

目录

前言	1.1
iOSOpenDev概览	1.2
安装iOSOpenDev	1.3
确认安装成功	1.3.1
普通的插件开发流程	1.4
新建iOSOpenDev项目	1.4.1
初始化项目配置	1.4.2
ssh免密登录	1.4.2.1
写hook插件代码	1.4.3
如何新增文件	1.4.3.1
调试插件代码	1.4.4
带界面的插件开发流程	1.5
前提和目标	1.5.1
基本流程	1.5.2
使用效果	1.5.3
常见问题	1.6
初始化环境	1.6.1
编译调试	1.6.2
经验心得	1.7
附录	1.8
参考资料	1.8.1

iOS逆向开发：iOSOpenDev开发插件

- 最新版本： v1.2.0
- 更新时间： 20231025

简介

介绍iOS逆向中如何用iOSOpenDev开发越狱插件tweak。先是对比iOSOpenDev概览；然后介绍如何安装iOSOpenDev，以及安装后确认安装成功；然后是普通的插件的开发流程，包括新建iOSOpenDev的Xcode项目、初始化项目的配置，其中包括ssh免密登录、写hook插件tweak代码，包括如何新增文件、编译代码调试代码等；以及带UI界面的插件开发的流程，包括前提和目标、基本流程、使用效果、常见问题等；以及常见问题和一些经验心得；常见问题包括初始化环境方面的问题和编译调试方面的问题。

源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

HonKit源码

- [crifan/ios_re_iosopendev_tweak: iOS逆向开发：iOSOpenDev开发插件](#)

如何使用此HonKit源码去生成发布为电子书

详见：[crifan/honkit_template: demo how to use crifan honkit template and demo](#)

在线浏览

- [iOS逆向开发：iOSOpenDev开发插件 book.crifan.org](#)
- [iOS逆向开发：iOSOpenDev开发插件 crifan.github.io](#)

离线下载阅读

- [iOS逆向开发：iOSOpenDev开发插件 PDF](#)
- [iOS逆向开发：iOSOpenDev开发插件 ePUB](#)
- [iOS逆向开发：iOSOpenDev开发插件 Mobi](#)

版权和用途说明

此电子书教程的全部内容，如无特别说明，均为本人原创。其中部分内容参考自网络，均已备注了出处。如发现有侵权，请通过邮箱联系我 `admin 艾特 crifan.com`，我会尽快删除。谢谢合作。

各种技术类教程，仅作为学习和研究使用。请勿用于任何非法用途。如有非法用途，均与本人无关。

鸣谢

感谢我的老婆陈雪的包容理解和悉心照料，才使得我 crifan 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

其他

作者的其他电子书

本人 crifan 还写了其他 150+ 本电子书教程，感兴趣可移步至：

[crifan/crifan_ebook_readme: Crifan的电子书的使用说明](#)

关于作者

关于作者更多介绍，详见：

[关于CrifanLi李茂 – 在路上](#)

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：

2023-10-25 19:58:23

iOSOpenDev概览

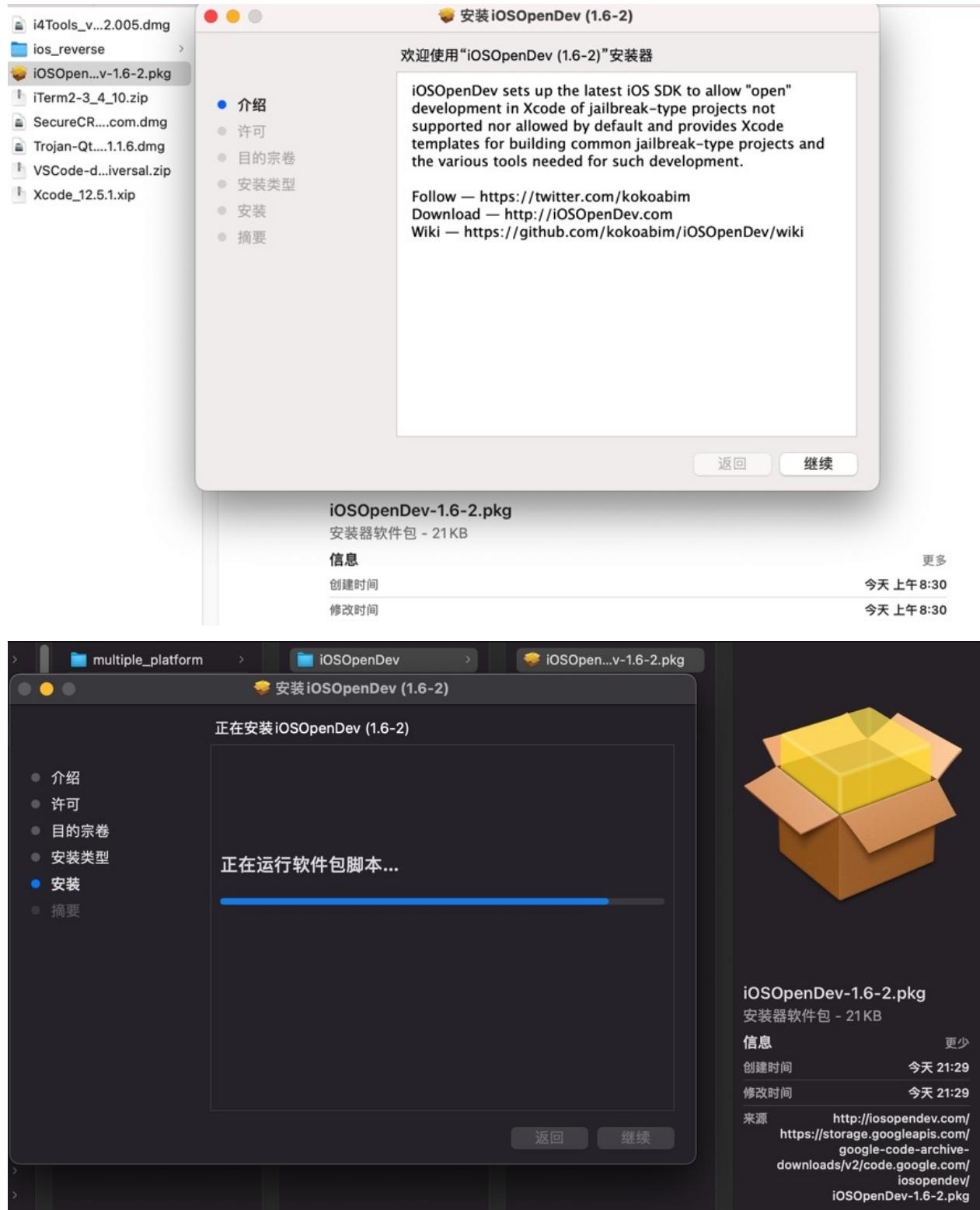
在iOS逆向期间，往往涉及到去[开发越狱插件tweak](#)，其中常见的工具=框架之一就是：`iOSOpenDev`。

- `iOSOpenDev`
 - 概述：iOS越狱插件tweak开发框架之一
 - 其他竞品
 - [Theos/Logos](#)
 - [MonkeyDev](#)
 - 常用核心功能：支持基于XCode（的模板）去创建Logos的tweak越狱插件
 - 官网
 - <http://iosopendev.com/>
 - 下载
 - <http://iosopendev.com/download/>
 - Github
 - <https://github.com/kokoabim/iOSOpenDev>
 - Wiki
 - <https://github.com/kokoabim/iOSOpenDev/wiki>
 - 对于Logos (=Theos?) 的支持
 - [https://github.com/kokoabim/iOSOpenDev/wiki/Logos-\(Theos\)-Support](https://github.com/kokoabim/iOSOpenDev/wiki/Logos-(Theos)-Support)
 - 项目转换
 - <https://github.com/kokoabim/iOSOpenDev/wiki/Convert-to-iOSOpenDev-Project>

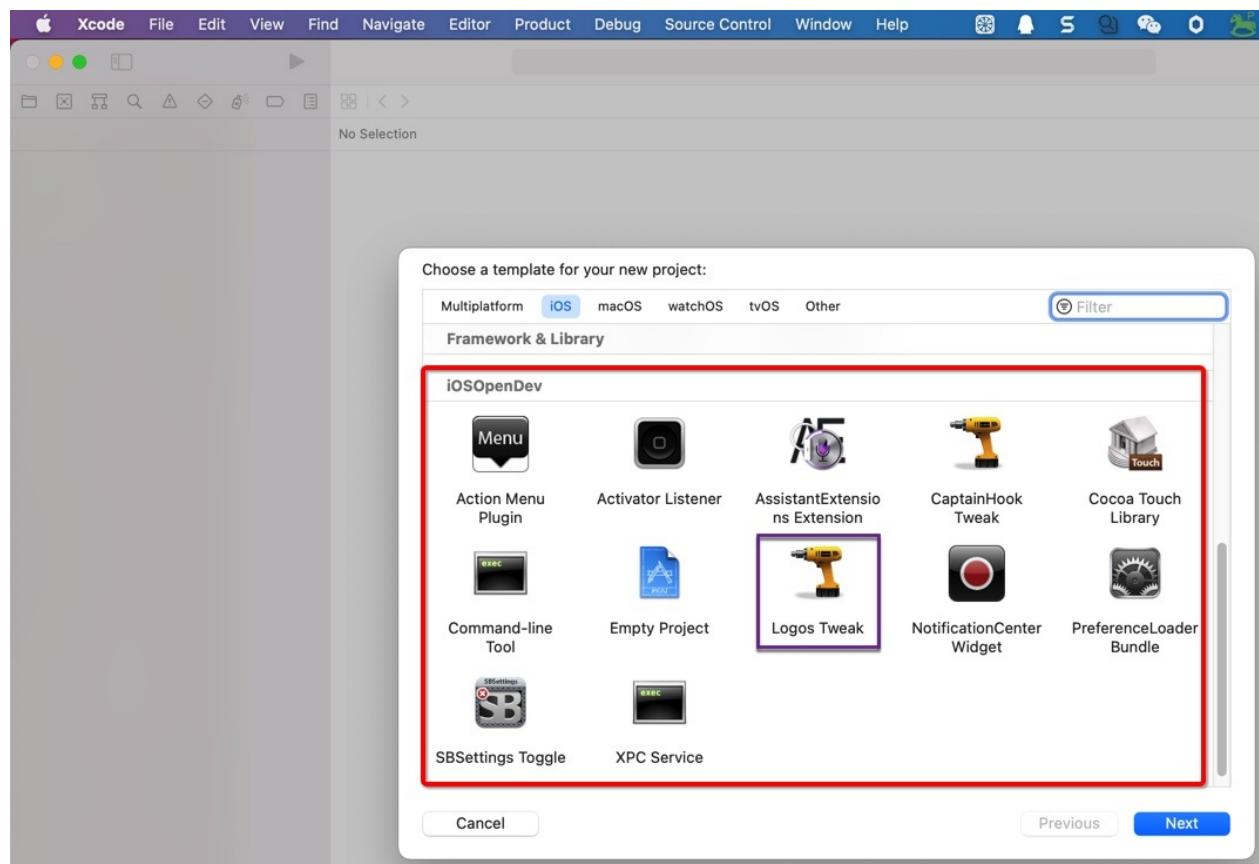
crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：
2023-10-25 19:55:34

安装iOSOpenDev

从官网[iOSOpenDev—Download](#)下载到: [iOSOpenDev-1.6-2.pkg](#), 双击去安装:



成功安装后, 去 Xcode 中新建 ios 项目, 即可看到 iOSOpenDev 的选项:



crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新:
2022-11-08 16:59:45

确认安装成功

环境变量

再去确认，是否把iOSOpenDev的相关环境变量，加到启动脚本（此处是 zsh，所以是 ./zshrc）中了：

```
→ ~ cat ~/.zshrc | grep iOSOpenDev
export iOSOpenDevPath /opt/iOSOpenDev
export iOSOpenDevDevice=
export PATH /opt/iOSOpenDev/bin:$PATH
```

如果没有：

```
x crifan@licrifandeMacBook-Pro ~ /opt/iOSOpenDevSetup/bin cat ~/.zshrc | grep iOSOpenDe
v
```

则自己手动去加上：

```
crifan@licrifandeMacBook-Pro ~ /opt/iOSOpenDevSetup/bin vi ~/.zshrc
crifan@licrifandeMacBook-Pro ~ /opt/iOSOpenDevSetup/bin cat ~/.zshrc | grep iOSOpenDev
export iOSOpenDevPath /opt/iOSOpenDev
export iOSOpenDevDevice=
export PATH /opt/iOSOpenDev/bin:$PATH
crifan@licrifandeMacBook-Pro ~ /opt/iOSOpenDevSetup/bin source ~/.zshrc
```

Xcode中的iOSOpenDev的模板

确认是否有多出的template模板：

```
→ ~ ll ~/Library/Developer/Xcode/
total 0
drwxr-xr-x 8 crifan staff 256B 10 14 11:13 DerivedData
srwxr-xr-x 1 crifan staff 0B 10 27 08:54 GPUToolsAgent.sock
drwxr-xr-x 3 crifan staff 96B 10 27 08:49 Templates
drwxr-xr-x 6 crifan staff 192B 10 26 22:37 UserData
drwxr-xr-x 5 crifan staff 160B 9 30 22:11 iOS Device Logs
drwxr-xr-x 4 crifan staff 128B 10 13 13:54 iOS DeviceSupport
→ ~ ll ~/Library/Developer/Xcode/Templates
total 0
lrwxr-xr-x 1 root staff 25B 10 27 08:49 iOSOpenDev -> /opt/iOSOpenDev/templates
```

此处是有的：

- 多出了软链接：
 - ~/Library/Developer/Xcode/Templates/iOSOpenDev
 - 指向的是：

```
■ /opt/iOSOpenDev/templates
```

以及接着去看看，具体有哪些模板：

```
→ ~ ll /opt/iOSOpenDev/templates
total 48
drwxr-xr-x  5 root  wheel  160B 10 27 08:32 Action Menu Plugin.xctemplate
drwxr-xr-x  6 root  wheel  192B 10 27 08:32 Activator Listener.xctemplate
drwxr-xr-x 12 root  wheel  384B 10 27 08:32 AssistantExtensions Extension.xctemplate
drwxr-xr-x  4 root  wheel  128B 10 27 08:32 Base.xctemplate
drwxr-xr-x  6 root  wheel  192B 10 27 08:32 CaptainHook Tweak.xctemplate
drwxr-xr-x  6 root  wheel  192B 10 27 08:32 Cocoa Touch Library.xctemplate
drwxr-xr-x  5 root  wheel  160B 10 27 08:32 Command-line Tool.xctemplate
drwxr-xr-x  4 root  wheel  128B 10 27 08:32 Debian Package.xctemplate
drwxr-xr-x  4 root  wheel  128B 10 27 08:32 Empty Project.xctemplate
drwxr-xr-x  -rw-r--r-- 1 root  wheel   18K 10 27 08:49 LICENSE
drwxr-xr-x  6 root  wheel  192B 10 27 08:32 Logos Tweak.xctemplate
drwxr-xr-x  5 root  wheel  160B 10 27 08:32 ManPage.xctemplate
drwxr-xr-x 11 root  wheel  352B 10 27 08:32 NotificationCenter Widget.xctemplate
drwxr-xr-x 12 root  wheel  384B 10 27 08:32 PreferenceLoader Bundle.xctemplate
drwxr-xr-x  7 root  wheel  224B 10 27 08:32 PreferenceLoader.xctemplate
drwxr-xr-x  -rw-r--r-- 1 root  wheel  352B 10 27 08:49 README.md
drwxr-xr-x  5 root  wheel  160B 10 27 08:32 SBSettings Toggle.xctemplate
drwxr-xr-x  4 root  wheel  128B 10 27 08:32 Unit Tests.xctemplate
drwxr-xr-x  7 root  wheel  224B 10 27 08:32 XPC Service.xctemplate
```

很明显，部分模板，应该就是对应着界面中看到的各个模板：

比如：

- Logos Tweak.xctemplate -> Logos Tweak
- Command-line Tool.xctemplate -> Command-line Tool
- PreferenceLoader Bundle.xctemplate -> PreferenceLoader Bundle

iOSOpenDev中的内容

顺带再去看看，当前iOSOpenDev目录中的内容：

```
→ /opt ll
total 0
drwxr-xr-x  9 root  wheel  288B 10 27 08:32 iOSOpenDev
drwxr-xr-x  3 root  wheel   96B 10 27 08:32 iOSOpenDevSetup
drwxr-xr-x  3 root  wheel   96B 10 27 08:32 iOSOpenDevUninstall
→ /opt cd iOSOpenDev
→ iOSOpenDev pwd
/opt/iOSOpenDev
→ iOSOpenDev ll
total 48
-rw-r--r--  1 root  wheel   18K 10 27 08:49 LICENSE
-rw-r--r--  1 root  wheel  352B 10 27 08:49 README.md
drwxr-xr-x  6 root  wheel  192B 10 27 08:32 bin
drwxr-xr-x  2 root  wheel   64B 10 27 08:32 frameworks
drwxr-xr-x  8 root  wheel  256B 10 27 08:32 include
```

```

drwxr-xr-x  5 root  wheel  160B 10 27 08:32 lib
drwxr-xr-x 21 root  wheel  672B 10 27 08:32 templates
→ iOSOpenDev 11 bin
total 3000
-rw xr-xr-x 1 root  wheel  428K 10 27 08:49 class-dump
-rw xr-xr-x 1 root  wheel  628K 10 27 08:49 class-dump-z
-rw xr-xr-x 1 root  wheel   59K 10 27 08:49 iosod
-rw xr-xr-x 1 root  wheel  383K 10 27 08:49 ldid
→ iOSOpenDev 11 frameworks
→ iOSOpenDev 11 include
total 48
drwxr-xr-x  3 root  wheel   96B 10 27 08:32 ActionMenu
drwxr-xr-x  3 root  wheel   96B 10 27 08:32 AssistantExtensions
drwxr-xr-x  3 root  wheel   96B 10 27 08:32 CaptainHook
drwxr-xr-x 10 root  wheel  320B 10 27 08:32 libactivator
drwxr-xr-x  3 root  wheel   96B 10 27 08:32 logos
-rw-r--r--  1 root  wheel   21K 10 27 08:49 substrate.h
→ iOSOpenDev 11 lib
total 1216
-rw xr-xr-x 1 root  wheel   77K 10 27 08:49 libactionmenu.dylib
-rw xr-xr-x 1 root  wheel  422K 10 27 08:49 libactivator.dylib
-rw xr-xr-x 1 root  wheel  101K 10 27 08:49 libsubstrate.dylib
→ iOSOpenDev 11 templates
total 48
drwxr-xr-x  5 root  wheel  160B 10 27 08:32 Action Menu Plugin.xctemplate
drwxr-xr-x  6 root  wheel  192B 10 27 08:32 Activator Listener.xctemplate
drwxr-xr-x 12 root  wheel  384B 10 27 08:32 AssistantExtensions Extension.xctemplate
drwxr-xr-x  4 root  wheel  128B 10 27 08:32 Base.xctemplate
drwxr-xr-x  6 root  wheel  192B 10 27 08:32 CaptainHook Tweak.xctemplate
drwxr-xr-x  6 root  wheel  192B 10 27 08:32 Cocoa Touch Library.xctemplate
drwxr-xr-x  5 root  wheel  160B 10 27 08:32 Command-line Tool.xctemplate
drwxr-xr-x  4 root  wheel  128B 10 27 08:32 Debian Package.xctemplate
drwxr-xr-x  4 root  wheel  128B 10 27 08:32 Empty Project.xctemplate
-rw-r--r--  1 root  wheel   18K 10 27 08:49 LICENSE
drwxr-xr-x  6 root  wheel  192B 10 27 08:32 Logos Tweak.xctemplate
drwxr-xr-x  5 root  wheel  160B 10 27 08:32 ManPage.xctemplate
drwxr-xr-x 11 root  wheel  352B 10 27 08:32 NotificationCenter Widget.xctemplate
drwxr-xr-x 12 root  wheel  384B 10 27 08:32 PreferenceLoader Bundle.xctemplate
drwxr-xr-x  7 root  wheel  224B 10 27 08:32 PreferenceLoader.xctemplate
-rw-r--r--  1 root  wheel  352B 10 27 08:49 README.md
drwxr-xr-x  5 root  wheel  160B 10 27 08:32 SBSettings Toggle.xctemplate
drwxr-xr-x  4 root  wheel  128B 10 27 08:32 Unit Tests.xctemplate
drwxr-xr-x  7 root  wheel  224B 10 27 08:32 XPC Service.xctemplate

```

普通的插件开发流程

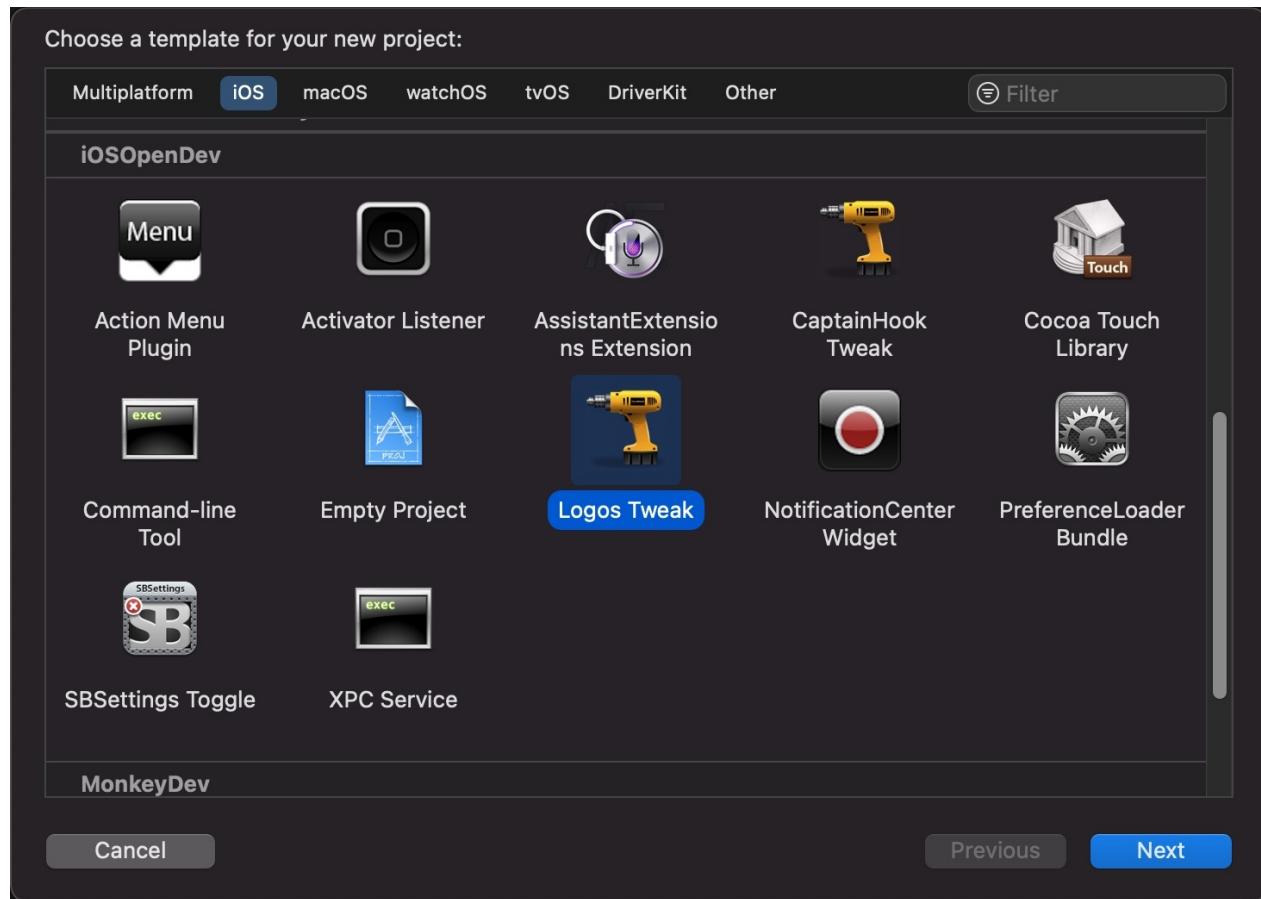
TODO:

- 【已解决】Mac中用iOSOpenDev开发iOS的theos的Logos的tweak插件
 - 【已解决】给iOSOpenDev的Logos的tweak的XCode项目去做基本配置
-

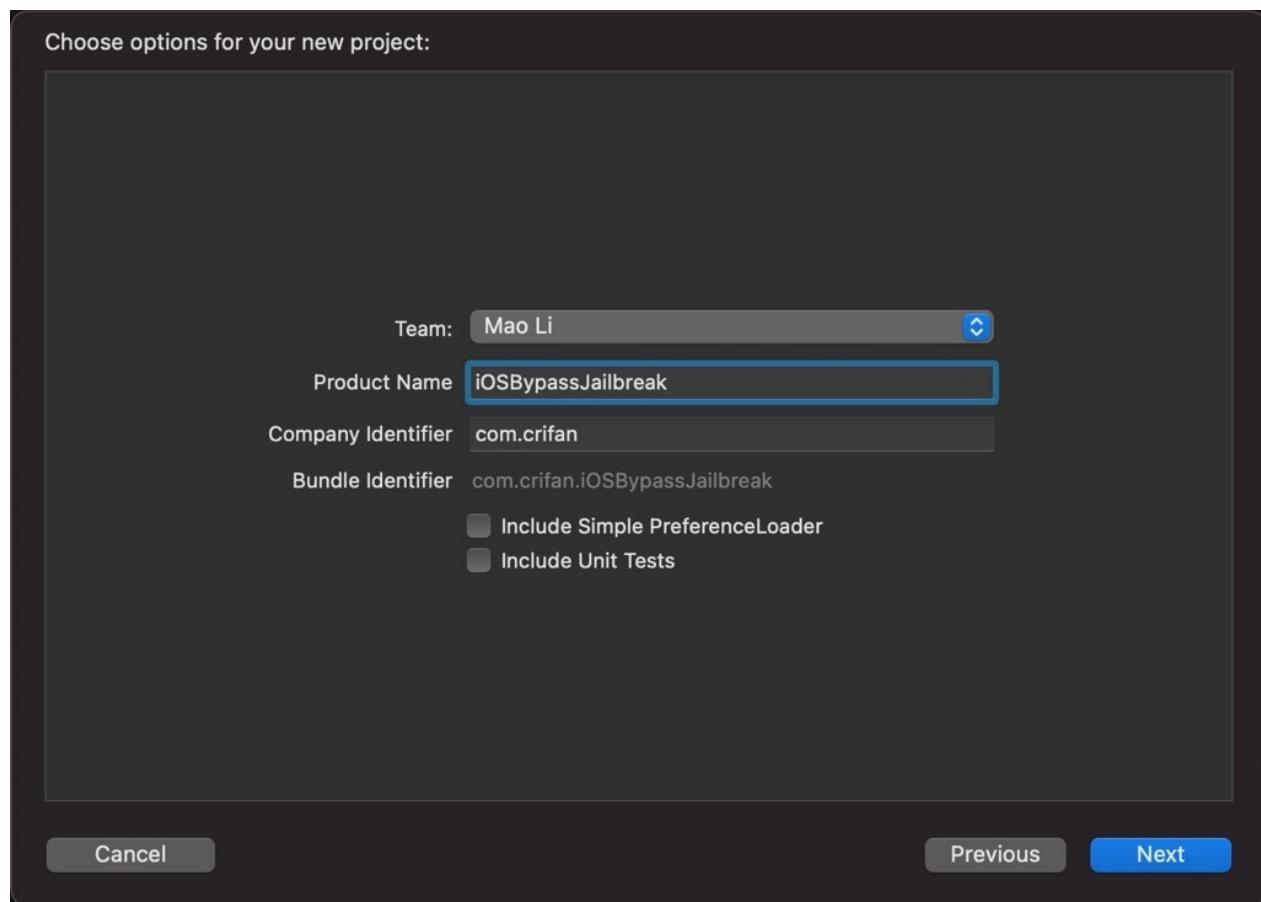
crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新:
2022-11-08 17:03:23

新建iOSOpenDev的Xcode项目

Xcode 中新建 iOS 项目，选择： iOSOpenDev -> Logos Tweak :



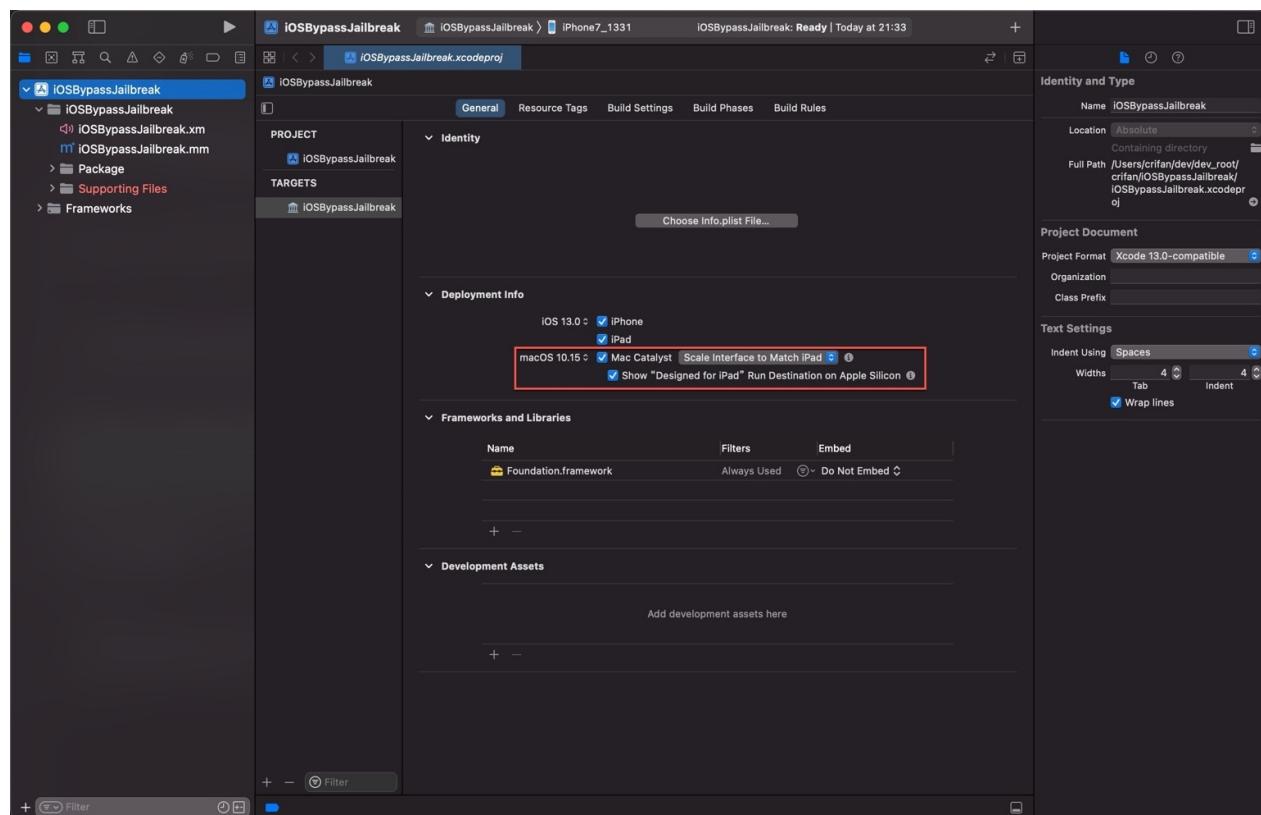
然后填写项目信息：



比如*iOSBypassJailbreak*的：

- Product Name : iOSBypassJailbreak
- Company Bundle : com.crifan
- Bundle Identifier : 自动生成出 com.crifan.iOSBypassJailbreak

点击 `Next` 继续，即可新建出，看起来和普通 `Xcode` 没多大区别的项目：



crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新:
2022-11-08 16:05:21

初始化配置iOSOpenDev的Xcode项目

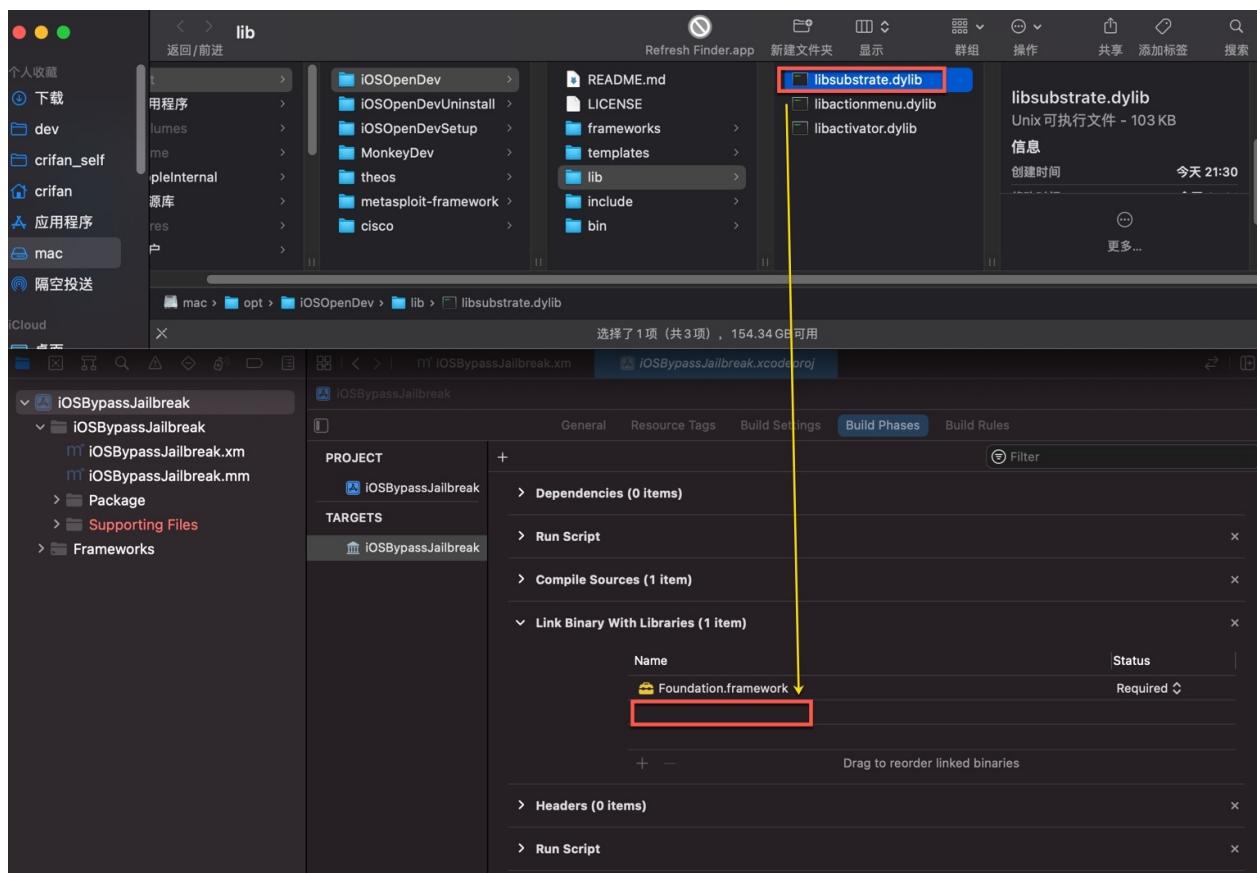
去掉 Deployment Info 中的 Mac 和确保iOS版本一致

此处，先去做第一个配置方面的改动：

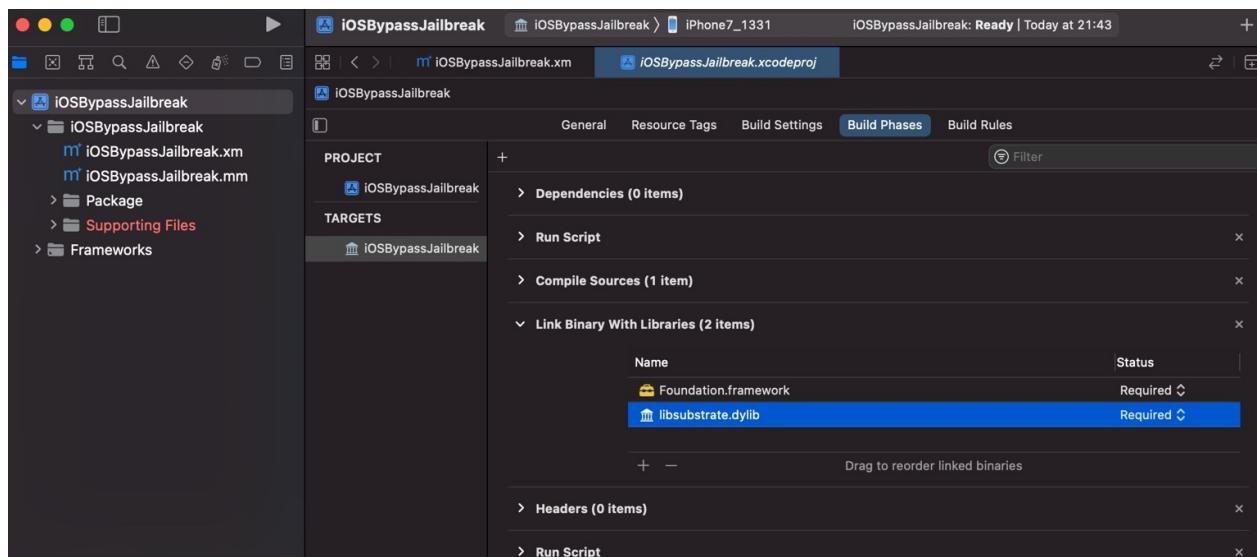
- TARGETS -> General -> Deployment Info : 去掉 Mac , 因为我们开发的是 iOS 的插件，不需要发布到 Mac , 且设置好iOS的最低版本
 -
- PROJECT -> Info -> Deployment Target -> iOS Deployment Target , 也要同步设置 iOS 的最低版本
 -
-

导入依赖库 libsubstrate.dylib

把 libsubstrate.dylib (一般在 /opt/iOSOpenDev/lib/libsubstrate.dylib):



导入到项目中的： Targets -> YourProjectName -> Build Phases -> Link Binary With Libraries



设置被hook的app包名或二进制文件名

去把要hook的，被拦截的app的包名，加到被hook的包名的列表中：

```
YourProjectName -> YourProjectName -> Package -> Libarary -> MobileSubstrate -> DynamicLibraries -> CurrentProjectBundleIdentifier.plist
```

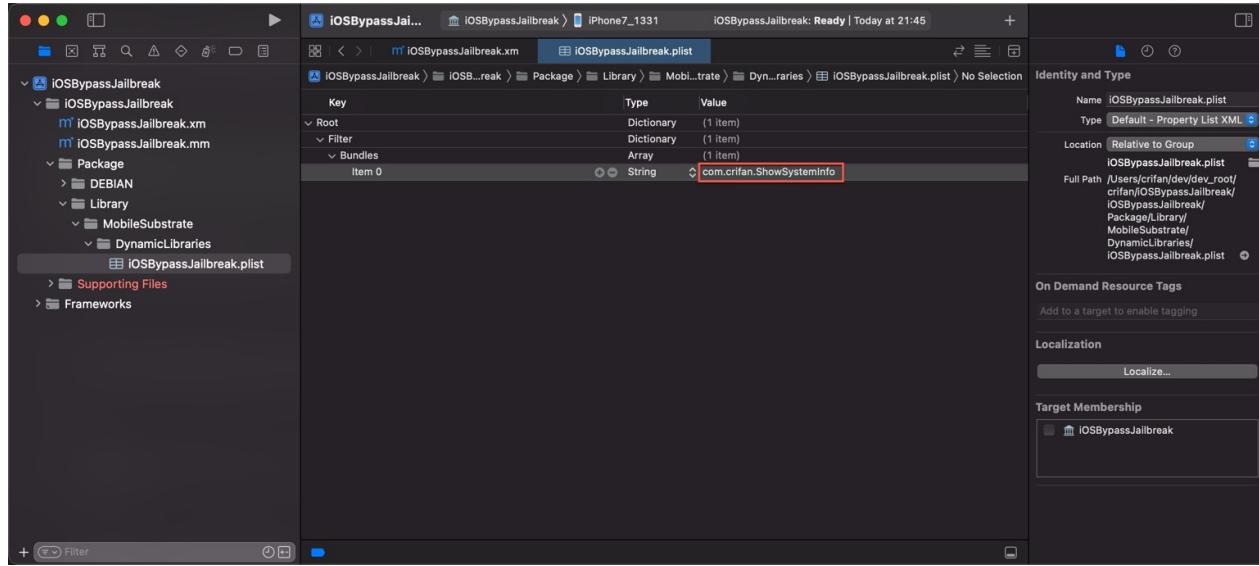
在 Root -> Filter -> Bundles，会看到 Item 0：

- Type : String

- `Value` : 填入你要hook的app的包名

- 举例

- `com.crifan.ShowSystemInfo`



- 另外

- 如果要新增一行

- 移动到 `Item 0` 所在的行，会看到出现个 `= 加号`，点击 `加号`，会新增一行

如何hook二进制？

如果需要hook二进制，则是新建 `Array` 类型的（和 `Bundles` 并列的）`Executables` 子项，再加上对应二进制文件名

举例：

- hook二进制：`akd = AuthKit.framework` 的daemon进程

◦

此时，对应的 `plist` 文件内容是：

- `jailAppleAccount/Package/Library/MobileSubstrate/DynamicLibraries/jailAppleAccount.plist`

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Filter</key>
    <dict>
        <key>Executables</key>
        <array>
            <string>akd</string>
            <string>amsaccounts</string>
            <string>AppleMediaServices</string>
            <string>AppleAccount</string>
            <string>Preferences</string>
        </array>
        <key>Bundles</key>
        <array>
            <string>com.apple.Preferences</string>
        </array>
    </dict>
</dict>
</plist>
```

设置iPhone的IP

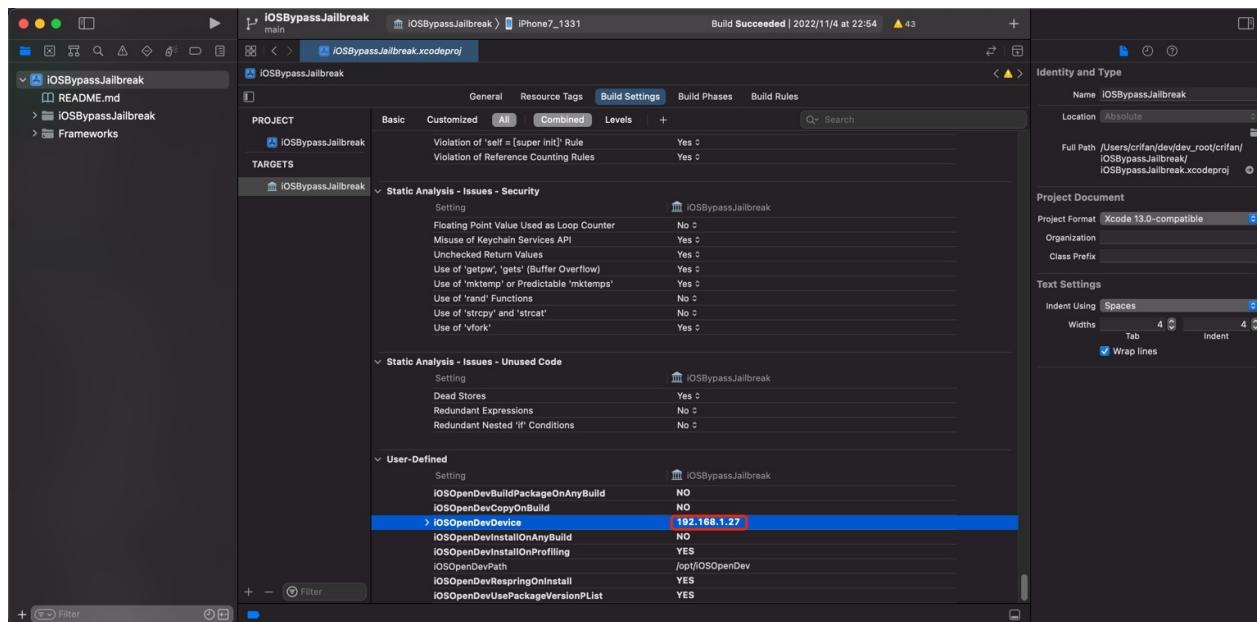
把此处要把iOS调试设备（iPhone）中的IP地址：



此处是：`192.168.1.27`

去加到配置中去：

- `iOSOpenDevDevice = 192.168.1.27`



附上原默认的更新后的配置：

```
iOSOpenDevBuildPackageOnAnyBuild = NO
iOSOpenDevCopyOnBuild = NO
iOSOpenDevDevice = 192.168.1.27
iOSOpenDevInstallOnAnyBuild = NO
iOSOpenDevInstallOnProfiling = YES
iOSOpenDevPath = /opt/iOSOpenDev
iOSOpenDevRespringOnInstall = YES
iOSOpenDevUsePackageVersionPList = YES
```

另外，理论上，去把对应变量加到环境变量：

```
→ ~ cat ~/.zshrc | grep iOSOpenDevDevice
export iOSOpenDevDevice:192.168.1.27
```

效果应该也是一样的。

crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新：
2023-07-20 14:48:30

ssh免密登录

在 `xcode + iOSOpenDev` 的编译安装最后阶段，涉及到，自动通过ssh访问iPhone设备，把生成的 `.deb` 插件的文件下载和安装到iPhone中

此时就需要先准备好环境：确保 Mac 中可以，ssh的免密登录iPhone

此处ssh免密登录的具体步骤是：

- 先用ssh登录一次iPhone

- 命令

```
ssh root@192.168.1.27
```

- 输入密码

- OpenSSH 的默认密码是： alpine

- 即可登录到iPhone中

- 把ssh的key拷贝到iPhone中

- 命令

```
ssh-copy-id root@192.168.1.27
```

- 输入密码： alpine

即可实现，ssh免密登录：

以后ssh直接可以访问iPhone，而无需输入密码

crifan.org，使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：

2023-07-20 14:38:39

写hook插件代码

iOSOpenDev中的hook插件代码的逻辑是：

- `.xm` : 原始的hook插件的代码
 - 写hook插件，是改动 `.xm` 文件
 - 而不需要，也不应该改动 `.mm` 文件
- `.mm` : 从 `.xm` 自动（在 Build后）自动生成的文件
 - 后续真正编译的文件是 `.mm` 文件
 - 可以在 `Compiled Sources` 中看到 `.mm` 文件（而不是 `.xm` 文件）

新建 iOSOpenDev 的项目中的 `.xm` 文件（此处是 `iOSBypassJailbreak.xm`）生成的默认代码，来自模板，一般是：

```
// Logos by Dustin Howett
// See http://iphonedevwiki.net/index.php/Logos

#error iOSOpenDev post project creation from template requirements (remove these lines
after completed) -- \
Link to libsubstrate.dylib: \
(1) go to TARGETS > Build Phases > Link Binary With Libraries and add /opt/iOSOpenDev
/lib/libsubstrate.dylib \
(2) remove these lines from *.xm files (not *.mm files as they're automatically gene
rated from *.xm files)

@hook ClassName

+ (id)sharedInstance
{
    %log;

    return %orig;
}

- (void)messageWithNoReturnAndOneArgument (id)originalArgument
{
    %log;

    %orig(originalArgument);

    // or, for example, you could use a custom value instead of the original argument: %
    orig(customValue);
}

- (id)messageWithReturnAndNoArguments
{
    %log;

    id originalReturnOfMessage = %orig;

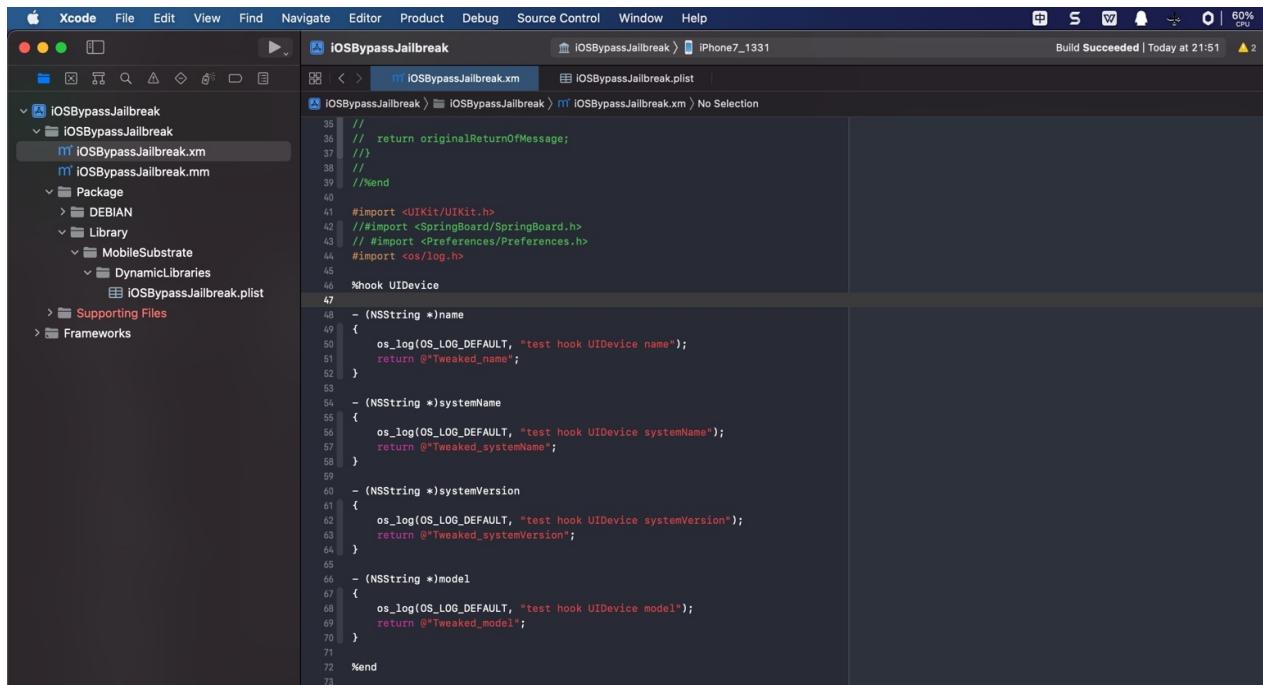
    // for example, you could modify the original return value before returning it: [Som
    eOtherClass doSomethingToThisObject:originalReturnOfMessage];
}
```

```
    return originalReturnOfMessage;
}

end
```

去删除掉，或注释掉，改用自己的hook的代码。

比如此处仅用于演示的代码：



The screenshot shows the Xcode interface with the project 'iOSBypassJailbreak' open. The left sidebar displays the project structure, including 'iOSBypassJailbreak', 'Supporting Files', and 'Frameworks'. The main editor area shows the 'iOSBypassJailbreak.xm' file content:

```
//import <UIKit/UIKit.h>
//#import <SpringBoard/SpringBoard.h>
// #import <Preferences/Preferences.h>
#import <os/log.h>

%hook UIDevice

- (NSString *)name
{
    os_log(OS_LOG_DEFAULT, "test hook UIDevice name");
    return @"Tweaked_name";
}

- (NSString *)systemName
{
    os_log(OS_LOG_DEFAULT, "test hook UIDevice systemName");
    return @"Tweaked_systemName";
}

- (NSString *)systemVersion
{
    os_log(OS_LOG_DEFAULT, "test hook UIDevice systemVersion");
    return @"Tweaked_systemVersion";
}

- (NSString *)model
{
    os_log(OS_LOG_DEFAULT, "test hook UIDevice model");
    return @"Tweaked_model";
}

%end
```

```
#import <UIKit/UIKit.h>
//#import <SpringBoard/SpringBoard.h>
// #import <Preferences/Preferences.h>
#import <os/log.h>

%hook UIDevice

- (NSString *)name
{
    os_log(OS_LOG_DEFAULT, "test hook UIDevice name");
    return @"Tweaked_name";
}

- (NSString *)systemName
{
    os_log(OS_LOG_DEFAULT, "test hook UIDevice systemName");
    return @"Tweaked_systemName";
}

- (NSString *)systemVersion
{
    os_log(OS_LOG_DEFAULT, "test hook UIDevice systemVersion");
    return @"Tweaked_systemVersion";
}
```

```
- (NSString *)model
{
    os_log(OS_LOG_DEFAULT, "test hook UIDevice model");
    return @"Tweaked_model";
}

@end
```

附录

iOS中常见的hook网络相关的请求

核心代码:

```
static char* LastUpdate = "20231025_1524";

#import "HookLogiOS.h"
#import "CrifanLib.h"

/*
----- NSURLConnection -----
----- */

hook NSURLConnection

+(instancetype)requestWithURL:(NSURL *)URL cachePolicy:(NSURLRequestCachePolicy)cachePolicy timeoutInterval:(NSTimeInterval)timeoutInterval{
    iosLogInfo("URL=%{public}@, cachePolicy=%lu, timeoutInterval=%f", URL, (unsigned long)cachePolicy, timeoutInterval);
    return &orig;
}

+(instancetype)requestWithURL:(NSURL *)URL{
    iosLogInfo("URL=%{public}@", URL);
    return &orig;
}

end

/*
----- NSHTTPURLResponse -----
----- */

hook NSHTTPURLResponse

-(NSHTTPURLResponse*)initWithURL:(NSURL *)url statusCode:(NSInteger)statusCode HTTPVersion:(NSString *)HTTPVersion headerFields:(NSDictionary< NSString *, NSString *> *)headerFields{
    NSHTTPURLResponse* newUrlResp = &orig;
    iosLogInfo("url=%{public}@, statusCode=%ld, HTTPVersion=%@, headerFields=%{public}@ -> newUrlResp=%{public}@", url, statusCode, HTTPVersion, headerFields, newUrlResp);
    return newUrlResp;
}
```

```

}

- (NSDictionary *)allHeaderFields{
    NSURL* curUrl = [self URL];
    NSDictionary* allHeader = %orig;

    //    iosLogInfo("curUrl=%{public}@ : allHeader=%{public}@", curUrl, allHeader);
    NSString* respUrlHeaderStr = [NSString stringWithFormat:@"NSHTTPURLResponse:allHeaderFields curUrl=%@ : allHeader=%@", curUrl, allHeader];
    logPossibleLargeStr(respUrlHeaderStr);

    return allHeader;
}

- (NSInteger)statusCode{
    NSURL* curUrl = [self URL];
    NSInteger respStatusCode = %orig;
    iosLogInfo("respStatusCode=%ld : curUrl=%{public}@", respStatusCode, curUrl);
    return respStatusCode;
}

end

/*=====
 *ctor
 =====*/

```

```

*ctor {
    iosLogInfo("%s: %s", LastUpdate, "HookWhatsApp ctor");
}

```

- 注:

- 调用的 `iosLogInfo` 等函数, 详见:
 - <https://github.com/crifan/crifanLib/blob/master/c/CrifanLib.h>
 - <https://github.com/crifan/crifanLib/blob/master/iOS/HookLogiOS.h>

如何新增(.xm 和 .mm)文件

有时候，需要去新增文件： .xm 和 .mm

具体步骤是：

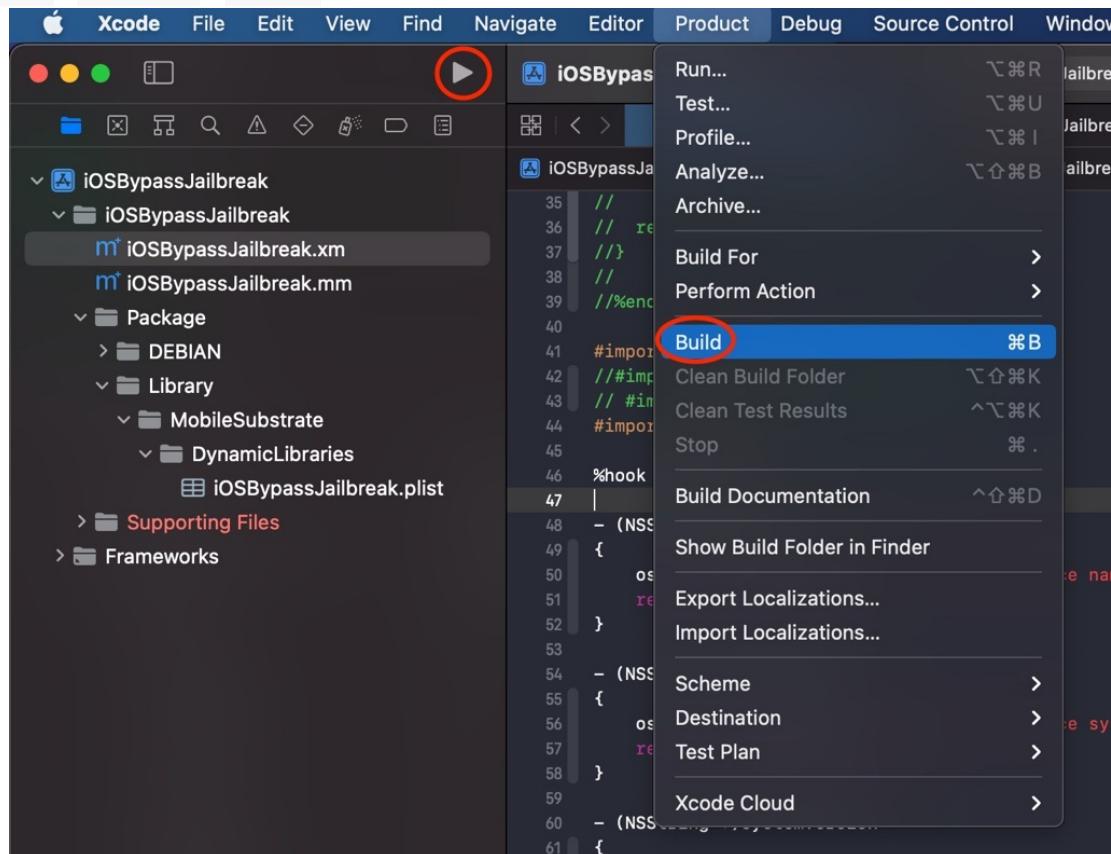
- 新建 .xm 文件
 - 选中要新增文件所属的位置 -> 右键 -> Add File -> iOS -> Other -> Empty -> 输入文件名: yourFilename.xm -> Create
- 编译 -> 会生成对应 .mm 文件
 - Product -> Build
 - 会从 yourFilename.xm 生成 yourFilename.mm
- 把 .mm 文件加到 compile Sources 中
 - 右键-> Add Files to {yourProjectName} -> 选择 (刚新生成的) yourFilename.mm
 - 项目文件列表中，即可新增对应文件 yourFilename.mm
 - 项目的待编译的文件中，也包含了对应的 .mm 文件
 - Targets -> Build Phase -> Compile Sources 中有了刚加入的 .mm 文件
 - 如果文件，点击 加号 = ，去新增导入进来
 - 这样后续编译代码时，才能真正编译到对应hook代码

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：
2023-07-20 14:50:35

调试插件代码

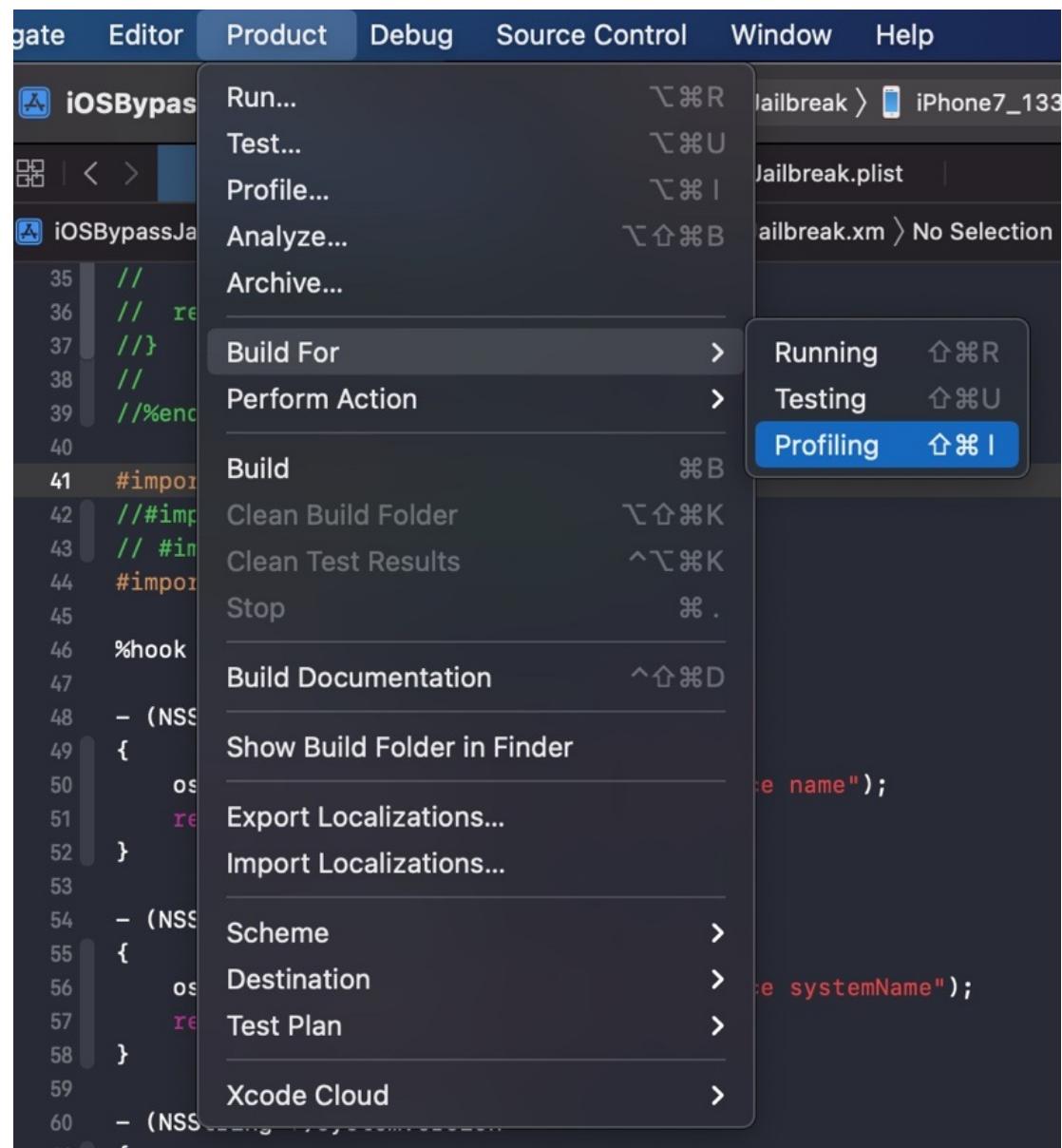
接下来就是典型的插件开发流程了：

- 写代码 = 写hook代码= 写tweak插件代码 = 改动 .xm 文件
- 编译代码 -> 确保语法没错，可以正常编译
 - Xcode -> Product -> Build



- 调试和运行 -> 把hook插件代码编译所生成的插件(.deb 文件)安装到iOS设备(iPhone)中，测试插件效果

- Product -> Build For -> Profiling



确认插件安装成功

- iPhone中看到自己的插件
 - Cydia -> 已安装 -> 最近 能看到自己的插件:



- 点击插件，可以看到插件基本信息



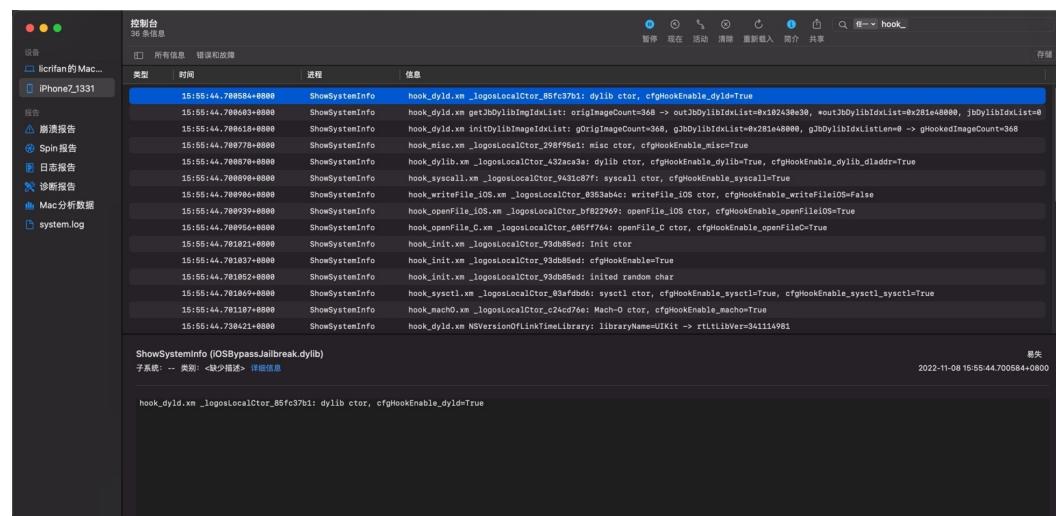
- 点击插件的文件，可以看到文件列表



确认插件的确正常工作

- 打开被测试的=被hook的app，看到此处测试代码：更改信息信息，显示是我们hook代码中的值，表示hook成功

-
- 查看对应log日志
 - Xcode -> Window -> Devices and Simulators -> Devices ->选中 Connected 中自己的 iPhone设备-> Open Console ->打开 Console = 控制台，显示出对应iPhone的log日志
 - 其中就有你的插件的log日志
 - 如果没有，则自己去右上角，搜索对应关键字，即可搜到
 - 此处贴出，后续更新了代码后的相关log



crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新:
2023-07-20 14:39:40

带界面的插件开发流程

TODO:

- 【未解决】用XCode开发iOS的app用于配置改机软件参数
 - 【已解决】越狱iPhone中安装deb包后iOS的反越狱插件没生效
 - 【已解决】把iOS的app和iOS的tweak插件打包成独立的deb安装包
 - 【记录】越狱iPhone中安装iOS的app和tweak合并出的deb安装包
 - 【已解决】越狱iOS如何用Theos开发带GUI图形界面的插件
 - 【记录】重新给iOSOpenDev的tweak加app打包deb看看app是否有权限写入Preferences目录
 - 【已解决】iOS的writeToURL报错： NSCocoaErrorDomain Code 513 You don't have permission to save the file in the folder Preferences
 - 【已解决】iOSOpenDev的tweak中读取app保存出的配置文件参数
 - 【已解决】iOSOpenDev的XCode编译报错： An empty identity is not valid when signing a binary for the product type Application
 - 【已解决】对比研究FakeWeChatLoc和自己的XCode项目的目录结构区别
 - 【已解决】给iOS的XCode项目中新增iOSOpenDev的Project Navigator的目录和文件
-

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：
2023-01-06 22:40:32

前提和目标

- 前提
 - 已实现：基于XCode用iOSOpenDev的Logos Tweak去创建出tweak插件
 - 已实现：用XCode开发出普通的带UI界面的iOS的app
 - 比如：实现了机型选择的功能
- 目标
 - 和tweak和app合并成单个deb安装包=单个tweak插件（安装出来后，带UI界面的tweak插件）

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：
2023-01-06 22:36:06

基本流程

TODO:

- 【已解决】XCode 13.1中Build Settings中Code Signing Identity没有Don't Sign Code选项
 - 【已解决】XCode中编译设置参数中如何新增User-Defined自定义参数
 - 【已解决】XCode项目中新增iOSOpenDev的Package目录到Target目录中
-

把Xcode中iOS的app项目转换成iOSOpenDev的项目

把普通的XCode项目，去改造成iOSOpenDev的项目：

核心步骤：

只需参考官网教程：

[Convert to iOSOpenDev Project · kokoabim/iOSOpenDev Wiki \(github.com\)](#)

只是有几个细节，需要更新和补充：

最新XCode（13.1）中：Code Signing Identity没有Don't Sign Code选项

解决办法：给 PROJECT -> Build Settings -> User-Defined ->增加参数：

- CODE_SIGNING_ALLOWED=NO

详见：

- 【已解决】XCode 13.1中Build Settings中Code Signing Identity没有Don't Sign Code选项

新版XCode中找不到新建User-Defined参数的入口

解决办法：选择 PROJECT （和或 TARGETS 中的某个 target ）-> Build Settings -» 最顶部
(和 Basic 、 Customized 、 All 、 Combined 所在的同) 一行的最右边有个加号 -» Add User-Defined Setting

详见：

- 【已解决】XCode中编译设置参数中如何新增User-Defined自定义参数

给Target的目录中新建Package

要点：选中自己项目的 Target 目录-» 右键-» New Group

即可新建组=Group=子目录

详见：

- 【已解决】XCode项目中新增iOSOpenDev的Package目录到Target目录中

app和tweak之间的通信

对于iOSOpenDev的app，想要和tweak插件之间通信，主要是互相共享配置参数，此处是通过：配置文件
具体做法是：

app端

MuJiaBaiHuoApp的ViewController.m

写入配置：

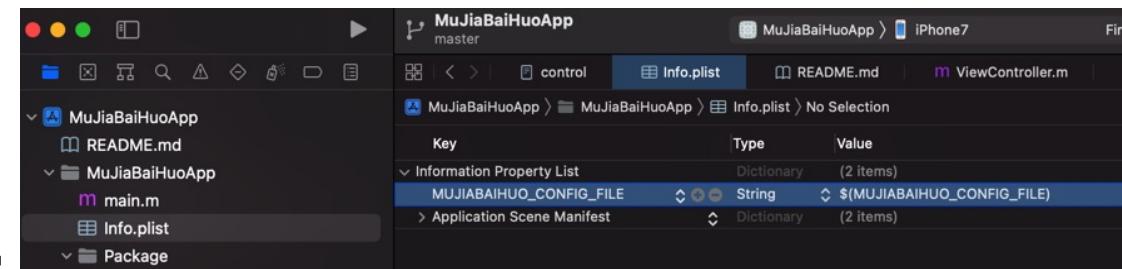
```
- (void) saveConfig:(NSDictionary *) curCfgDict {
    NSLog(@"%@", curCfgDict);
    NSString *curCfgFile = [[NSBundle mainBundle] objectForKey:@"MUJIABAI_HUO_CONFIG_FILE"];
    NSLog(@"%@", curCfgFile);
    NSString *curCfgFileUrl = [NSString stringWithFormat:@"file://%@", curCfgFile];
    NSLog(@"%@", curCfgFileUrl);
    NSURL *curCfgUrl = [NSURL URLWithString:curCfgFileUrl];
    NSLog(@"%@", curCfgUrl);
    NSError *writeErr = NULL;
    BOOL isWriteOk = [curCfgDict writeToURL:curCfgUrl error:&writeErr];
    NSLog(@"isWriteOk=%d", isWriteOk);
    if (isWriteOk == NO){
        NSLog(@"%@", writeErr);
    }
}
```

到对应的配置文件：

/var/mobile/Library/Preferences/MuJiaBaiHuo.plist

注：

- MUJIABAIHUO_CONFIG_FILE 是加的 User-Defined 的参数
 -
 - 且：同时加 MUJIABAIHUO_CONFIG_FILE=\$(MUJIABAIHUO_CONFIG_FILE) 到 info.plist，才生效



tweak端

MuJiaBaiHuotweak的MuJiaBaiHuotweak.xm

```
const NSString * CONFIG_FILE = @"/var/mobile/Library/Preferences/MuJiaBaiHuo.plist";

- (NSString *)model
{
    os_log(OS_LOG_DEFAULT, "MuJiaBaiHuotweak hook UIDevice model");
    NSString * hookedModel = @"Tweaked_model";
    NSDictionary * curCfgDict = [[NSDictionary alloc] initWithContentsOfFile:(NSString *)CONFIG_FILE];
    os_log(OS_LOG_DEFAULT, "MuJiaBaiHuotweak curCfgDict=%{public}@", curCfgDict);
    if (curCfgDict) {
        NSString * phoneIdStr = [curCfgDict objectForKey:@"phoneId"];
        os_log(OS_LOG_DEFAULT, "MuJiaBaiHuotweak phoneIdStr=%{public}@", phoneIdStr);
        hookedModel = phoneIdStr;
        os_log(OS_LOG_DEFAULT, "MuJiaBaiHuotweak hookedModel=%{public}@", hookedModel);
    }

    os_log(OS_LOG_DEFAULT, "MuJiaBaiHuotweak return hookedModel=%{public}@", hookedModel);
}
return hookedModel;
}
```

即可：

从配置文件中

/var/mobile/Library/Preferences/MuJiaBaiHuo.plist

读取出之前保存的 NSDictionary，获取到参数 phoneId 的值。

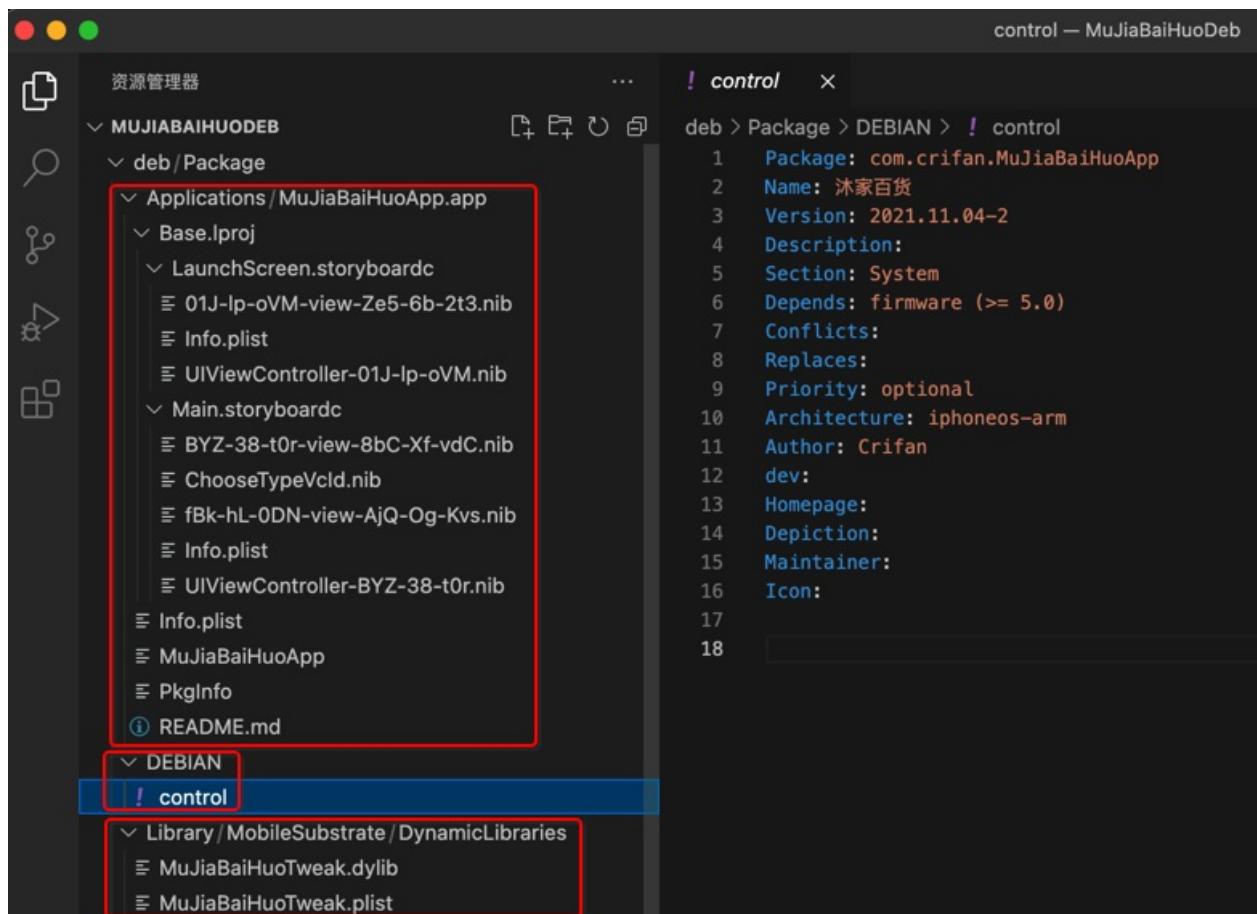
把tweak和app合并成单个deb文件

此处把：

- **tweak**: 的Package包中的Library文件夹
 - 去掉DEBIAN目录
 - 中的control
 - 无需tweak的control
- **app**: 的Package的目录中的所有内容
 - Applications

- DEBIAN
 - control
 - 需要app的control文件

合并到一起后的效果：



对应的目录结构：

```
→ MuJiaBaiHuodeb tree deb/Package
deb/Package
└── Applications
    └── MuJiaBaiHuApp.app
        ├── Base.lproj
        │   ├── LaunchScreen.storyboardc
        │   │   ├── 01J-lp-oVM-view-Ze5-6b-2t3.nib
        │   │   ├── Info.plist
        │   │   └── UIVViewController-01J-lp-oVM.nib
        │   └── Main.storyboardc
        │       ├── BYZ-38-t0r-view-8bC-Xf-vdC.nib
        │       ├── ChooseTypeVcId.nib
        │       ├── Info.plist
        │       ├── UIVViewController-BYZ-38-t0r.nib
        │       └── fBk-hL-0DN-view-AjQ-Og-Kvs.nib
        └── Info.plist
        └── MuJiaBaiHuApp
            └── PkgInfo
            └── README.md
└── DEBIAN
```

```
| └── control  
└── Library  
    └── MobileSubstrate  
        └── DynamicLibraries  
            ├── MuJiaBaiHuotweak.dylib  
            └── MuJiaBaiHuotweak.plist
```

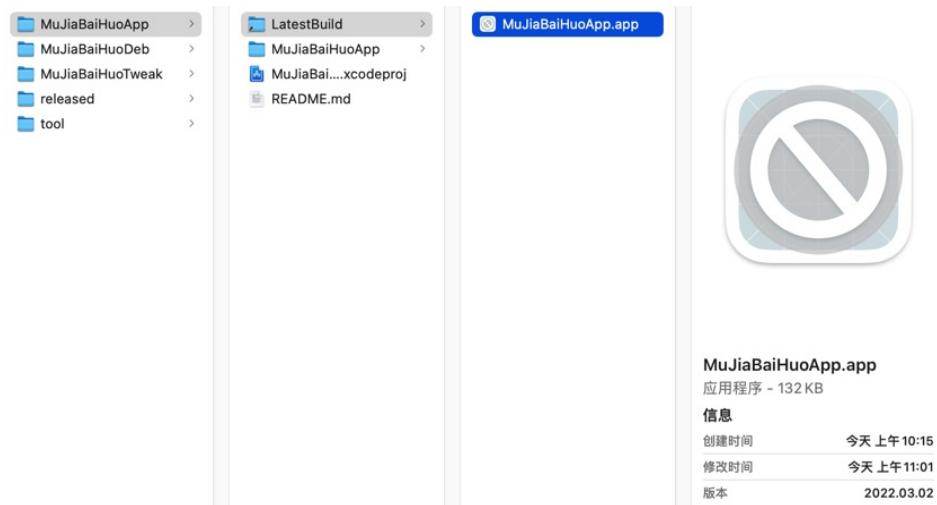
9 directories, 15 files

Finder中的效果：

- Applications
 -
- DEBIAN
 -
- Library
 -
-

关于目录中的各个文件的详细解释：

- Package : 要打包的根目录
 - Applications
 - MuJiaBaiHuotweak.dylib
 - 拷贝自：iOS的app的XCode项目下的LatestBuild
 - LatestBuild中保存了每次最新编译之后的版本

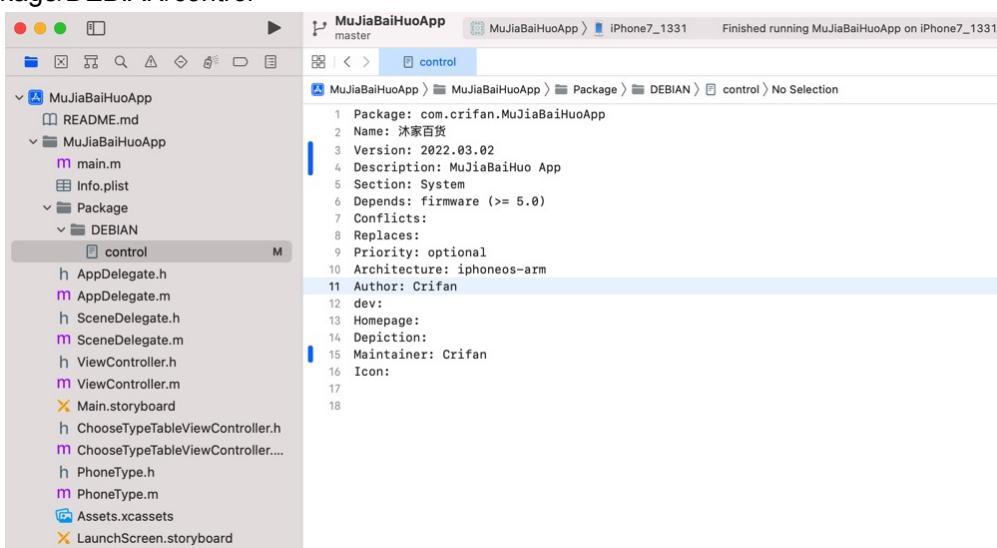


- DEBIAN

- control

- 用的是：iOS的app的Package/DEBIAN/control，不是iOS的tweak的

Package/DEBIAN/control



- Library

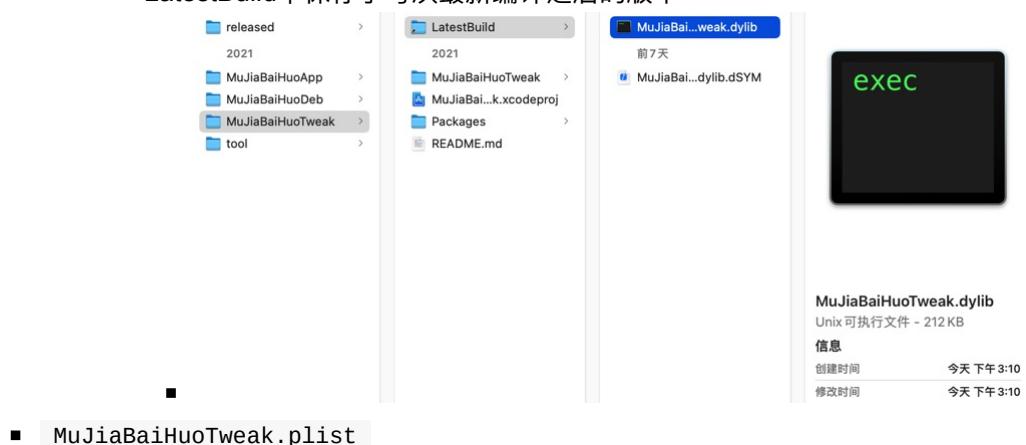
- MobileSubstrate

- DynamicLibraries

- MuJiaBaiHuoTweak.dylib

- 拷贝自：iOS的tweak的XCode项目下的LatestBuild

- LatestBuild中保存了每次最新编译之后的版本



- MuJiaBaiHuoTweak.plist

- 拷贝自iOS的tweak中的 Library/MobileSubstrate/DynamicLibraries

然后写个shell脚本：

- buildPackage.sh

```
#!/bin/bash
find ./Package -name ".DS_Store" -depth -exec rm {} \;
dpkg-deb -Zgzip -b Package MuJiaBaiHuo.deb
```

- 注：其实核心就一句： dpkg-deb -Zgzip -b Package MuJiaBaiHuo.deb

去加上可执行权限：

```
chmod +x ./buildPackage.sh
```

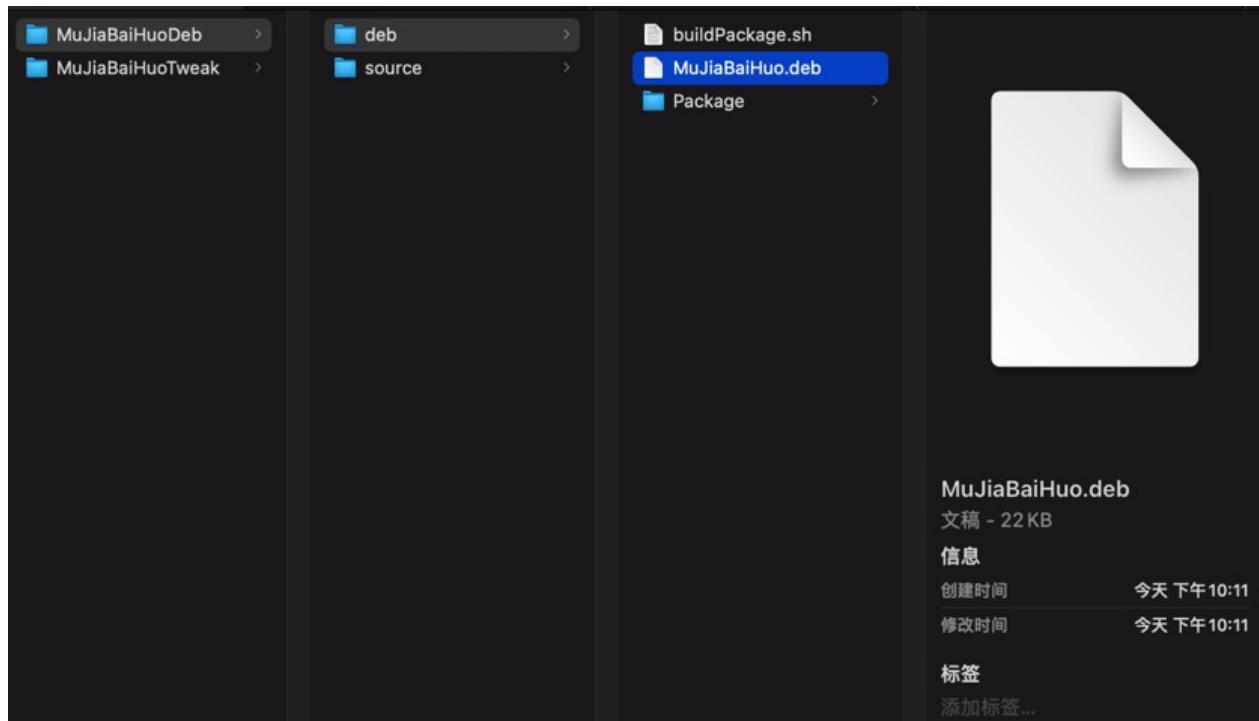
运行脚本去打包出deb

```
./buildPackage.sh
```

- log输出举例

```
→ 20220302 pwd
/Users/crifan/dev/DevRoot/zry/MuJiaBaiHuo/MuJiaBaiHuoDeb/20220302
→ 20220302 ./buildPackage.sh
dpkg-deb: 正在 'MuJiaBaiHuo.deb' 中构建软件包 'com.crifan.mujiabaihuoapp'。
```

即可得到deb插件安装包： MuJiaBaiHuo.deb



注：

- 如何安装

- 通过 Filza 或命令行 `dpkg` (命令是 `dpkg -i filename.deb`) 去安装deb文件，即可安装到 iPhone中。

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：
2023-01-06 23:01:29

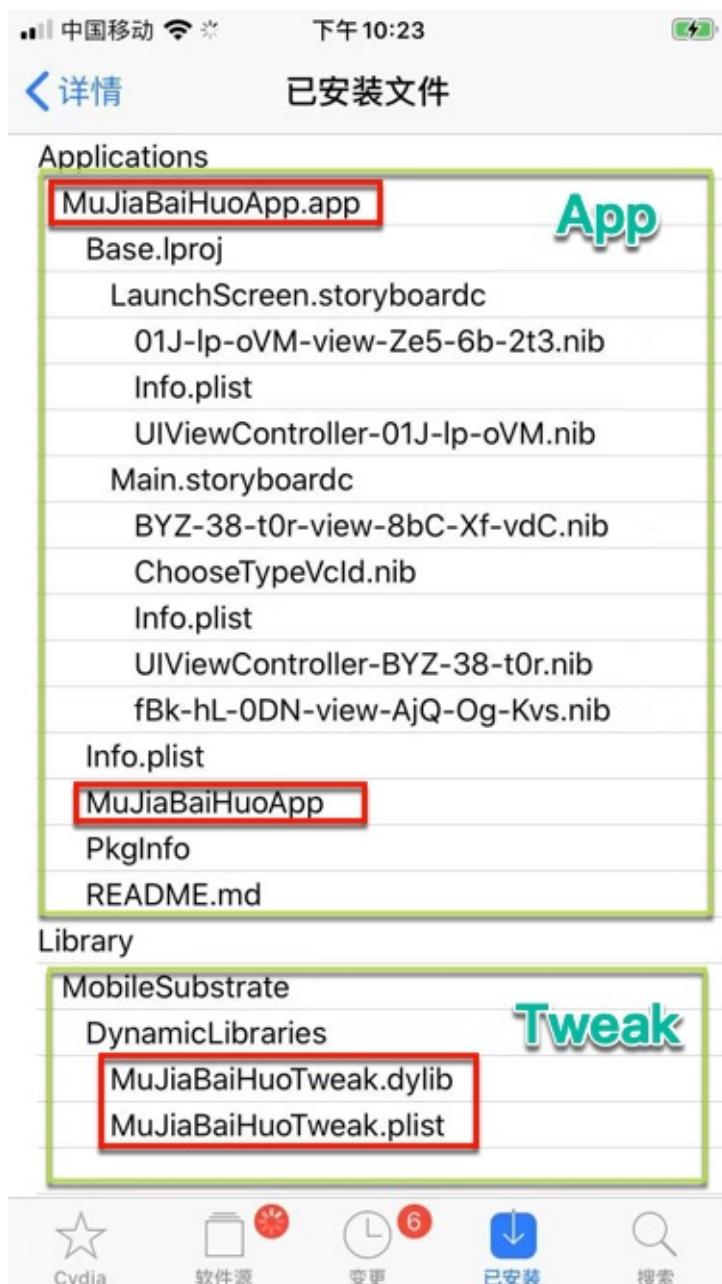
使用效果

插件安装后的效果

Cydia中可以看到安装后的插件：



其中详情中，点击 文件系统内容，可以看到包含的详细内容：

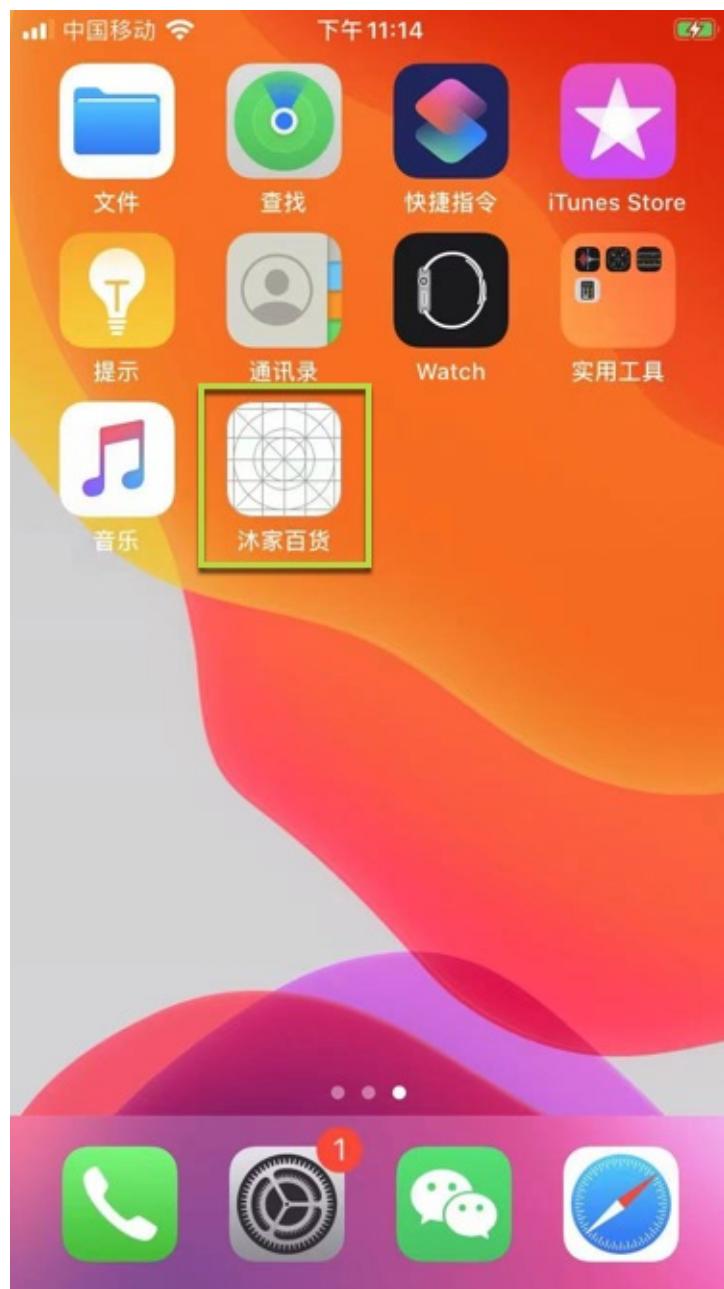


其中就有 Applications 和 Library

插件的使用效果

iOS的app部分：

桌面上的Logo=iOS的app的图标：



点击启动后，显示主界面：

中国移动

下午 10:57



选择机型

当前所选机型

点击 选择机型，出现：机型列表



iPhone 6

iPhone7,2

iPhone 6 Plus

iPhone7,1

iPhone 6s

iPhone8,1

iPhone 6s Plus

iPhone8,2

iPhone SE 一代

iPhone8,4

iPhone 7

iPhone9,1

iPhone 7 美版

iPhone9,3

iPhone 7 Plus

iPhone9,2

iPhone 7 Plus 美版

iPhone9,4

iPhone 8

iPhone10,1

iPhone 8 美版

iPhone10,4

去选择一个:

中国移动

下午 6:39



选择机型

iPhone 6 (iPhone7,2)

然后回去打开，被hook的普通的iOS的app： ShowSysInfo

可以检测当前机型，是我们所hook选择的机型：

中国移动 下午 6:42



v20211026_2139

获取 name	name
获取 systemName	systemName
获取 systemVersion	systemVersion
获取 model	iPhone7,2
获取 sysctl	hw.machine generation variant A number
获取 IDFV	identifierForVendor
获取运营商信息	allowsVOIP carrierName isoCountryCode mobileCountryCode mobileNetworkCode
获取状态栏 运营商	serviceString

-》说明上述的：tweak插件，iOS的app（用于配合插件做配置），是生效的。

注：

对应保存到了配置文件中的内容是：

phoneId=iPhone7,2



-》至此，跑通了：

- 带UI界面的tweak插件=单个deb文件，集成了包含了
 - app
 - 实现用户UI界面
 - 选择配置，保存配置到配置文件 /var/mobile/Library/Preferences/MuJiaBaiHuo.plist
 - tweak
 - 实现hook功能
 - 返回的值，根据配置文件中保存的值决定

常见问题

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：
2023-10-25 19:30:01

初始化环境方面的问题

安装器遇到了一个错误，导致安装失败

安装到最后，报错： 安装失败 安装器遇到了一个错误，导致安装失败



解决办法：

其实此时 `iosOpenDev` 的主体文件已安装到了默认的位置 `/opt` 中，接着去用工具初始化即可解决问题：

```
cd /opt/iOSOpenDevSetup/bin  
sudo ./iod-setup base  
sudo ./iod-setup sdk -sdk iphoneos
```

PrivateFramework directory not found XCode iPhoneOS15.0.sdk

iod-setup sdk -sdk iphoneos 时报错：

```
→ bin sudo ./iod-setup sdk -sdk iphoneos  
Setting up iPhoneOS 15.0 SDK...  
Modifying SDK settings...  
Symlinking to private frameworks header files...  
PrivateFramework directory not found: /Applications/Xcode.app/Contents/Developer/Platfo
```

```
rms/iPhoneOS.platform/Developer/SDKs/iPhoneOS15.0.sdk/System/Library/PrivateFrameworks
```

原因：

此处是比较新的 XCode 13 和对应的 iOS 15

->而最新版XCode和iOS早已将私有库PrivateFrameworks移走了

->即 iPhoneOSxx.xx.sdk/System/Library/ 下面没有 PrivateFrameworks 了

解决办法：

- 自己之后是否用到私有库PrivateFrameworks

◦ 否

- 直接新建一个空目录即可

```
cd /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS15.0.sdk/System/Library  
sudo mkdir PrivateFrameworks
```

◦ 是

- 除了新建目录外，还要把相关iPhoneOS版本的私有库的内容放过去

- 先要找到相关iPhoneOS的PrivateFrameworks

- 举例

- iPhoneOS 9.2 的 sdk，可以从这里下载到：

- zhangkn/knPrivateFrameworks:

[/Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS9.2.sdk/System/Library/PrivateFrameworks \(github.com\)](https://github.com/zhangkn/knPrivateFrameworks)

File not found XCode Specifications iPhoneOSPackageTypes.xcspec

iod-setup sdk -sdk iphoneos 报错：

```
→ bin sudo ./iod-setup sdk -sdk iphoneos  
Password:  
Setting up iPhoneOS 15.0 SDK...  
Modifying SDK settings...  
Symlinking to private frameworks header files...  
Adding specifications to platform...  
File not found: /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Developer/Library/Xcode/Specifications/iPhoneOSPackageTypes.xcspec
```

原因：

找不到specifications

解决办法：

下载别人给的：

- 4个iPhoneOS的spec文件
- 4个iPhoneSimulator的spec文件

分别放到对应位置，即可。

下载来源：

- 来源1：
 - [iosopendev专用Specifications.zip](#)
- 来源2：
 - [越狱开发:用iosOpenDev配置越狱开发环境 编写第一个hello world_我的杯洗具的博客-CSDN博客](#)

下载后，可以看到Specifications中有8个spec。

分别新建Specifications目录：

```
sudo mkdir /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Developer/Library/Xcode/Specifications
sudo mkdir /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneSimulator.platform/Developer/Library/Xcode/Specifications
```

再去

- 移动文件
 - 把
 - 4个 iPhoneos 的文件
 - iPhoneOSPackageTypes.xcspec
 - iPhoneOSPackageTypes.xcspec.iOSOpenDev
 - iPhoneOSProductTypes.xcspec
 - iPhoneOSProductTypes.xcspec.iOSOpenDev
 - 放到：
 - /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Developer/Library/Xcode/Specifications
 - 把：
 - 4个 iPhoneSimulator 的文件
 - iPhone Simulator PackageTypes.xcspec
 - iPhone Simulator PackageTypes.xcspec.iOSOpenDev
 - iPhone Simulator ProductTypes.xcspec
 - iPhone Simulator ProductTypes.xcspec.iOSOpenDev
 - 放到：
 - /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneSimulator.platform/Developer/Library/Xcode/Specifications

放好后是：

```
→ Xcode 11 /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Developer/Library/Xcode/Specifications
total 48
-rwxr-xr-x@ 1 crifan wheel 3.2K 12 24 2015 iPhoneOSPackageTypes.xcspec
```

```
-rwxr-xr-x@ 1 crifan wheel 5.4K 12 24 2015 iPhoneOSPackageTypes.xcspec.iOSOpenDev
-rwxr-xr-x@ 1 crifan wheel 4.0K 12 24 2015 iPhoneOSProductTypes.xcspec
-rwxr-xr-x@ 1 crifan wheel 6.4K 12 24 2015 iPhoneOSProductTypes.xcspec.iOSOpenDev
→ Xcode 11 /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneSimulator.platform/Developer/Library/Xcode/Specifications
total 48
-rwxr-xr-x@ 1 crifan wheel 3.4K 12 24 2015 iPhone Simulator PackageTypes.xcspec
-rwxr-xr-x@ 1 crifan wheel 6.9K 12 24 2015 iPhone Simulator PackageTypes.xcspec.iOSOpenDev
-rwxr-xr-x@ 1 crifan wheel 3.4K 12 24 2015 iPhone Simulator ProductTypes.xcspec
-rwxr-xr-x@ 1 crifan wheel 6.1K 12 24 2015 iPhone Simulator ProductTypes.xcspec.iOSOpenDev
```

另外，新建usr的bin目录：

```
sudo mkdir /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneSimulator.platform/Developer/usr/bin
```

即可。

Host key verification failed

- 现象：

Xcode编译期间报错：

```
Preparing to run Xcode Build Phase...
Signing /Users/crifan/Library/Developer/Xcode/DerivedData/iOSBypassJailbreak-bfqqgivvnccwmeaykhtbtvgylkkq/Build/Products/Release-iphoneos/iOSBypassJailbreak.dylib with ldid...
Done.
Copying /Users/crifan/Library/Developer/Xcode/DerivedData/iOSBypassJailbreak-bfqqgivvnccwmeaykhtbtvgylkkq/Build/Products/Release-iphoneos/iOSBypassJailbreak.dylib to package directory at /Users/crifan/dev/dev_root/crifan/iOSBypassJailbreak/iOSBypassJailbreak/Package/Library/MobileSubstrate/DynamicLibraries...
Preparing to build package...
Setting control file /Users/crifan/dev/dev_root/crifan/iOSBypassJailbreak/iOSBypassJailbreak/Package/DEBIAN/control Version field to 1.0-1 using /Users/crifan/dev/dev_root/crifan/iOSBypassJailbreak/iOSBypassJailbreak/PackageVersion.plist... Done.
Building package... Done.
Creating zip /Users/crifan/dev/dev_root/crifan/iOSBypassJailbreak/Packages/com.crifan.iOSBypassJailbreak_1.0-1_iphoneos-arm.zip... Done.
Host key verification failed.
Failed to create directory /var/root/iOSOpenDevPackages on device 192.168.1.27
Command PhaseScriptExecution failed with a nonzero exit code
```

- 原因：没有ssh免密登录
- 解决办法：设置好ssh免密登录

编译调试方面的问题

An empty identity is not valid when signing a binary for the product type 'Dynamic Library'

- 现象

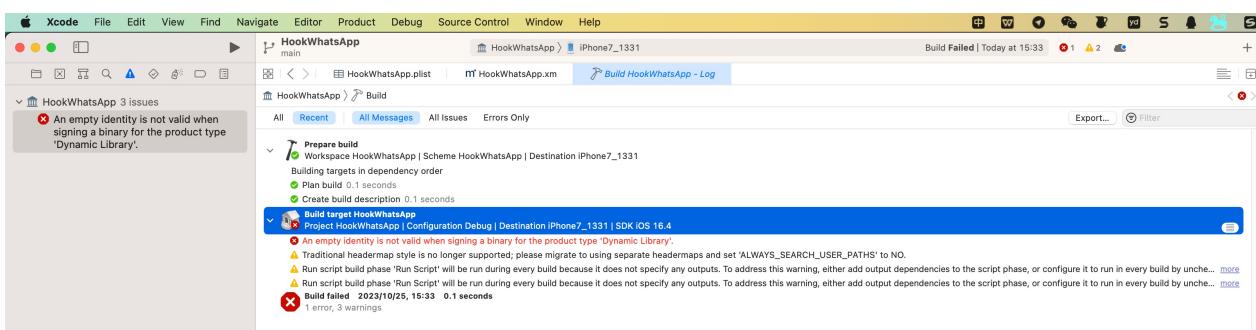
M2 Max Mac + Xcode 14.3.1 中，新建iOSOpenDev项目，去Build编译，报错：

```
An empty identity is not valid when signing a binary for the product type 'Dynamic Library'
```

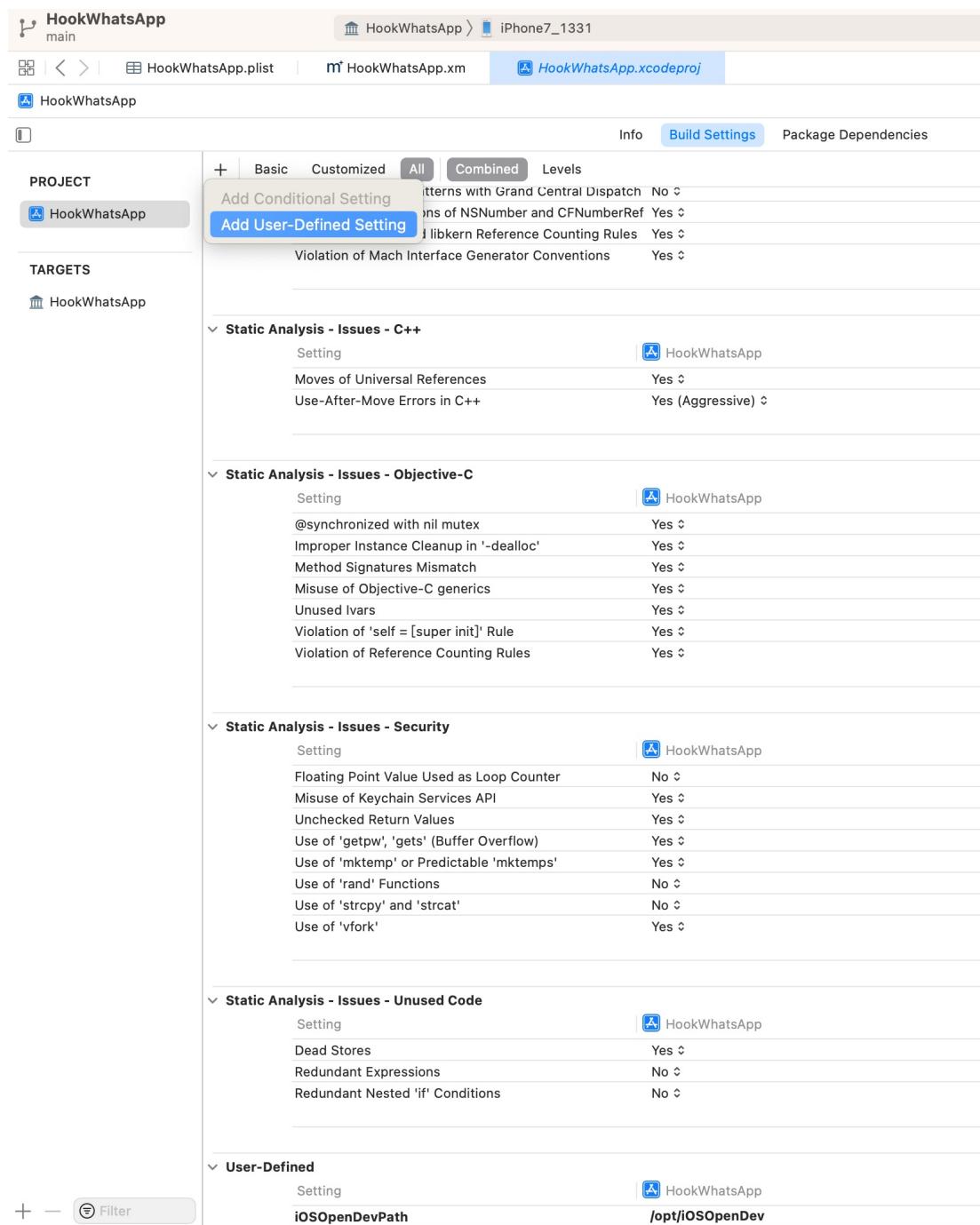
拷贝出的详细错误信息：

```
Build target HookWhatsApp of project HookWhatsApp with configuration Debug
error: An empty identity is not valid when signing a binary for the product type 'Dynamic Library' (in target 'HookWhatsApp' from project 'HookWhatsApp')
warning: Traditional headermap style is no longer supported; please migrate to using separate headermaps and set 'ALWAYS_SEARCH_USER_PATHS' to NO. (in target 'HookWhatsApp' from project 'HookWhatsApp')
warning: Run script build phase 'Run Script' will be run during every build because it does not specify any outputs. To address this warning, either add output dependencies to the script phase, or configure it to run in every build by unchecking "Based on dependency analysis" in the script phase. (in target 'HookWhatsApp' from project 'HookWhatsApp')
warning: Run script build phase 'Run Script' will be run during every build because it does not specify any outputs. To address this warning, either add output dependencies to the script phase, or configure it to run in every build by unchecking "Based on dependency analysis" in the script phase. (in target 'HookWhatsApp' from project 'HookWhatsApp')
```

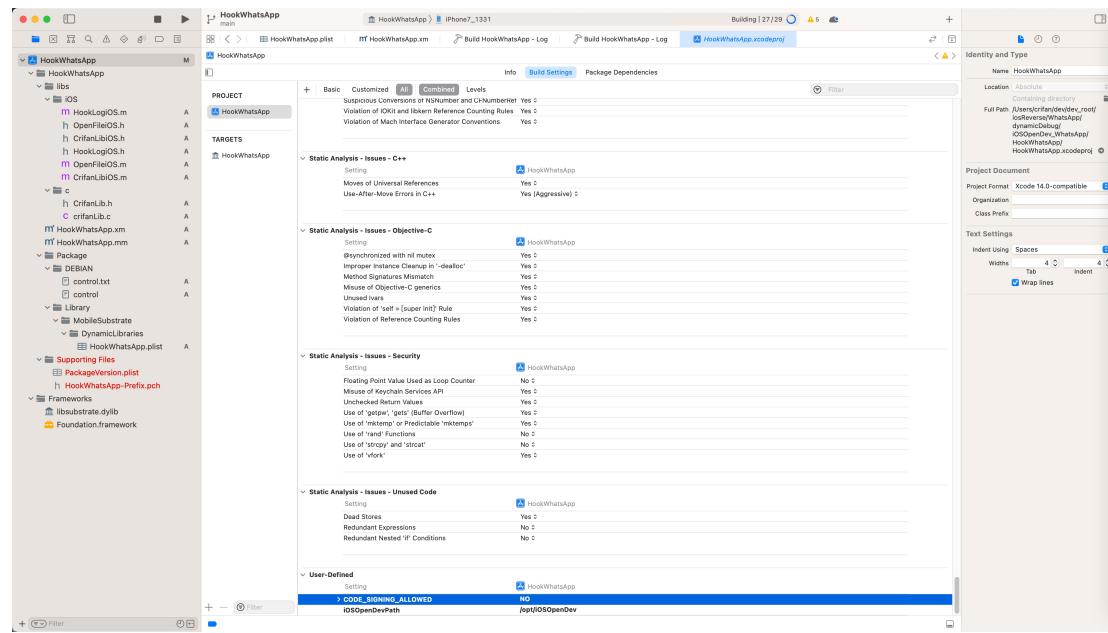
```
An empty identity is not valid when signing a binary for the product type 'Dynamic Library'
```



- 原因：不是很清楚
 - 只是大概知道，属于Xcode的自动管理codesign签名方面的问题
- 解决办法：禁用自动签名
- 具体步骤：Xcode -> Project -> {YourProjectName} -> Build Settings -> User-Defined ->
 - (点击左上角的 加号 = -> Add User-Defined Setting)



- 新增选项: CODE_SIGNING_ALLOWED = NO



- 额外说明

- 如果还不行，多试几次Clean：
 - Xcode -> Product -> Clean Build Folders
 - Xcode -> Product -> Clean All Issues

control的Version版本号的改动会丢失

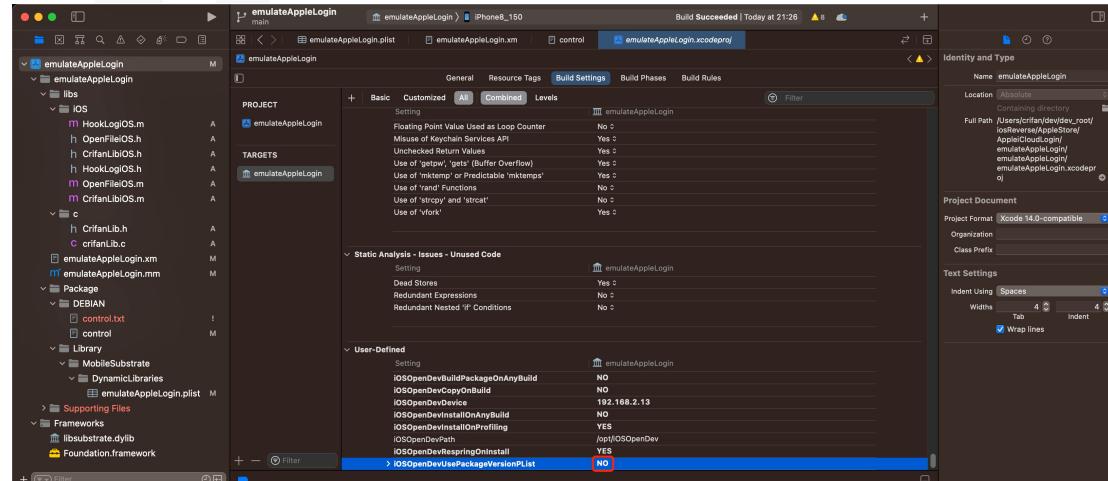
- 现象：

项目中的 .plist 中的 Version 的值，默认是 1.0-1

当想要去改动版本号，比如改为 2023.07.19.2126，结果重新编译后，改动后的Version值丢失，又恢复到之前的默认值 1.0-1 了

- 解决办法

- TARGETS -> Build Settings -> User-Defined -> iOSOpenDevUsePackageVersionPlist 从（默认的） YES 改为 NO



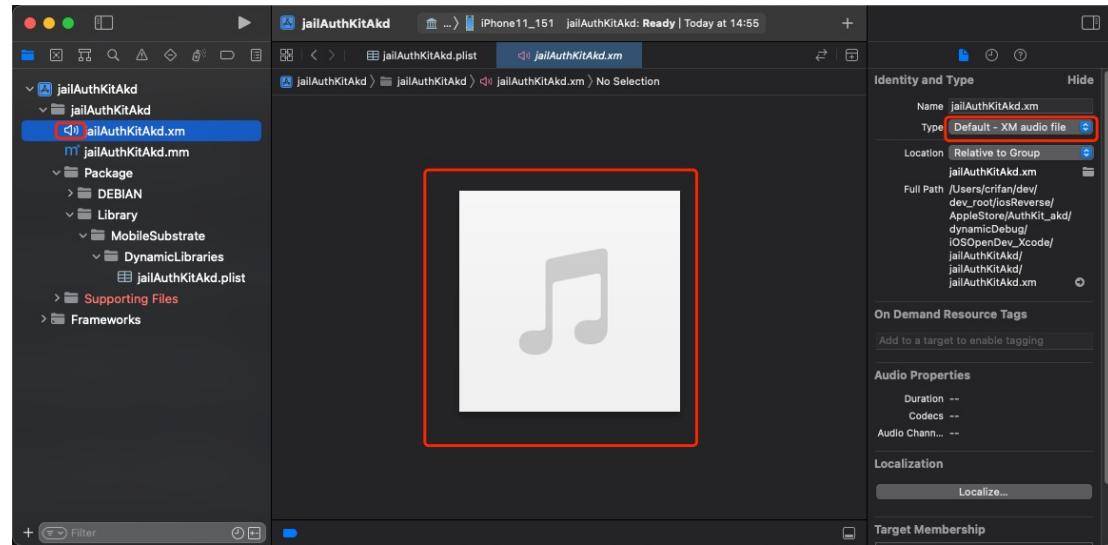
.xm 被识别为Audio音频文件

- 问题：iosOpenDev 的 Xcode 中，默认的 .xm 被识别成音频文件，无法显示对应的源代码

- 具体现象

- xm的图标是 小喇叭
- 且右边显示的是：音乐的图标
- 右边文件类型Type显示是： Default - XM audio file

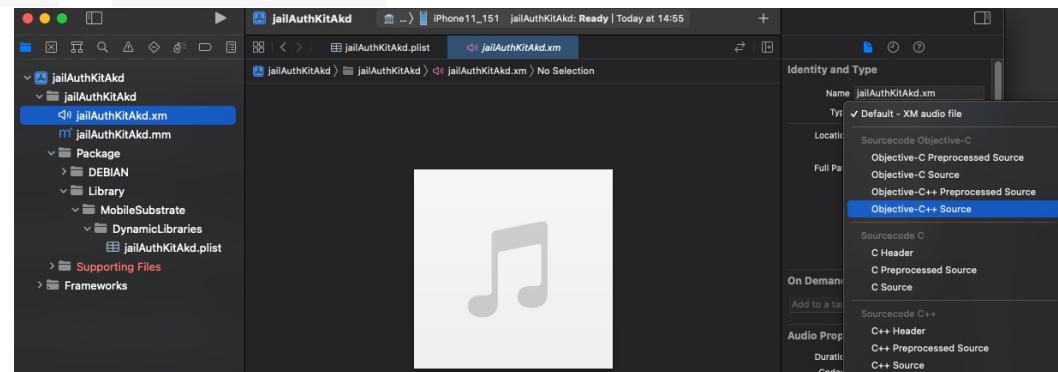
- 图



- 解决办法：

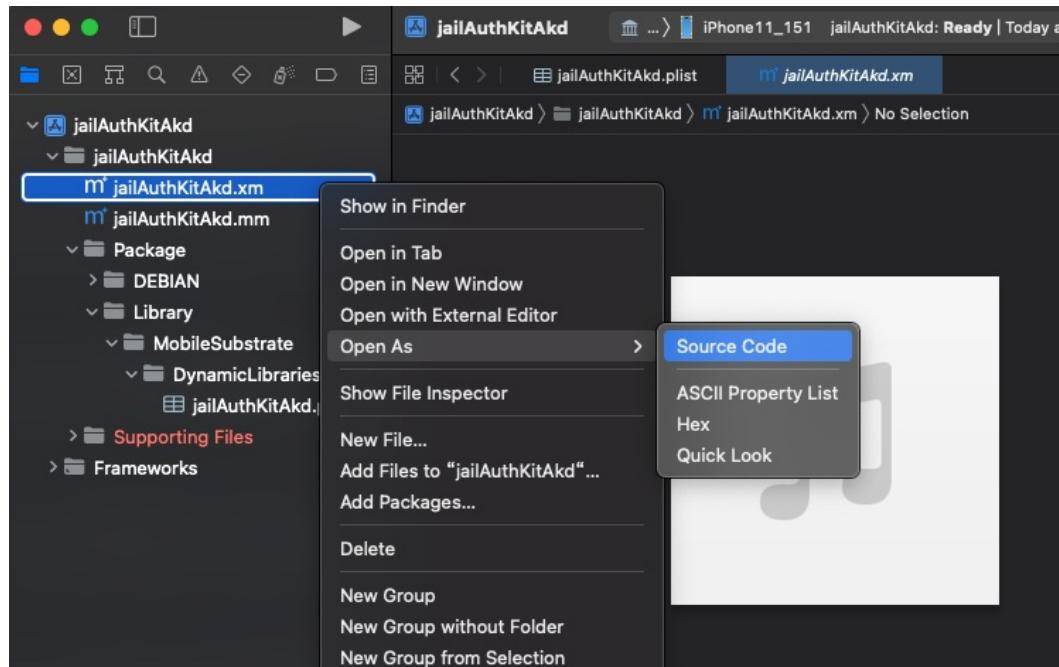
- 先去：改变.xm的文件类型

- Xcode右边的文件属性-> Type , 从 Default - XM audio file 改为 Objective-C++ Source (或 Objective-C Source)

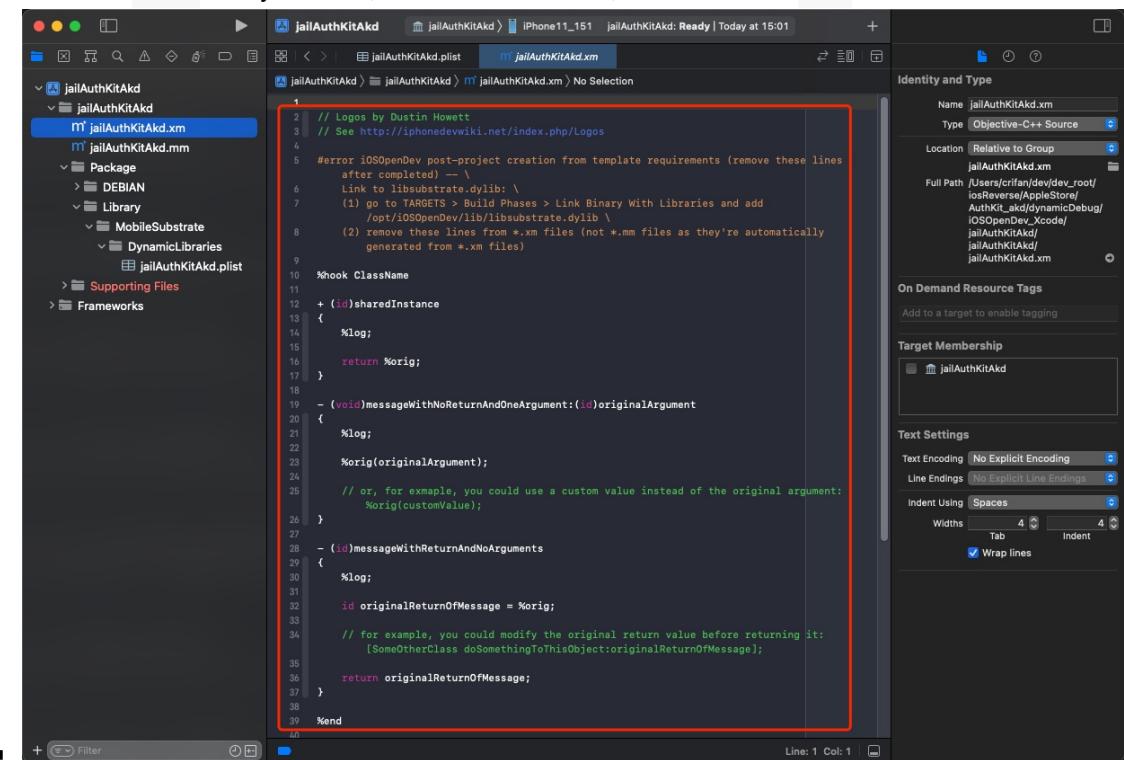


- 再去改变 .xm 的文件的打开方式

- Xcode左边文件列表->右键 .xm 文件-> Open As -> Source Code



- 即可正常显示 .xm 为ObjC的代码，并且带语法高亮了，且文件图标是 .m 的图标



安装插件后桌面上看不到iOS的app图标

- 现象：带UI界面的tweak或app，（通过Filza）安装到iPhone中后，桌面上看不到iOS的app的图标
- 原因：iPhone的UI界面没有刷新=icon图标没有刷新
- 解决办法：
 - 如果是通过Filza安装deb的话
 - Filza的安装完成界面的点击右上角： 动作 -> 选择： UIcache
 - 图



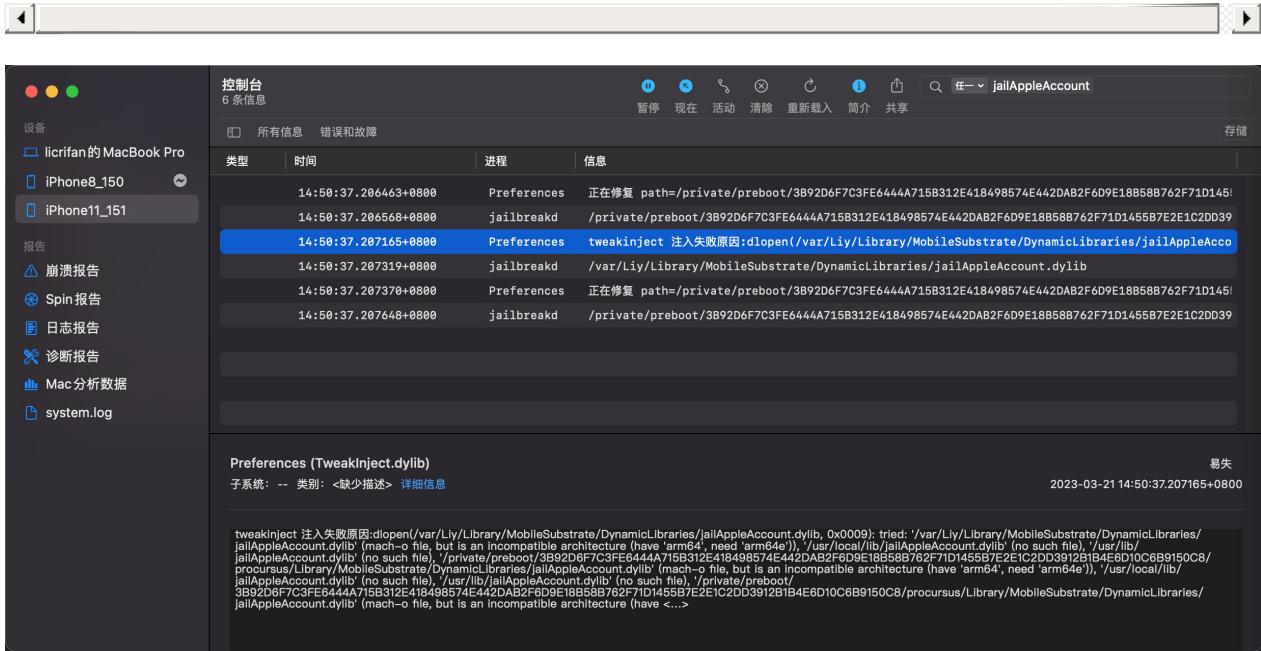
- 稍等片刻→ 桌面上即可出现iOS的app的logo图标了
 - 注：此时点击 `注销 = Respring = 重启SpringBoard`，虽然理论上可行，但实际是无效的，无法让桌面出现app图标的
 - 如果不是，则可以单独命令行去运行：`uicache`

mach-o file, but is an incompatible architecture have 'arm64', need 'arm64e'

- 现象

iOSOpenDev的Xcode编译出了插件dylib插件，但是启动加载时报错：

```
'/private/preboot/xxx/procursus/Library/MobileSubstrate/DynamicLibraries/jailAppleAccount.dylib' (mach-o file, but is an incompatible architecture (have 'arm64', need 'arm64e'))
```



- 原因：此处目标设备 iPhone11 的 CPU 是 A12，其架构是 arm64e 的，而插件代码编译出的架构是针对 arm64 的，不兼容，所以报错
- 解决办法：
 - Xcode中去把架构改为（包含=支持） arm64e
- 具体步骤：
 - Xcode -> TARGETS -> YourProjectName -> Build Settings -> Architectures -> Architectures
 - 从默认的: \$(ARCHS_STANDARD) == arm64, armv7
 - 改为: Other 的 arm64 arm64e

如此，即可确认所编译出来的代码（插件），支持arm64e了。

注：

- 如果额外引用到库文件，则也要确保库文件是支持此处的arm64e的
 - 比如此处遇到 libsubstrate.dylib，就是：
 - 默认（iOSOpenDev自带的）不支持arm64e，最后是另外找支持arm64e的
 - 比如
 - XinaA15越狱后的iPhone11中有
 - /private/preboot/3B92D6F7C3FE6444A715B312E418498574E442DAB2F6D9E18B58B762F71D1455B7E2E1C2DD3912B1B4E6D10C6B9150C8/procursus/usr/lib/libsubstrate.dylib
 - 大小: 218KB
 - (就是我们要的)FAT格式的，支持2种架构： arm64 和 arm64e
 - 去拷贝替换掉原先的： Mac 中的 /opt/iOSOpenDev/lib/libsubstrate.dylib
 - 才顺利编译和链接，才能确保插件正常工作

心得

TODO:

- 【未解决】 XCode调试警告: was compiled with optimization stepping may behave oddly variables may not be available
- 【未解决】 iOSOpenDev的XCode的tweak插件编译尝试去掉优化加上调试信息
-
- 【已解决】 iOSOpenDev的XCode调试iPhone6报错: Unable to install The application could not be verified
- 【已解决】 XCode中删除掉User-Defined的自定义参数
- 【已解决】 XCode中删除用户自定义配置User-Defined中的 CODE_SIGNING_ALLOWED=NO
-
- 【已解决】 iOSOpenDev的XCode的iOS的tweak插件中实现ObjC的通用全局函数
- 【已解决】 iOSOpenDev的XCode的iOS插件运行报错: ImageLoaderMachO doModInitFunctions和 _logosLocallInit
- 【已解决】 iOS代码报错: objc Class is implemented in both app and dylib One of the two will be used Which one is undefined
-
- 【已解决】 iOSOpenDev的XCode项目编译报错: iPhone Developer no identity found
- 【已解决】 调试iOSOpenDev的XCode的iOS的app
- 【已解决】 研究iOSOpenDev的XCode项目编译过程以确保如何链接自定义.c文件的.o文件
- 【已解决】 iOSOpenDev的XCode项目偶尔编译非常慢卡死
- 【已解决】 iOSOpenDev的XCode中新增.c和.h文件并正常编译
- 【已解决】 如何把XCode的iOS的app项目转换成iOSOpenDev的项目
- 【已解决】 对比研究FakeWeChatLoc和自己的XCode项目的目录结构区别
- 【已解决】 iOSOpenDev的XCode调试iPhone7报错: Unable to install A system application with the given bundle identifier is already installed on the device and cannot be replaced
- 【记录】 更新iOSOpenDev的Logos插件的code signing签名配置
- 【已解决】 XCode中iOSOpenDev开发插件代码报错: No matching function for call to strcpy
- 【已解决】 XCode中iOSOpenDev的Tweak项目中Build Settings中User-Defined中添加和引用变量 THEOS
- 【记录】 研究XCode中clang编译mm文件的过程和编译参数
- 【记录】 深究为何此处XCode编译strcpy会报错No matching function for call to
- 【未解决】 把之前theos的tweak改机剩余功能移植到iOSOpenDev的XCode中
-

.xm 文件和 .mm 文件

TODO:

- 【已解决】 Xcode中xm源码中无法看到和添加断点
- 【已解决】 iOSOpenDev的XCode中.xm文件包含.c中函数找不到报错: Undefined symbols for architecture arm64 referenced from

- 【已解决】 XCode的iOSOpenDev项目报错： Failed Logos Processor Could not open xm
- 【已解决】 iOSOpenDev的XCode中xm代码%hook编译报错： Expected unqualified-id
- 【已解决】 iOSOpenDev的XCode中如何把Tweak的xm代码拆分成多个文件模块
- 【未解决】 iOSOpenDev的iosod的bug修复： Logos的预处理不支持group子目录中的xm文件

.xm 文件

- .xm 文件， 默认会被 Xcode 识别为 音频文件
 - 需要去改变文件类型为 Objective C (或 Objective C++) 文件，再去显示为源代码

代码高亮

- 【已解决】 iOSOpenDev的XCode中新增xm文件设置为Logos语法高亮但无效
- 【已解决】 让XCode的iOSOpenDev中Logos的xm文件支持语法高亮

iOSOpenDev内部逻辑和过程

TODO:

- 【未解决】 研究iOSOpenDev的XCode项目编译过程以确保如何链接自定义.c文件的.o文件
- 【已解决】 XCode编译iOSOpenDev的Logo Tweak项目报错： Command PhaseScriptExecution failed with a nonzero exit code Failed to locate Logos Processor
- 【未解决】 XCode中编译iOSOpenDev的Logos的Tweak时shell从sh换为zsh
- 【已解决】 给iOS的XCode项目中新增iOSOpenDev的Project Navigator的目录和文件
- 【已解决】 XCode项目中新增iOSOpenDev的Package目录到Target目录中
- 【已解决】 XCode中如何把libsubstrate.dylib动态库导入到Link Binary With Libraries

如何卸载带UI的插件app

- 通过Cydia去卸载已安装的（带UI界面的）插件tweak即可

步骤：

Cydia -> 已安装 -> 最近 ->找到插件->进入详情页



点击右上角的 卸载 :



即可卸载掉插件。

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新:
2023-07-20 14:41:43

附录

下面列出相关参考资料。

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：
2022-11-08 10:39:15

参考资料

- 【整理】iOS越狱插件开发工具：iOSOpenDev
- 【已解决】把iOS的app和iOS的tweak插件打包成独立的deb安装包
- 【已解决】把iOSOpenDev的tweak加app的deb文件安装到已越狱的iPhone中
- 【已解决】如何把普通iOS的app的XCode项目和iOSOpenDev的Logos插件tweak集成到一起
- 【已解决】如何把XCode的iOS的app项目转换成iOSOpenDev的项目
- 【已解决】给iOSOpenDev的app和tweak用配置文件互相通信
- 【已解决】已越狱iPhone中卸载tweak加app的deb插件
- 【记录】确认iPhone中安装后的tweak加app是否正常使用
- 【已解决】把iOSOpenDev的tweak插件和app合并打包成deb文件
- 【已解决】把iOSOpenDev的tweak加app的deb文件安装到已越狱的iPhone中
- 【已解决】iOSOpenDev的XCode去Build For Profiling安装后iPhone桌面上找不到iOS的app的图标
- 【已解决】用Filza安装tweak加app的deb后iPhone桌面中仍没出现iOS的app的logo图标
- 【已解决】ssh登录iPhone失败：Host key verification failed
- 【已解决】Mac中安装iOSOpenDev
- 【已解决】Mac中安装iOSOpenDev报错：安装器遇到了一个错误，导致安装失败
- 【已解决】Mac中初始化iOSOpenDev环境并新建插件项目
- 【已解决】iOSOpenDev设置SDK报错：File not found XCode Specifications
iPhoneOSPackageTypes.xcspec
- 【已解决】iOSOpenDev设置SDK报错：PrivateFramework directory not found XCode
iPhoneOS15.0.sdk
- 【已解决】用iOSOpenDev去开发带GUI图形界面的iOS的app和tweak插件集成在一起的插件deb包
- 【已解决】iOS逆向：如何去hook一个进程而不是带包名的iOS的app
- 【已解决】XCode中iOSOpenDev中修改control的Version版本号无效会被重置
- 【未解决】iOS逆向akd：新建iOSOpenDev的Xcode插件项目
- 【已解决】iOSOpenDev的插件dylib注入iPhone11失败：mach-o file but is an incompatible
architecture have arm64 need arm64e
- 【已解决】寻找支持arm64e的libsubstrate.dylib
- 【已解决】Xcode项目中引用新的libsubstrate.dylib无效：始终链接是旧的库文件
- 【已解决】Mac M2 Max中iOSOpenDev编译报错：An empty identity is not valid when signing a
binary for the product type Dynamic Library
-
- iosOpenDev-install 失败官方wiki无法解决看这里（尝试有效） - PoloKey - 博客园 (cnblogs.com)
- zhangkn/knPrivateFrameworks:
[/Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS9.2.sdk/System/Library/PrivateFrameworks](http://Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS9.2.sdk/System/Library/PrivateFrameworks) (github.com)
- iosopendev专用Specifications.zip
- 越狱开发:用iosOpenDev配置越狱开发环境 编写第一个hello world_我的杯洗具的博客-CSDN博客
- jackrex/FakeWeChatLoc: 手把手教你制作一款iOS越狱App (github.com)
- ios - An empty identity is not valid when signing a binary for the product type 'Application' in xcode
version 10.2 - Stack Overflow
-

2023-10-25 19:49:32