Android补丁包，把android\_ 换成ios\_

### 补丁工具/BuildZip AssetBundle完整输出zip到对应目录 VERSION.txt 方便内部测试

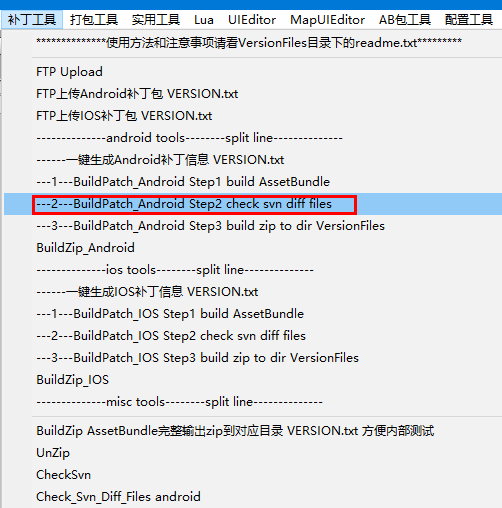
把整个AssetBundle输出到zip包里, zip名称包含的版本号由VersionFiles/VERSION.txt决定

### 补丁工具/UnZip

解压压缩包，现在固定名字路径是Assets/../a.zip

### CheckSvn

跟下面红框功能差不多:



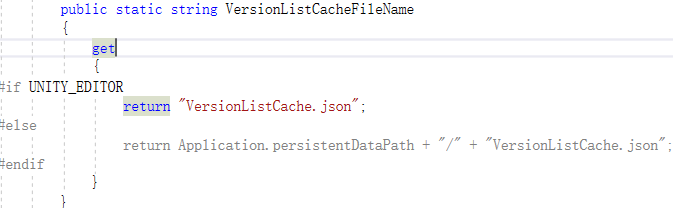
### Check\_Svn\_Diff\_Files\_android

跟上面红框功能差不多

### 补丁工具新（注释？）



生成全部AB包的校验码文件(在根目录),输出FileVersionList，放在Web服务器上，客户单动态去下载放在本地缓存文件VersionListCache：





\*\*\*\*\*生成几个核心AB包验码文件(在resources目录)、并确保StreamAssets目录下有相应的核心文件。

核心文件标志:

file.Contains("/lua/lua")

file.Contains("AssetBundles/AssetBundles")

file.Contains("fsfirm")

file.Contains("/engine/")



生成全部AB包校验码文件(在resources目录)、并把根目录下的AssetBundle目录拷贝到StreamAssets目录？



就是整包排除 skin 和 weapons剩下的ab包生成校验文件并把相应的ab拷贝到

StreamingAssets目录

## 现有流程弊端

步骤多

版本号要手动填

手动SVN处理

# 工具

StickNote

VNote

## Python

VS2017 python路径

<https://blog.csdn.net/haiqingonly/article/details/80168427>

# Jenkins

## Memo

### 更改管理员密码

<https://www.cnblogs.com/huandada/p/10894576.html>



### Python-jenkins

https://pypi.org/project/python-jenkins/0.4.14/

pip install python-jenkins==0.4.14



## HCR Android Jenkins

用的python版本3.7.1

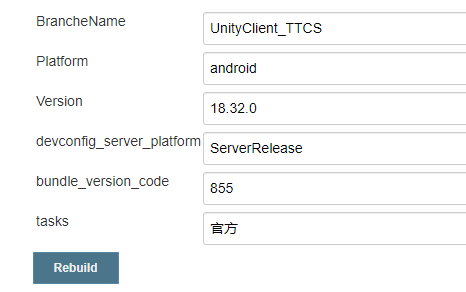
### step00\_svnup

BrancheName: UnityClient\_TTCS

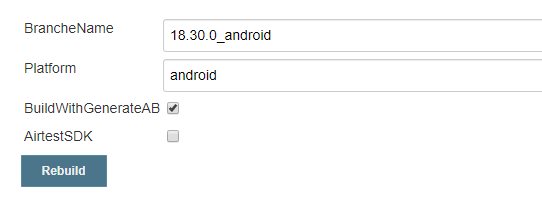
SvnRevision: 21876

Platform: android

### step01\_Modify\_Json\_File

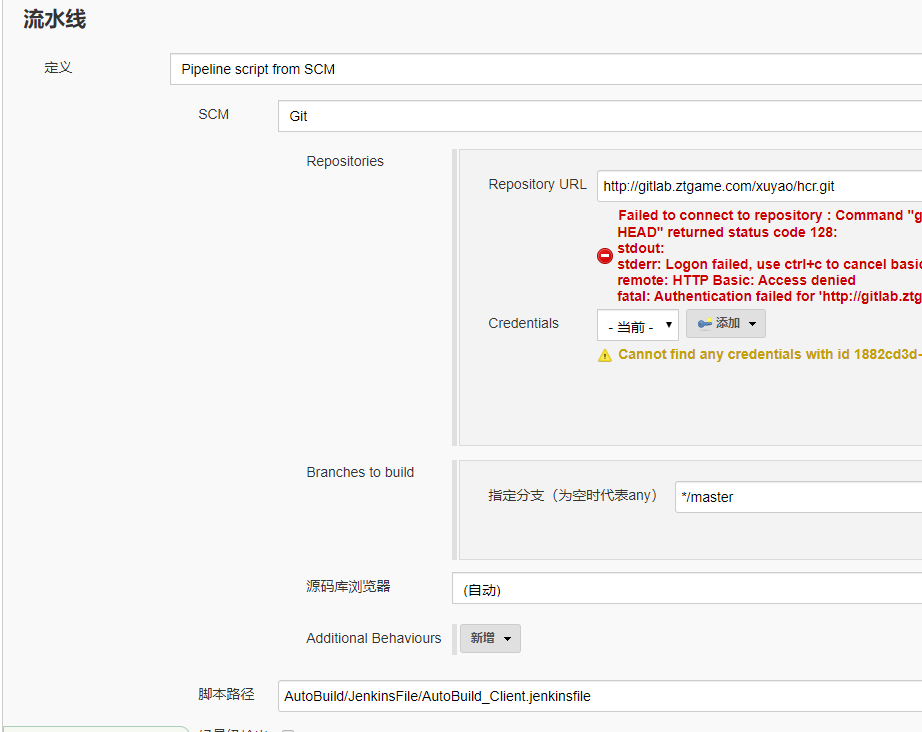


### step02\_Build\_Client

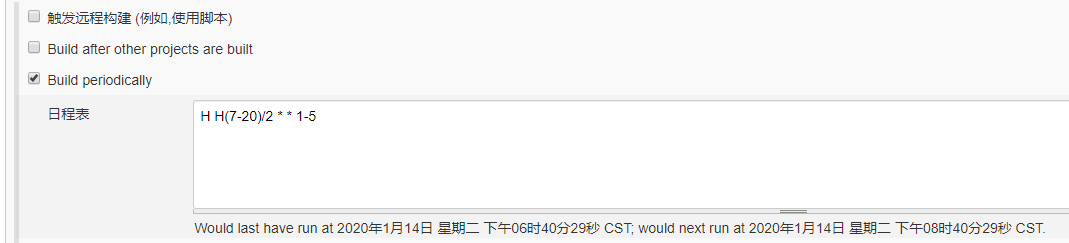


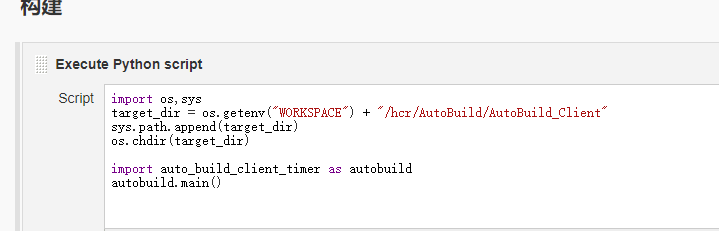
### Copy

### 流水线



### 计划版本





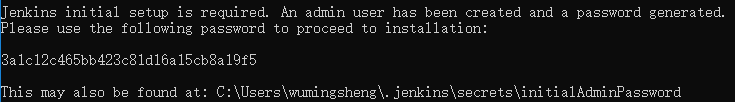
Jenkins.war

批处理:

e:

cd E:\Jenkins -----这里是 war包的路径

java -jar jenkins.war --ajp13Port=-1 --httpPort=8081

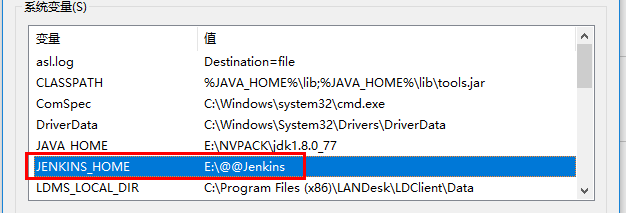


不起窗口运行jenkins

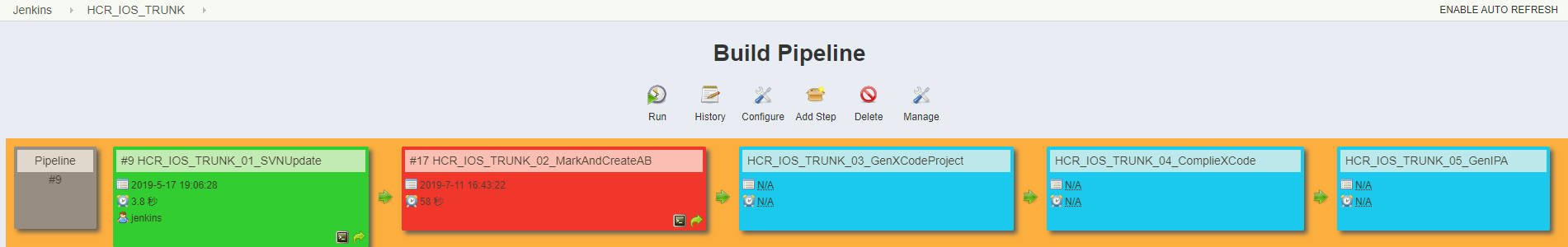
set JENKINS\_HOME=D:\Jenkins\Home

javaw -jar jenkins.war --ajp13Port=-1 --httpPort=8081

### 变更Jenkins主目录：（关闭Jenkins->添加环境变量🡪启动Jenkins）



## HCR\_IOS\_TRUNK



### HCR\_IOS\_TRUNK\_01\_SVNUpdate

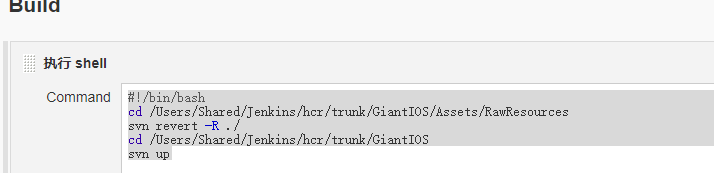
#!/bin/bash

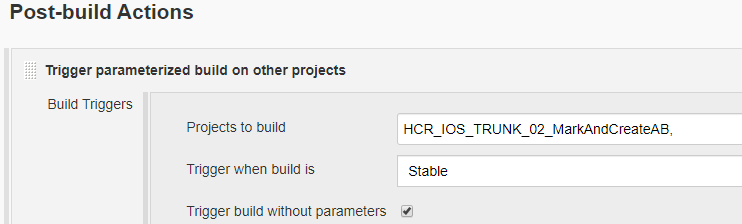
cd /Users/Shared/Jenkins/hcr/trunk/GiantIOS/Assets/RawResources

svn revert -R ./

cd /Users/Shared/Jenkins/hcr/trunk/GiantIOS

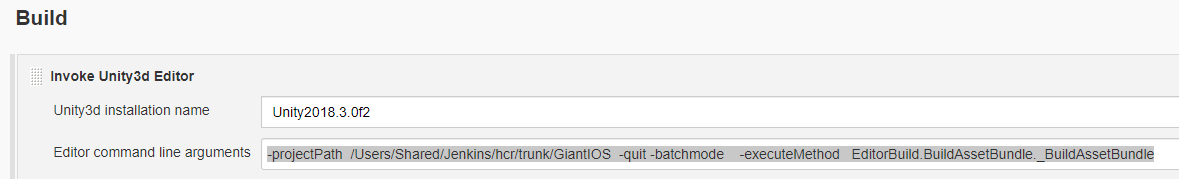
svn up

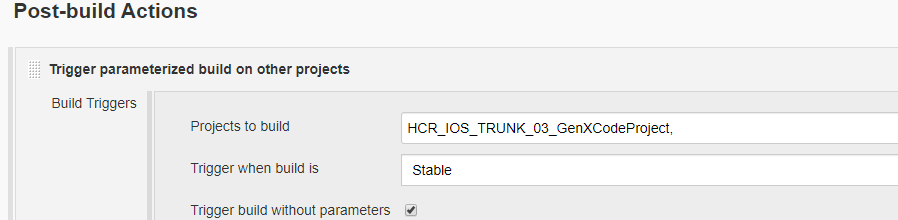




### HCR\_IOS\_TRUNK\_02\_MarkAndCreateAB

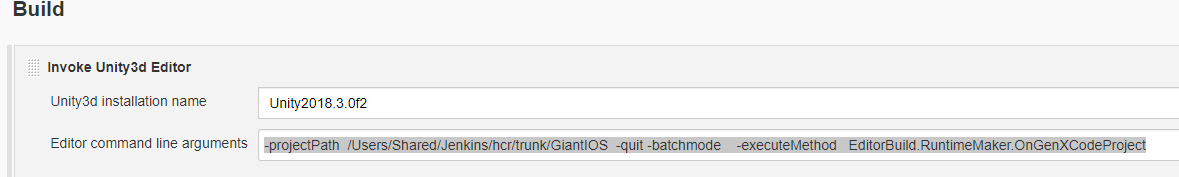
-projectPath /Users/Shared/Jenkins/hcr/trunk/GiantIOS -quit -batchmode -executeMethod EditorBuild.BuildAssetBundle.\_BuildAssetBundle

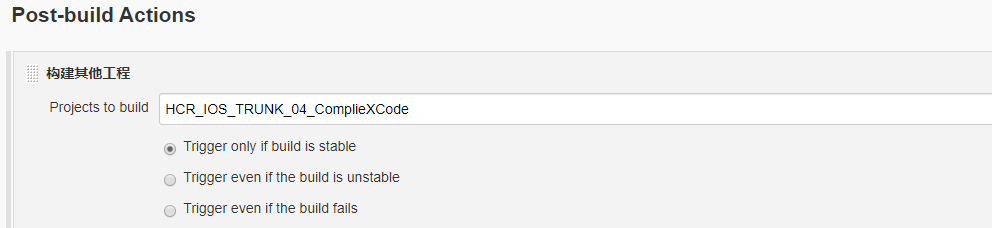




### HCR\_IOS\_TRUNK\_03\_GenXCodeProject

-projectPath /Users/Shared/Jenkins/hcr/trunk/GiantIOS -quit -batchmode -executeMethod EditorBuild.RuntimeMaker.OnGenXCodeProject





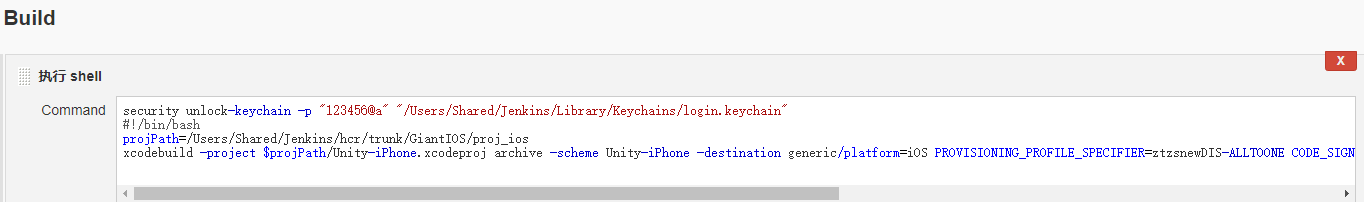
### HCR\_IOS\_TRUNK\_04\_ComplieXCode

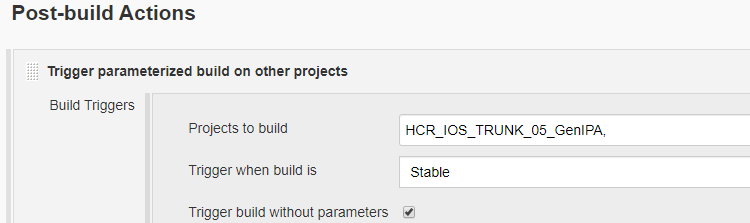
security unlock-keychain -p "123456@a" "/Users/Shared/Jenkins/Library/Keychains/login.keychain"

#!/bin/bash

projPath=/Users/Shared/Jenkins/hcr/trunk/GiantIOS/proj\_ios

xcodebuild -project $projPath/Unity-iPhone.xcodeproj archive -scheme Unity-iPhone -destination generic/platform=iOS PROVISIONING\_PROFILE\_SPECIFIER=ztzsnewDIS-ALLTOONE CODE\_SIGN\_IDENTITY="iPhone Distribution: Shanghai Giant Network Technology Co., Ltd." -archivePath $projPath/Unity-iPhone.xcarchive





### HCR\_IOS\_TRUNK\_05\_GenIPA

#!/bin/bash

projPath=/Users/Shared/Jenkins/hcr/trunk/GiantIOS/proj\_ios

xcodebuild -exportArchive -archivePath $projPath/Unity-iPhone.xcarchive -exportPath $projPath/out -exportOptionsPlist $projPath/../build.plist

cp $projPath/out/Unity-iPhone.ipa /Library/WebServer/Documents/jdhcr/ios/jdhcr-$(date +%m%d%H%M%y).ipa

mv $projPath/out/Unity-iPhone.ipa /Library/WebServer/Documents/jdhcr/ios/jdhcr2019.ipa



## ios\_branch\_18.6.0

### ios\_branche\_01\_svnupdate

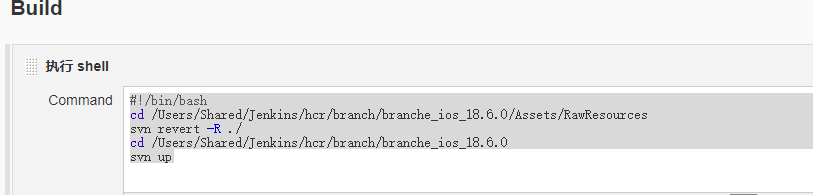
#!/bin/bash

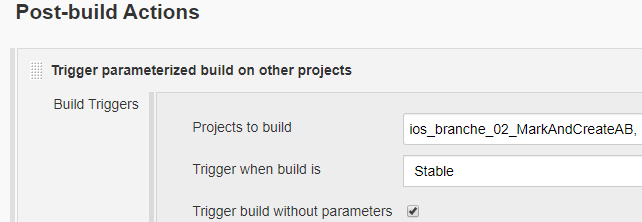
cd /Users/Shared/Jenkins/hcr/branch/branche\_ios\_18.6.0/Assets/RawResources

svn revert -R ./

cd /Users/Shared/Jenkins/hcr/branch/branche\_ios\_18.6.0

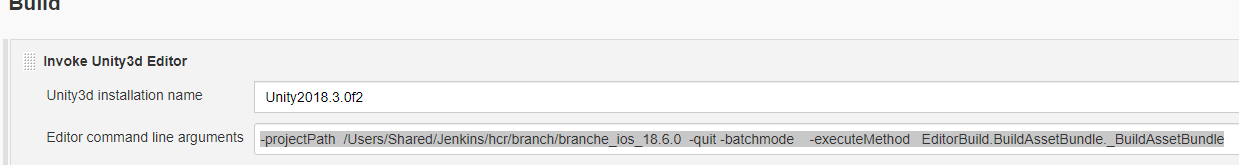
svn up

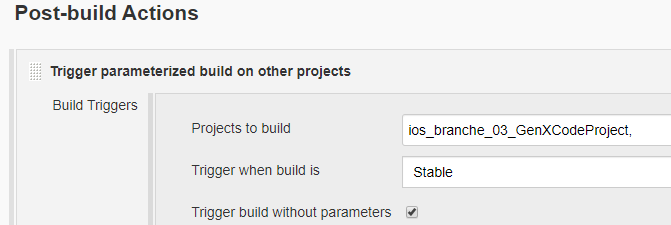




### ios\_branche\_02\_MarkAndCreateAB

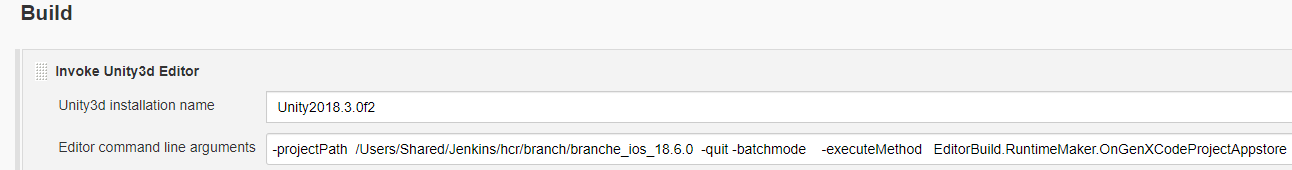
-projectPath /Users/Shared/Jenkins/hcr/branch/branche\_ios\_18.6.0 -quit -batchmode -executeMethod EditorBuild.BuildAssetBundle.\_BuildAssetBundle

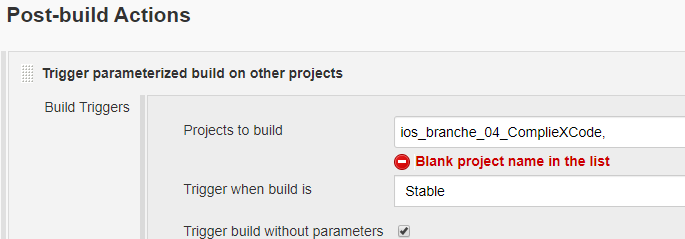




### ios\_branche\_03\_GenXCodeProject

-projectPath /Users/Shared/Jenkins/hcr/branch/branche\_ios\_18.6.0 -quit -batchmode -executeMethod EditorBuild.RuntimeMaker.OnGenXCodeProjectAppstore





### ios\_branche\_04\_ComplieXCode

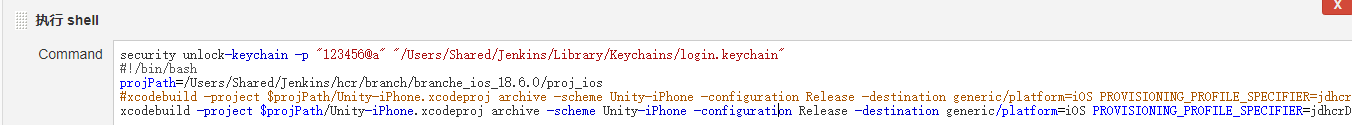
security unlock-keychain -p "123456@a" "/Users/Shared/Jenkins/Library/Keychains/login.keychain"

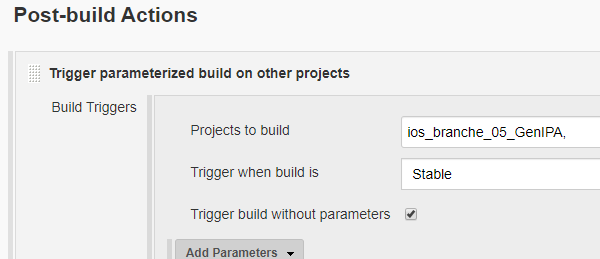
#!/bin/bash

projPath=/Users/Shared/Jenkins/hcr/branch/branche\_ios\_18.6.0/proj\_ios

#xcodebuild -project $projPath/Unity-iPhone.xcodeproj archive -scheme Unity-iPhone -configuration Release -destination generic/platform=iOS PROVISIONING\_PROFILE\_SPECIFIER=jdhcrDIS CODE\_SIGN\_IDENTITY="iPhone Distribution: Shanghai Zhengju Information Technology Co., Ltd. (TG93K6XG5H)" -archivePath $projPath/Unity-iPhone.xcarchive

xcodebuild -project $projPath/Unity-iPhone.xcodeproj archive -scheme Unity-iPhone -configuration Release -destination generic/platform=iOS PROVISIONING\_PROFILE\_SPECIFIER=jdhcrDEV CODE\_SIGN\_IDENTITY="iPhone Developer: junjie zhang (N86KABWGCZ)" -archivePath $projPath/Unity-iPhone.xcarchive





### ios\_branche\_05\_GenIPA

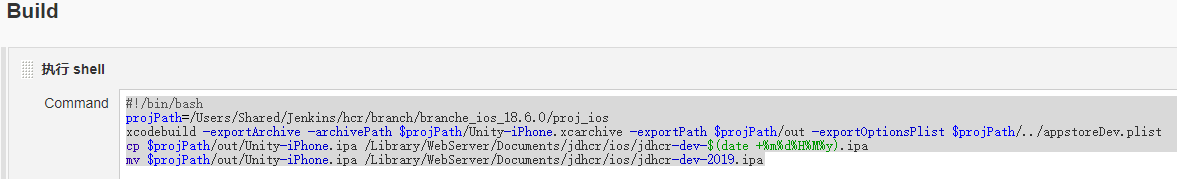
#!/bin/bash

projPath=/Users/Shared/Jenkins/hcr/branch/branche\_ios\_18.6.0/proj\_ios

xcodebuild -exportArchive -archivePath $projPath/Unity-iPhone.xcarchive -exportPath $projPath/out -exportOptionsPlist $projPath/../appstoreDev.plist

cp $projPath/out/Unity-iPhone.ipa /Library/WebServer/Documents/jdhcr/ios/jdhcr-dev-$(date +%m%d%H%M%y).ipa

mv $projPath/out/Unity-iPhone.ipa /Library/WebServer/Documents/jdhcr/ios/jdhcr-dev-2019.ipa



## ios\_tmp\_dis

### ios\_branche\_04\_CompileXCode\_Appstore-dis

security unlock-keychain -p "123456@a" "/Users/Shared/Jenkins/Library/Keychains/login.keychain"

#!/bin/bash

projPath=/Users/Shared/Jenkins/hcr/branch/branche\_ios\_18.6.0/proj\_ios

#disappstoreprofile

#xcodebuild -project $projPath/Unity-iPhone.xcodeproj archive -scheme Unity-iPhone -configuration Release -destination generic/platform=iOS PROVISIONING\_PROFILE\_SPECIFIER=jdhcrDIS CODE\_SIGN\_IDENTITY="iPhone Distribution: Shanghai Zhengju Information Technology Co., Ltd. (TG93K6XG5H)" -archivePath $projPath/Unity-iPhone.xcarchive

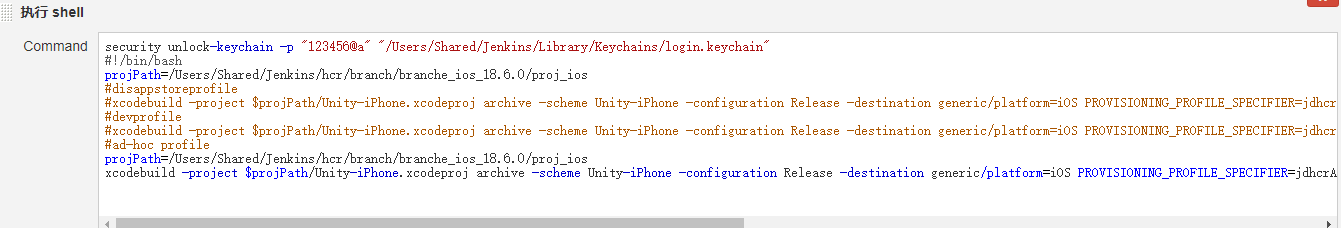
#devprofile

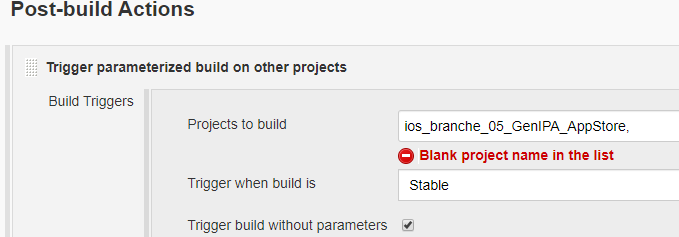
#xcodebuild -project $projPath/Unity-iPhone.xcodeproj archive -scheme Unity-iPhone -configuration Release -destination generic/platform=iOS PROVISIONING\_PROFILE\_SPECIFIER=jdhcrDEV CODE\_SIGN\_IDENTITY="iPhone Developer: junjie zhang (N86KABWGCZ)" -archivePath $projPath/Unity-iPhone.xcarchive

#ad-hoc profile

projPath=/Users/Shared/Jenkins/hcr/branch/branche\_ios\_18.6.0/proj\_ios

xcodebuild -project $projPath/Unity-iPhone.xcodeproj archive -scheme Unity-iPhone -configuration Release -destination generic/platform=iOS PROVISIONING\_PROFILE\_SPECIFIER=jdhcrADHOCdis CODE\_SIGN\_IDENTITY="iPhone Distribution: Shanghai Zhengju Information Technology Co., Ltd. (TG93K6XG5H)" -archivePath $projPath/Unity-iPhone.xcarchive





### ios\_branche\_05\_GenIPA\_AppStore

#!/bin/bash

projPath=/Users/Shared/Jenkins/hcr/branch/branche\_ios\_18.6.0/proj\_ios

xcodebuild -exportArchive -archivePath $projPath/Unity-iPhone.xcarchive -exportPath $projPath/out -exportOptionsPlist $projPath/../appstoreAdHoc.plist

cp $projPath/out/Unity-iPhone.ipa /Library/WebServer/Documents/jdhcr/ios/jdhcr-dis-$(date +%m%d%H%M%y).ipa

mv $projPath/out/Unity-iPhone.ipa /Library/WebServer/Documents/jdhcr/ios/jdhcr-dis-2019.ipa



## zttest(all green)

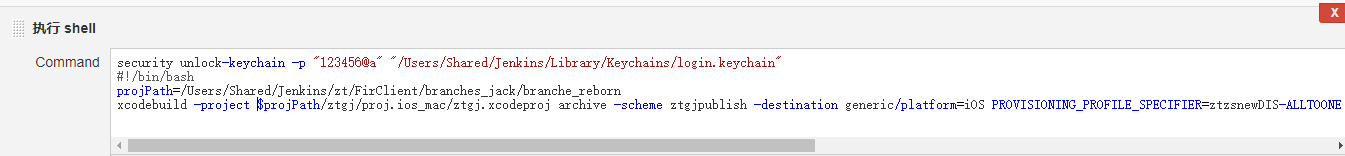
### ztapp\_ios\_002\_complie

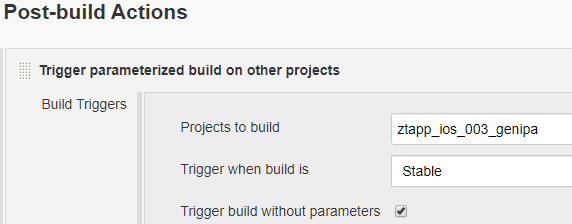
security unlock-keychain -p "123456@a" "/Users/Shared/Jenkins/Library/Keychains/login.keychain"

#!/bin/bash

projPath=/Users/Shared/Jenkins/zt/FirClient/branches\_jack/branche\_reborn

xcodebuild -project $projPath/ztgj/proj.ios\_mac/ztgj.xcodeproj archive -scheme ztgjpublish -destination generic/platform=iOS PROVISIONING\_PROFILE\_SPECIFIER=ztzsnewDIS-ALLTOONE CODE\_SIGN\_IDENTITY="iPhone Distribution: Shanghai Giant Network Technology Co., Ltd." -archivePath $projPath/Ztgj-iPhone.xcarchive



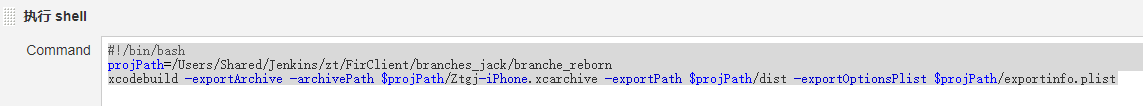


### ztapp\_ios\_003\_genipa

#!/bin/bash

projPath=/Users/Shared/Jenkins/zt/FirClient/branches\_jack/branche\_reborn

xcodebuild -exportArchive -archivePath $projPath/Ztgj-iPhone.xcarchive -exportPath $projPath/dist -exportOptionsPlist $projPath/exportinfo.plist



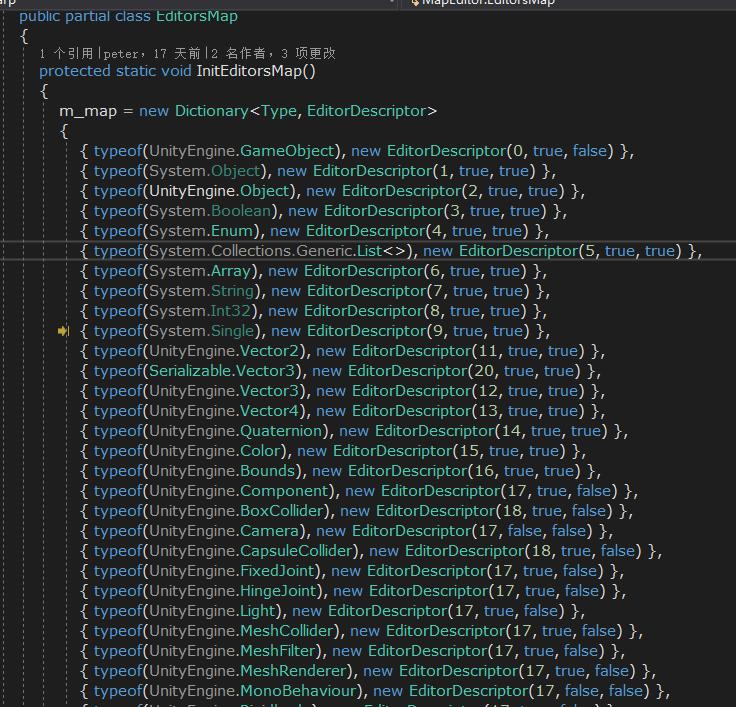
# Macpro

# 地编

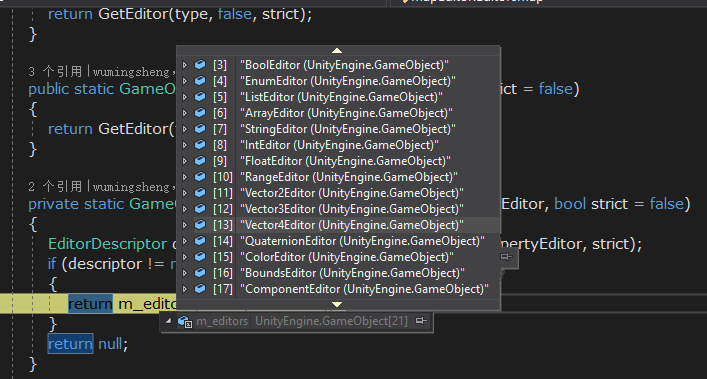
组合（搭积木）

## 属性编辑

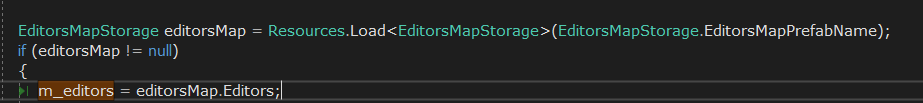
### 类型定义



### 编辑原型：



Resource下的MapEditorsMap保存了所有编辑原型



**ComponentEditor**

**//editor:根据模板新创建出的UI, ComponentsPanel:是gameObjectEditor下的专门用来挂组件UI的**

**editor**.transform.SetParent(ComponentsPanel, false);

**editor**.Component = component;



### 组件描述

# Reflection

## 获取属性：

PropertyInfo prop = type.GetProperty("enabled", BindingFlags.Public | BindingFlags.DeclaredOnly | BindingFlags.Instance);

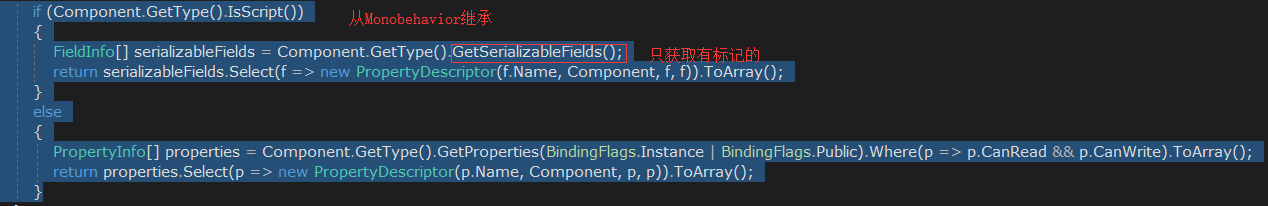
## 属性的类型：

prop.**PropertyType**

## 取得类型的名字：

Component.GetType().Name;

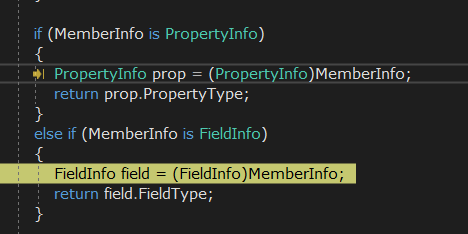
## 取得所有属性



## 判断子类：

type.IsSubclassOf(typeof(MonoBehaviour));

## 判断是PropertyInfo还是FieldInfo



## 根据类型取得所有可以序列号的字段

FieldInfo[] serializableFields = Reflection.**GetSerializableFields**(memberInfo.GetType());

## 根据filed取value

com:实体

field.GetValue(com)

## XXXX

**fi.GetType()**

**和**

**fi.FieldType**

**是两回事**