

# 目录

前言	1.1
iOS逆向动态调试概览	1.2
反调试和反反调试	1.3
举例	1.3.1
抖音	1.3.1.1
调试代码逻辑	1.4
Xcode	1.4.1
MonkeyDev	1.4.2
lldb+debugserver	1.4.3
Frida	1.4.4
调试界面元素	1.5
Reveal	1.5.1
Cycrypt	1.5.2
初始化环境	1.5.2.1
基本用法	1.5.2.2
使用心得	1.5.2.3
输出举例	1.5.2.4
LLDBTools	1.5.3
chisel	1.5.4
FLEX	1.5.5
Passionfruit	1.5.6
动态调试心得	1.6
调试手段对比	1.6.1
Xcode相关	1.6.2
ObjC	1.6.3
objc_msgSend	1.6.3.1
Runtime	1.6.3.2
po	1.6.4
子教程	1.7
附录	1.8
参考资料	1.8.1

# iOS逆向开发：动态调试

- 最新版本: v1.2.4
- 更新时间: 20231021

## 简介

介绍iOS逆向中的动态调试，包括动态调试的概览；以及调试代码逻辑方面，包括调试工具的MonkeyDev、lldb+debugserver、Frida等；以及相关子领域，比如反调试和反反调试等；以及查看界面元素的工具，比如Reveal、Cycrypt、LLDBTools、chisel、FLEX等；且详细介绍了Cycrypt的使用和心得；最后给出一些经验心得。

## 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\\_dynamic\\_debug: iOS逆向开发：动态调试](#)

### 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：动态调试 book.crifan.org](#)
- [iOS逆向开发：动态调试 crifan.github.io](#)

### 离线下载阅读

- [iOS逆向开发：动态调试 PDF](#)
- [iOS逆向开发：动态调试 ePUB](#)
- [iOS逆向开发：动态调试 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](mailto:admin@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：

2023-10-21 19:57:42

# iOS逆向动态调试概览

iOS逆向，从 是否要运行代码 的角度来说，分：

- 不要运行代码的：静态分析
- 要运行代码的：动态调试

此文主要介绍 动态调试 的相关内容：

- 输入=前提：砸壳出的ipa文件（或已把ipa安装到iOS设备中）
- 主要涉及的内容=领域
  - 调试代码逻辑
    - 常见调试工具
      - 图形界面：[Xcode + MonkeyDev](#)
      - 命令行：[debugserver + lldb](#)
        - 主流调试器：LLDB
      - [Frida](#)
      - [IDA](#)
    - 涉及到的相关子领域
      - [反调试和反反调试](#)
      - [Xcode开发：调试心得](#)
  - 调试界面元素
    - [Reveal](#)
    - [Cycrypt](#)
    - [LLDBTools](#)
    - [chisel](#)
    - [FLEX](#)

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：  
2023-07-13 22:00:27

# 反调试和反反调试

TODO:

- 【整理】iOS反越狱相关：反调试 反反调试
  - 【已解决】iOS反调试和反反调试：syscall的ptrace
  - 【未解决】iOS反调试和反反调试：svc 0x80的syscall的ptrace
  - 【已解决】debugserver启动iOS的app抖音报错：Segmentation fault 11
- 

由于现在多数iOS的app，都做了 反调试 的防护，导致想要能顺利调试iOS的app之前，都要去解决： 反反调试 。

所以此处就分别涉及到：

- 正向的： 反调试
- 逆向的： 反反调试

## 反调试

## 反反调试

- 概述=核心思路
  - 先去研究清楚，具体对方用了什么反调试手段
  - 再去针对性的处理

## 举例

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：  
2023-10-21 19:36:46

## 反调试和反反调试举例

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：  
2023-10-21 19:35:51

## 反调试和反反调试举例：抖音

此处以 抖音 为例，介绍，反调试和反反调试的具体过程：

### 先去研究清楚，具体对方用了什么反调试手段

Mac中用lldb去调试iPhone中的iOS的app抖音，结果报错：

```

...
espresso version: 2.7.11.0520
espresso version: 2.9.33.1117
mobilecv2: 1.9.0.1013
Process 10174 exited with status = 45 (0x0000002d)

lldb (lldb)
byteaudio (0x114678530) and /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore (0x10149e630)
. One of the two will be used. Which one is undefined.
objc[10174]: Class PodsDummy_FileMD5Hash is implemented in both /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/byteaudio.framework/byteaudio (0x114678580) and /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore (0x10149e680). One of the two will be used. Which one is undefined.
objc[10174]: Class MTLJSONAdapter is implemented in both /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/byteaudio.framework/byteaudio (0x1146787d8) and /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore (0x1017ef960). One of the two will be used. Which one is undefined.
objc[10174]: Class MTLModel is implemented in both /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/byteaudio.framework/byteaudio (0x114678828) and /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore (0x1017ef9b0). One of the two will be used. Which one is undefined.
objc[10174]: Class MTLValueTransformer is implemented in both /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/byteaudio.framework/byteaudio (0x1146788a0) and /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore (0x1017efa28). One of the two will be used. Which one is undefined.
objc[10174]: Class MTLReversibleValueTransformer is implemented in both /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/byteaudio.framework/byteaudio (0x114678878) and /private/var/containers/Bundle/Application/9AB25481-0AD3-435C-A02E-68F9623535BB/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore (0x1017efa00). One of the two will be used. Which one is undefined.
espresso version: 2.7.11.0520
espresso version: 2.9.33.1117
mobilecv2: 1.9.0.1013
Process 10174 exited with status = 45 (0x0000002d)
(lldb) 
```

研究后找到定义：

- [/System/Library/Frameworks/Kernel.framework/Versions/A/Headers/sys/errno.h](#)
- [errno.h \(apple.com\)](#)

```
#define ENOTSUP 45           /* Operation not supported */
```

-»

- 45 = ENOTSUP = Operation not supported
  - 意思是：lldb调试的操作，不被允许
  - 原因：app内部做了反调试
    - 那就意味着是 ptrace 的 PT\_DENY\_ATTACH 的事情了

而实现 ptrace 的 PT\_DENY\_ATTACH，之前已知有4种方式：

- ptrace
 

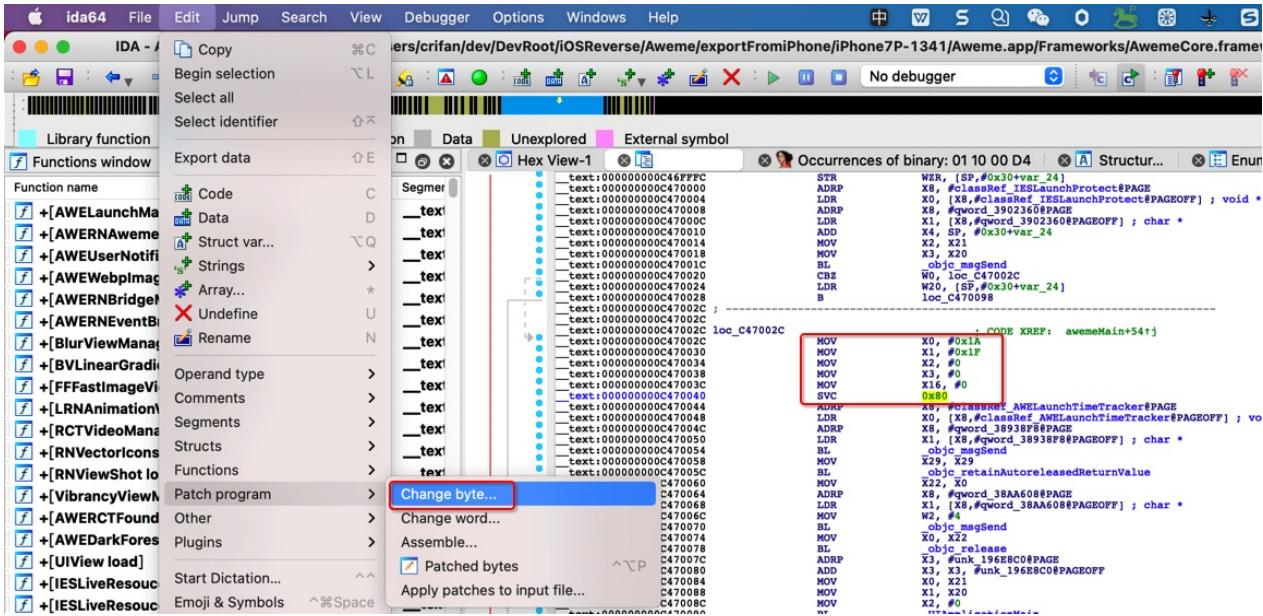
```
ptrace(PT_DENY_ATTACH, 0, 0, 0);
```
- syscall() = syscall + ptrace + PT\_DENY\_ATTACH
 

```
syscall(SYS_ptrace, PT_DENY_ATTACH, 0, NULL, 0);
```
- sysctl ?
- svc 0x80 + syscall + ptrace = inline asm = ARM汇编
 

```
MOV X0, #26 // ptrace
MOV X1, #31 // PT_DENY_ATTACH
MOV X2, #0
MOV X3, #0
MOV X16, #0
SVC #0x80
```

继续后续调试，找到代码：

```
__text:000000000C47002C loc_C47002C ; CODE XREF: _awemeMain+54↑j
__text:000000000C47002C MOV X0, #0x1A
__text:000000000C470030 MOV X1, #0x1F
__text:000000000C470034 MOV X2, #0
__text:000000000C470038 MOV X3, #0
__text:000000000C47003C MOV X16, #0
__text:000000000C470040 SVC 0x80
```



至此，清楚了：

- 抖音的反调试手段
    - svc 0x80 的内联汇编 实现的 syscall 的 ptrace 的 PT\_DENY\_ATTACH

**再去针对性的处理=反反调试**

此处，针对：

- 抖音的反调试手段
    - svc 0x80 的内联汇编 实现的 syscall 的 ptrace 的 PT\_DENY\_ATTACH

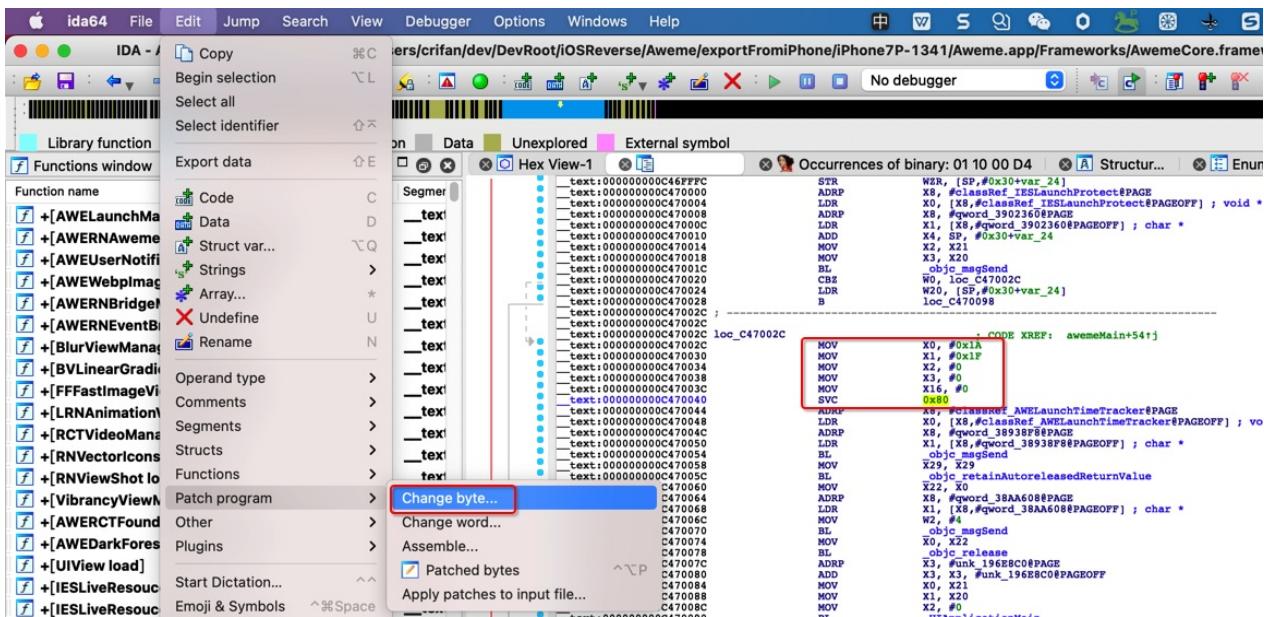
去实现：

- 反反调试的抖音的手段
    - 把内联的ARM汇编的 svc 0x80 指令替换成 NOP 指令=空指令

## 具体操作：

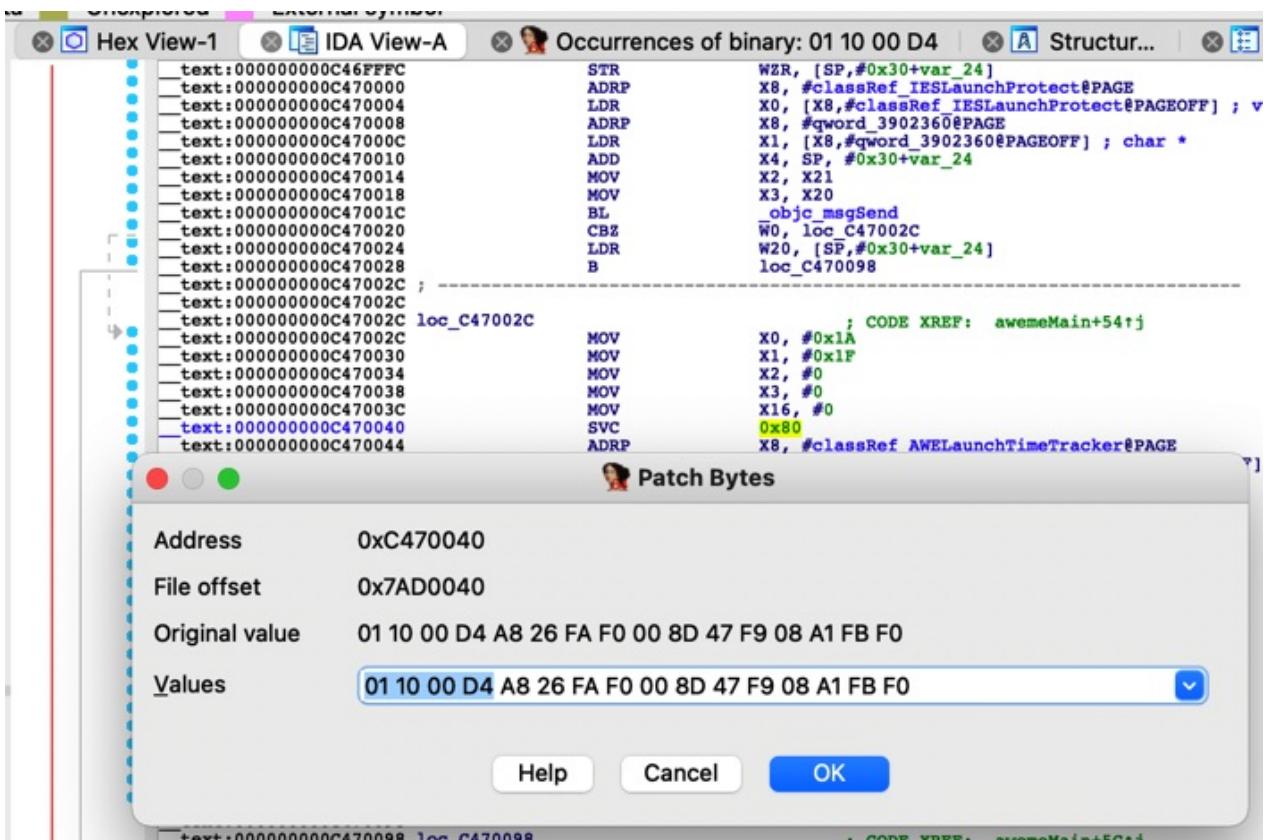
- 前提：已找到 `svc 0x80` 的ARM汇编指令的位置了
    - 如果有多处，也要全部都找到
  - 思路：借助于IDA实现指令替换，把 `svc 0x80` 相关指令，替换成NOP指令
  - 步骤：

IDA Pro ->鼠标停留=选中对应的要修改的汇编代码（所在的行）-> Edit -> Patch Program -> Change byte :



会出现：

- ARM汇编指令：
  - svc 0x80 对应的二进制=bytecode=字节码： 01 10 00 D4



此处借助于：

- Online ARM to HEX Converter ([armconverter.com](http://armconverter.com))

去搞清楚：

要替换成的：

- NOP指令

- (ARM的 ARM64 的, 此处的端是: Little Endian )

- 对应的字节码=bytecode: 1F 20 03 D5

The screenshot shows the Online ARM to HEX Converter. In the 'Assembly code' tab, the input is 'NOP'. The output in the 'ARM' tab is '1F2003D5'. Below the input field is an 'Offset (hex)' section with '0x 0 - for branch and LDR put hex value here'. At the bottom is a green 'CONVERT' button.

即, 把: 01 10 00 D4 改为 1F 20 03 D5

The screenshot shows the IDA Pro interface with the assembly view selected. A patch dialog is open over the assembly code. The assembly code at address 0xC470040 is:

```

text:00000000C470040    STR      WZR, [SP,#0x30+var_24]
text:00000000C470040    ADRP    X8, #classRef_IESLaunchProtect@PAGE
text:00000000C470040    LDR     X0, [X8,#classRef_IESLaunchProtect@PAGEOFF] ; void
text:00000000C470040    ADRP    X8, #qword_3902360@PAGE
text:00000000C470040    LDR     X1, [X8,#qword_3902360@PAGEOFF] ; char *
text:00000000C470040    ADD     X4, SP, #0x30+var_24
text:00000000C470040    MOV     X2, X21
text:00000000C470040    MOV     X3, X20
text:00000000C470040    BL     _objc_msgSend
text:00000000C470040    CBZ    W0, loc_C47002C
text:00000000C470040    LDR     W20, [SP,#0x30+var_24]
text:00000000C470040    B      loc_C470098
text:00000000C470040    ; -----
text:00000000C470040    text:00000000C47002C loc_C47002C
text:00000000C470040    MOV     X0, #0x1A
text:00000000C470040    MOV     X1, #0x1F
text:00000000C470040    MOV     X2, #0
text:00000000C470040    MOV     X3, #0
text:00000000C470040    SVC     X16, #0
text:00000000C470040    ADRP    X8, #classRef_AWELaunchTimeTracker@PAGE

```

The patch bytes dialog shows the current value as '01 10 00 D4' and the new value as '1F 20 03 D5'. The address is 0xC470040.

点击 OK , 即可修改成功:

```

text:00000000C46FFF C    STR    WZR, [SP,#0x30+var_24]
text:00000000C470000 ADRP   X8, #classRef_IESLaunchProtect@PAGE
text:00000000C470008 LDR    X0, [X8,#word_3902360@PAGEOFF] ; void *
text:00000000C470010 ADD    X8, SP, #0x30+var_24
text:00000000C470014 MOV    X2, X0
text:00000000C470018 MOV    X3, X0
text:00000000C47001C BL     objc_msgSend
text:00000000C470020 CBZ   W0, loc_C47002C
text:00000000C470024 LDR    W20, [SP,#0x30+var_24]
text:00000000C470028 B     loc_C470098
text:00000000C47002C ; -----
loc_C47002C:           MOV    X0, #0x1A ; CODE XREF: awemeMain+54+j
text:00000000C470030 MOV    X1, #0x1F
text:00000000C470034 MOV    X2, #0
text:00000000C470038 MOV    X3, #0
text:00000000C47003C MOV    X16, #0
text:00000000C470040 NOP
text:00000000C470044 ADRP   X8, #classRef_AWELaunchTimeTracker@PAGE

```

再去保存，保存到原输入文件：

```

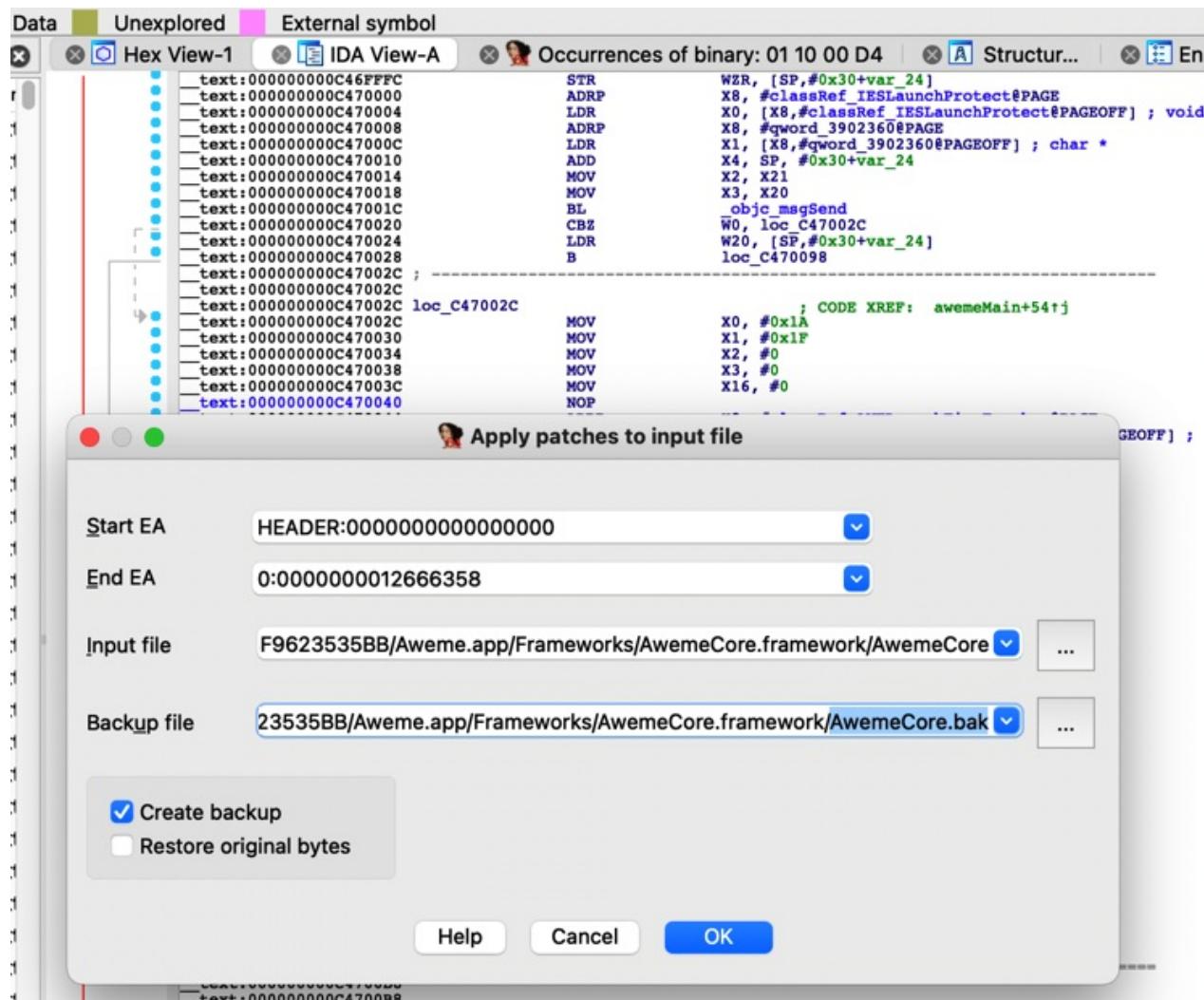
text:00000000C46FFF C    STR    WZR, [SP,#0x30+var_24]
text:00000000C470000 ADRP   X8, #classRef_IESLaunchProtect@PAGE
text:00000000C470008 LDR    X0, [X8,#word_3902360@PAGEOFF] ; void *
text:00000000C470010 ADD    X8, SP, #0x30+var_24
text:00000000C470014 MOV    X2, X0
text:00000000C470018 MOV    X3, X0
text:00000000C47001C BL     objc_msgSend
text:00000000C470020 CBZ   W0, loc_C47002C
text:00000000C470024 LDR    W20, [SP,#0x30+var_24]
text:00000000C470028 B     loc_C470098
text:00000000C47002C ; -----
loc_C47002C:           MOV    X0, #0x1A ; CODE XREF: awemeMain+54+j
text:00000000C470030 MOV    X1, #0x1F
text:00000000C470034 MOV    X2, #0
text:00000000C470038 MOV    X3, #0
text:00000000C47003C MOV    X16, #0
text:00000000C470040 NOP
text:00000000C470044 ADRP   X8, #classRef_AWELaunchTimeTracker@PAGE

```

Patch program

- > Change byte...
- > Change word...
- > Assemble...
- Patched bytes
- Apply patches to input file...

此处，为了更好保留原文件，点击勾选： Create backup



会自动生成 xxx.bak

- Aweme.app/Frameworks/AwemeCore.framework/AwemeCore
  -
- Aweme.app/Frameworks/AwemeCore.framework/AwemeCore.bak

确认文件的确已变化：

```
→ AwemeCore.framework pwd
/Users/crifan/dev/DevRoot/iOSReverse/Aweme/exportFromiPhone/iPhone7P-1341/Aweme.app/Frameworks/AwemeCore.framework
→ AwemeCore.framework ls -l
total 18632888
-rw-r--r-- 1 crifan staff 240666608 1 8 09:43 AwemeCore
-rw-r--r-- 1 crifan staff 240666608 1 5 15:00 AwemeCore.backup
-rw-r--r-- 1 crifan staff 240666608 1 8 09:43 AwemeCore.bak
...
```

->

- AwemeCore
  - 240666608
- AwemeCore.bak

- 240666608

-》文件大小没变，最后改动时间变了。

另外，再去：拷贝到iPhone7中，重新调试。

最终效果：

去掉反调试后，可以正常用[debugserver+lldb](#)去调试启动抖音了：



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:  
2023-10-21 19:57:08

## 调试代码逻辑

TODO:

- 【未解决】如何调试iPhone中iOS的app
- 

iOS逆向中的动态调试，其中主要是关于，用各种调试工具去调试代码逻辑。

常用调试工具有：

- MonkeyDev
- lldb+debugserver
- Frida
- IDA
  - 概述：其实IDA更多的是用来[静态分析](#)代码逻辑，偶尔用来[动态调试](#)

以及相关心得：[独立子教程](#)

- Xcode调试心得
  - [Xcode开发：调试心得](#)
- LLDB调试心得
  - [主流调试器：LLDB](#)

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：

2023-07-13 21:57:47

## Xcode

iOS逆向期间，对于想要调试 app / 二进制 的代码逻辑的手段不少。

此处之所以把，看起来是单独属于 [Xcode开发：调试心得](#) 中的 `xcode`，单独拿出来说，原因是：

- Xcode调试的优势
  - 是GUI图形界面的，比 `lldb+debugserver` 更加直观和方便
- Xcode调试需要满足前提
  - `app / 二进制` 是可调试的 = 任意进程可调试

其中最重要的是：

- 要满足任意进程可调试的前提条件
  - 如何实现任意进程可调试
    - 概述
      - 如果是XinaA15越狱后，自动已实现任意进程可调试，无需额外操作
      - 否则就要自己手动去操作： Mac 中用 `codesign` 给 `app / 二进制` 重签名，替换掉 iPhone中原有的 `app / 二进制`
    - 详解
      - [任意进程可调试 · iOS逆向开发：签名和权限](#)
        - [XinaA15自带已支持](#)
        - [手动重签名](#)

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：  
2023-07-21 16:22:19

# MonkeyDev

详见独立子教程：

[iOS逆向开发：MonkeyDev调试 \(crifan.org\)](#)

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：

2022-11-03 15:06:55

## lldb+debugserver

iOS逆向时，调试代码逻辑的常用调试工具之一是：命令行的 lldb + debugserver

- 概述
  - 用 codesign / ldid 给 debugserver 重签名，支持任意进程可调试后，放到 /usr/bin/debugserver，即可启动iPhone端的 debugserver 和Mac端的 lldb 去调试iOS的 app的进程了
- 详见
  - [iOS逆向调试：debugserver+lldb](#)

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：

2023-10-21 10:52:19

# Frida

- 概述
  - Frida
    - 概述
      - Android 、 iOS 的app逆向等领域中，最常用的工具之一
    - 主要用法：iOS逆向期间，用 frida 和 frida-trace 去动态调试代码逻辑
    - 一句话描述
      - A world-class Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers
      - Inject JavaScript to observe and reprogram running programs on Windows, macOS, GNU/Linux, iOS, watchOS, tvOS, Android, FreeBSD, and QNX
    - 主页
      - <https://frida.re/>
    - 作者：`oleavr` = Ole André Vadla Ravnås
      - Github
        - <https://github.com/oleavr>
      - 所属公司
        - NowSecure
          - <https://www.nowsecure.com/>
- 详解
  - 独立子教程
    - 逆向调试利器：Frida

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：  
2023-07-10 09:26:55

# 调试界面元素

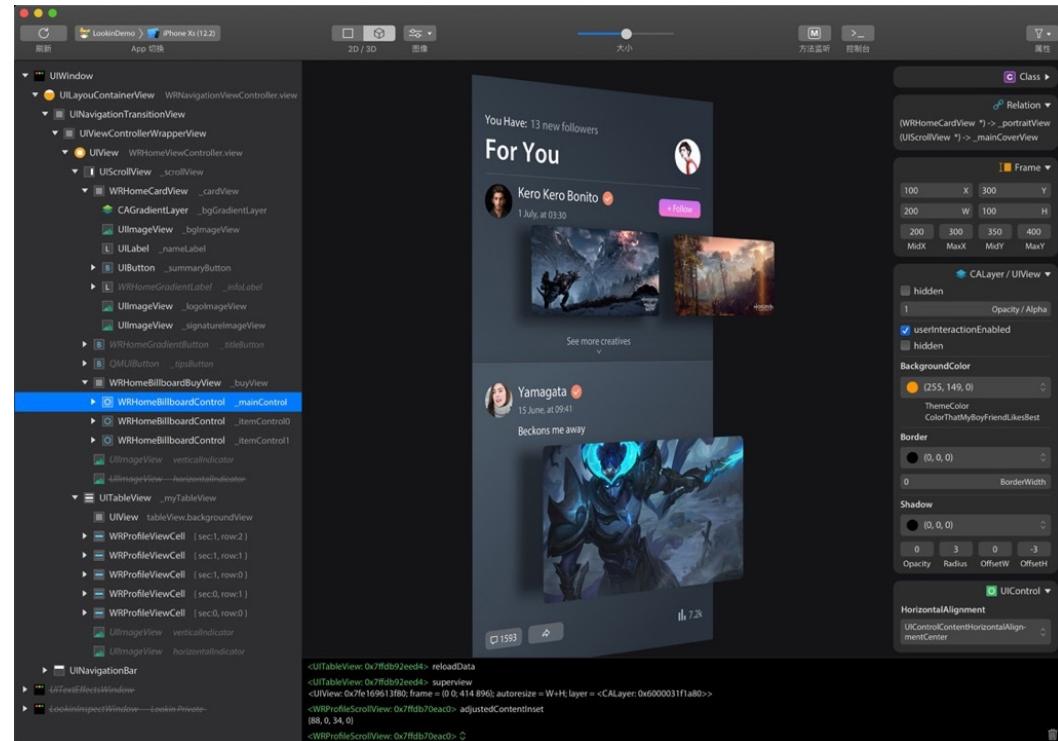
TODO:

- 【整理】页面元素调试结果对比：Reveal、Cycrypt、LLDBTools、chisel
- 【记录】XCode+MonkeyDev动态调试抖音：从点赞关注UI界面入手找底层代码逻辑

iOS逆向的动态调试，也常会，从app的界面入手找对应的按钮等元素，此时就会涉及到：调试界面元素

常用的iOS的app的界面调试工具：

- Reveal
- Cycrypt
- (MonkeyDev的) LLDBTools
- chisel
- FLEX
- 其他
  - LookinLoader
    - <https://github.com/creantan/LookinLoader>



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：  
2022-10-23 16:20:14

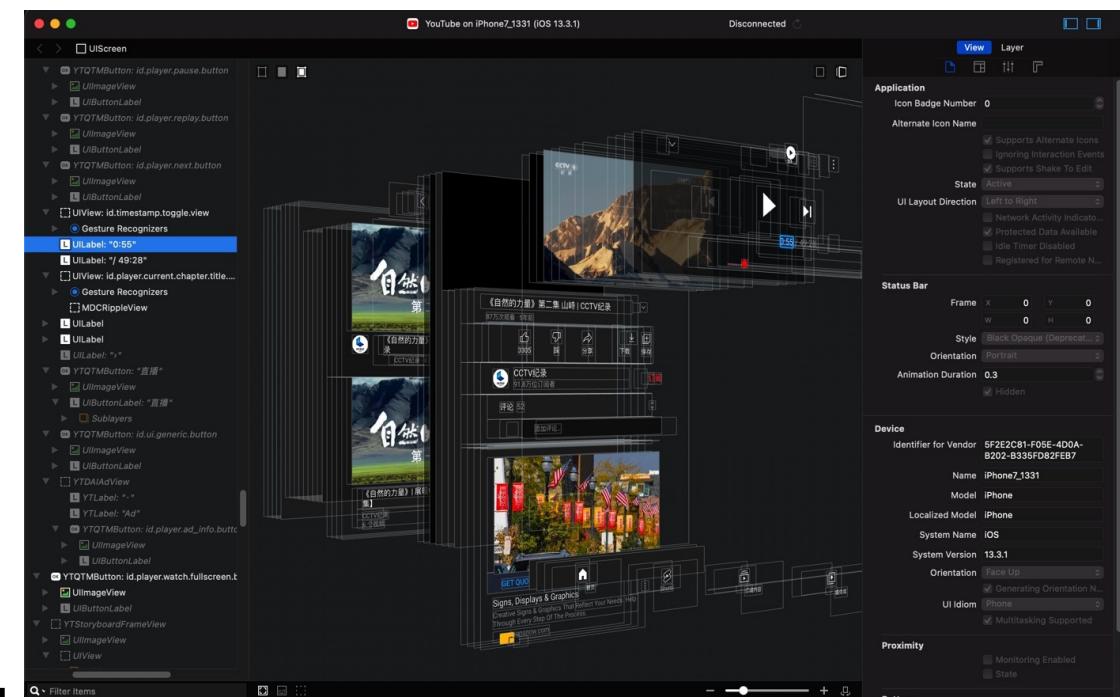
# Reveal

TODO:

- 【已解决】用Reveal查看抖音UI界面中点赞关注按钮相关的类和实现
- 【记录】找抖音关注按钮响应事件: pactions
- 【记录】通过Reveal查看页面元素找YouTube广告相关类
- 【记录】通过Reveal查看YouTube广告页面元素

iOS逆向中，用来调试界面元素，比较好用的工具之一就是： Reveal

- 效果举例
  - YouTube



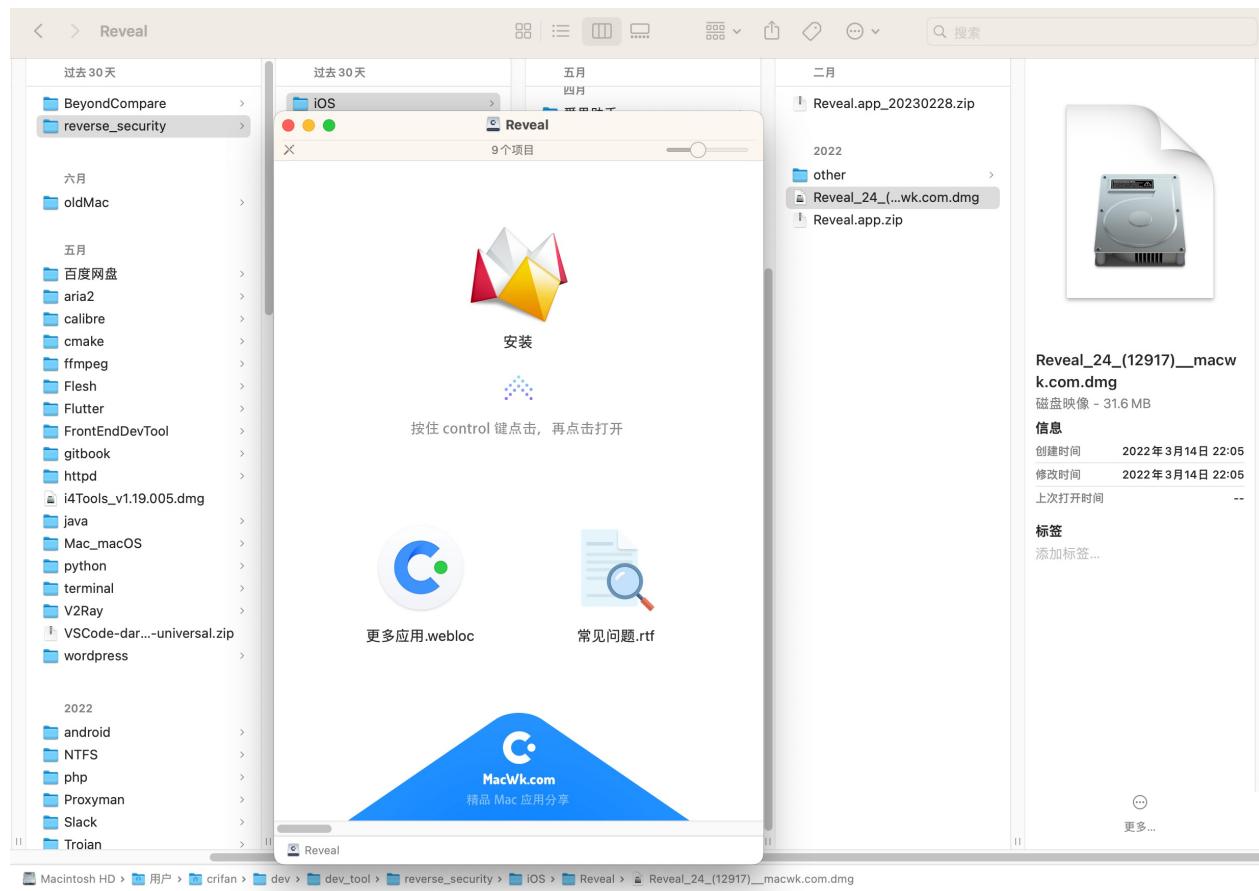
## Xcode+MonkeyDev配合Reveal调试UI界面元素

### 下载安装Reveal

从网上下载到Reveal的dmg

注：<https://www.macwk.com> 网站2022年10月5日已关站，无法访问

然后安装：



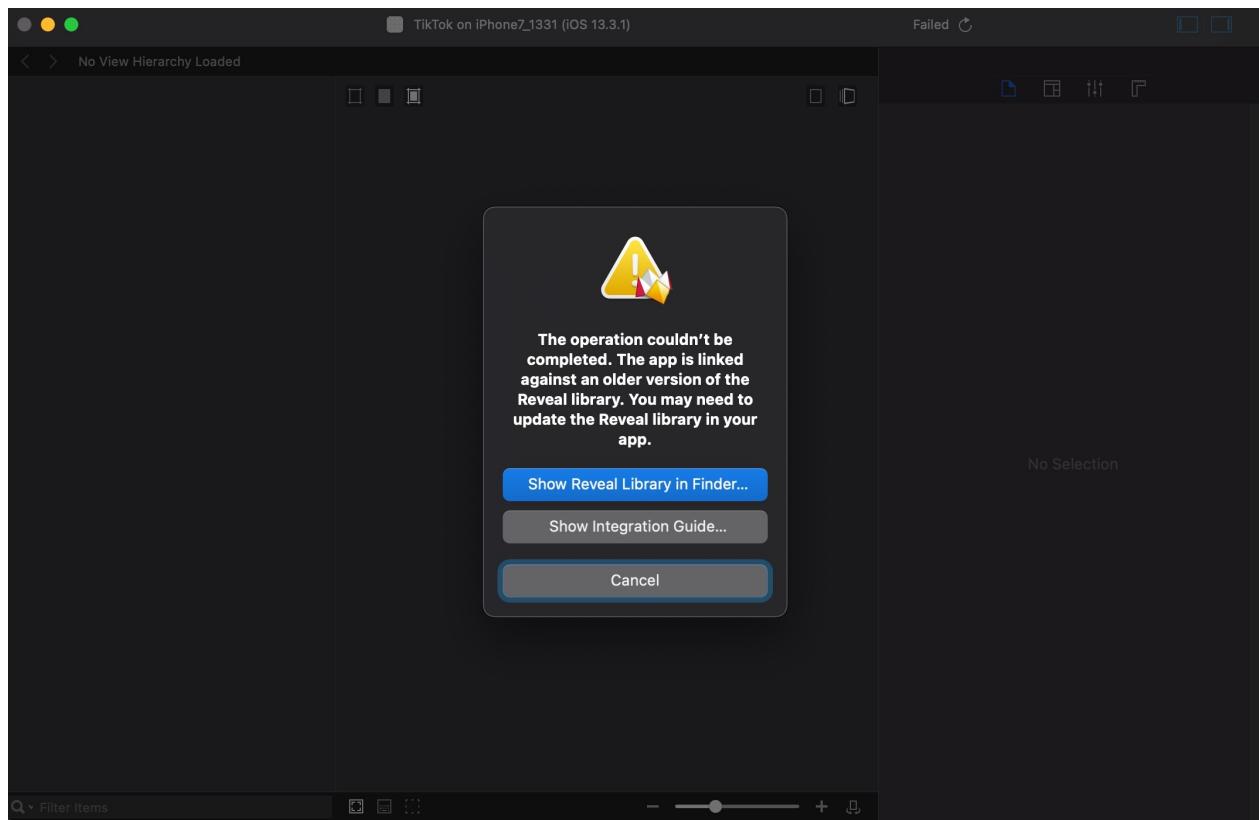
## 确保 RevealServer.framework 版本一致

- Mac : Reveal.app 中的 RevealServer.framework
  - 默认位置: /Users/{YourUserName}/Library/Application Support/Reveal/RevealServer/iOS/RevealServer.framework
- iPhone : 所运行的是 MonkeyDev 内部集成的 RevealServer.framework
  - 默认位置: /opt/MonkeyDev/Frameworks/RevealServer.framework

要确保版本一致。

否则 Reveal.app 连接 iPhone 调试时会报错：

The operation couldn't be completed. The app is linked against an older version of the Reveal library. You may need to update the Reveal library in your app.



解决办法：

点击弹框中的：`Show Reveal Library in Finder...`，会自动打开（当前 Mac 中）最新版本的 `RevealServer.framework`

```
/Users/{YourUserName}/Library/Application  
Support/Reveal/RevealServer/iOS/RevealServer.framework
```

然后将其拷贝过去，替换掉旧的 `MonkeyDev` 的：

```
/opt/MonkeyDev/Frameworks/RevealServer.framework
```

即可。

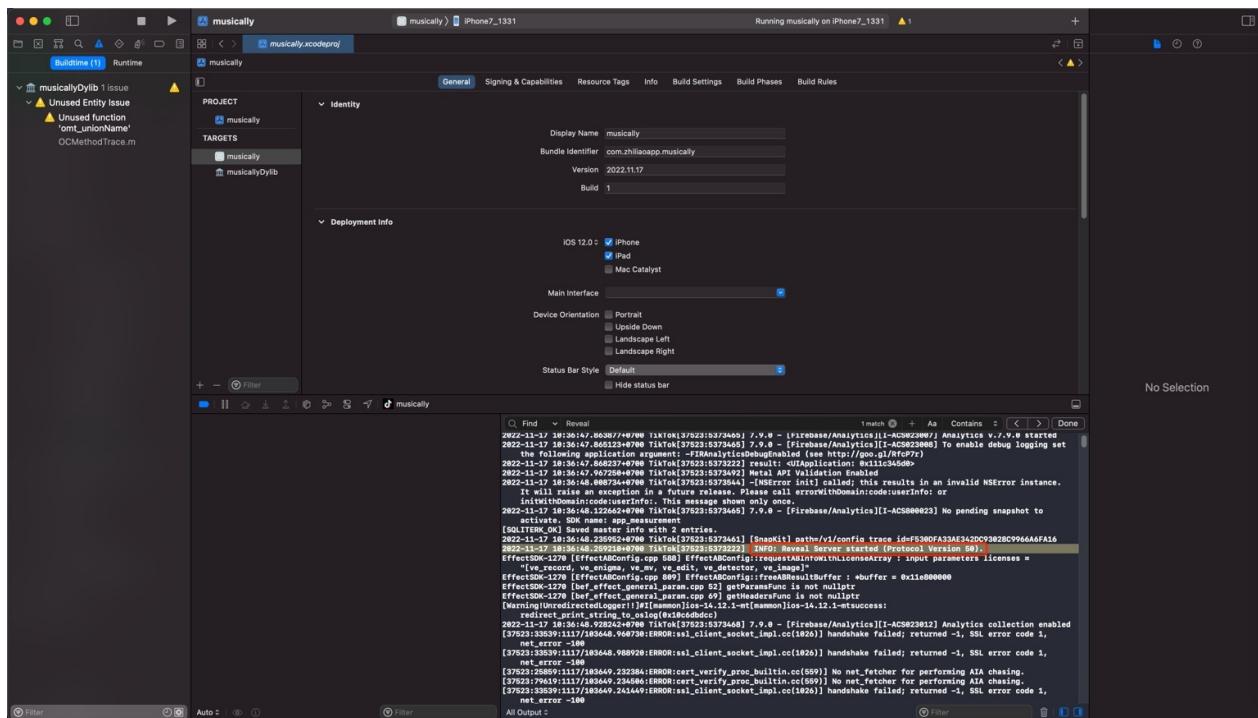
注：

- `/opt/MonkeyDev/Frameworks/`
  - 是 `MonkeyDev` 的常见的默认的安装路径
- 复制时需要root权限
  - 所以命令行复制时，需要 `sudo`，否则会报错没有权限
  - Finder 界面中复制时，需要输入当前 Mac 用户的密码

## 用 XCode + MonkeyDev 调试 iOS 的 app ( ipa )

其中 Xcode 中能输出=能搜到对应的 log：

```
2022-11-17 10:36:48.259210+0700 TikTok[37523:5373222] INFO: Reveal Server started (Protocol Version 50).
```



表示Reveal Server服务已启动

注意：

- 确保最后一条Reveal的log是Started
  - 》意思是Reveal的确在运行
  - 否则也可能遇到，中间Reveal是Started，但之后还有Stopped的log，则表示Reveal服务是停止掉了
  - 那样的话，Reveal是无法使用的

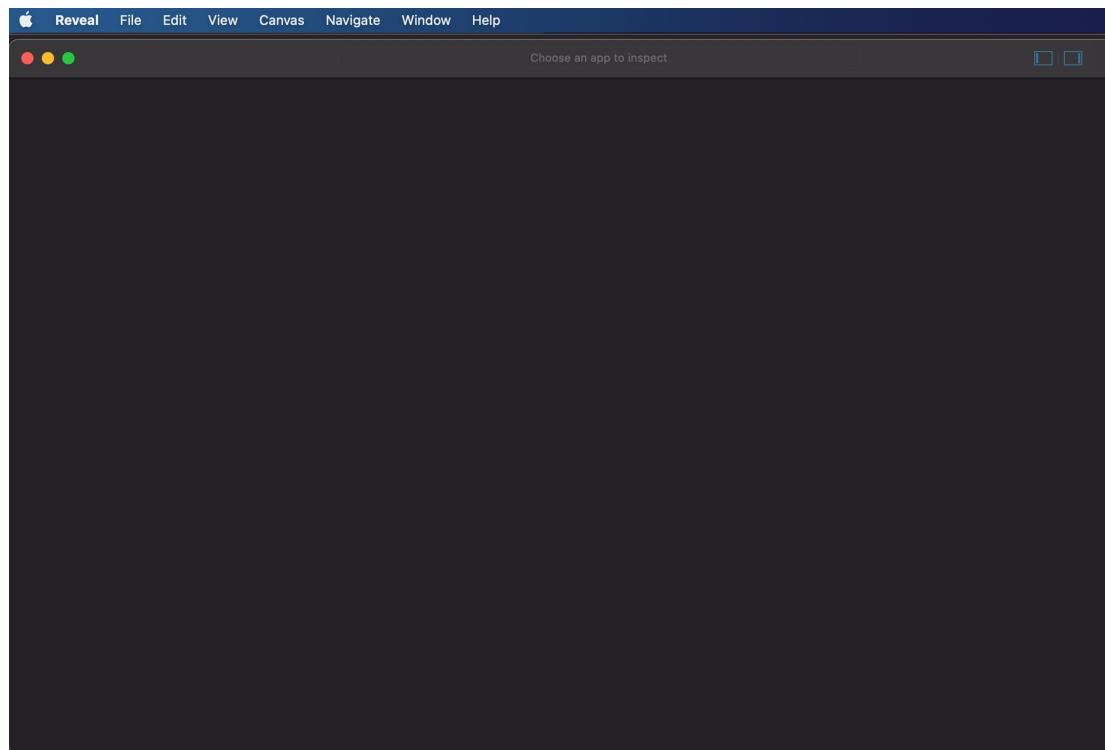
## 打开Mac中的 Reveal.app ，去连接和调试设备中的app的界面

Reveal -> File -> New Tab

点击 Discovered 所显示出iPhone设备了

注意：

- 首次启动Reveal后，往往看不到iPhone设备（中的app）
  - 图

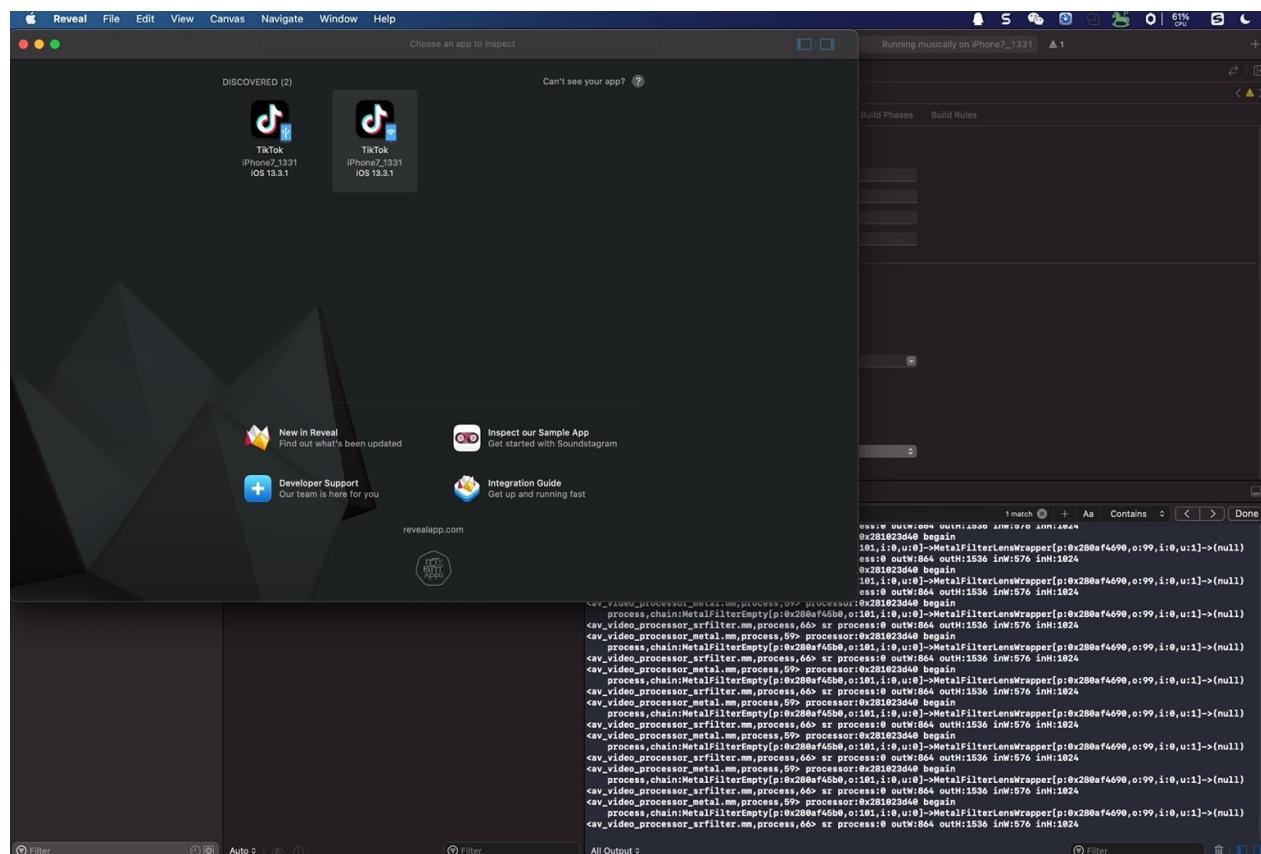


- 解决办法：
    - 关闭Reveal，重启Reveal，即可。

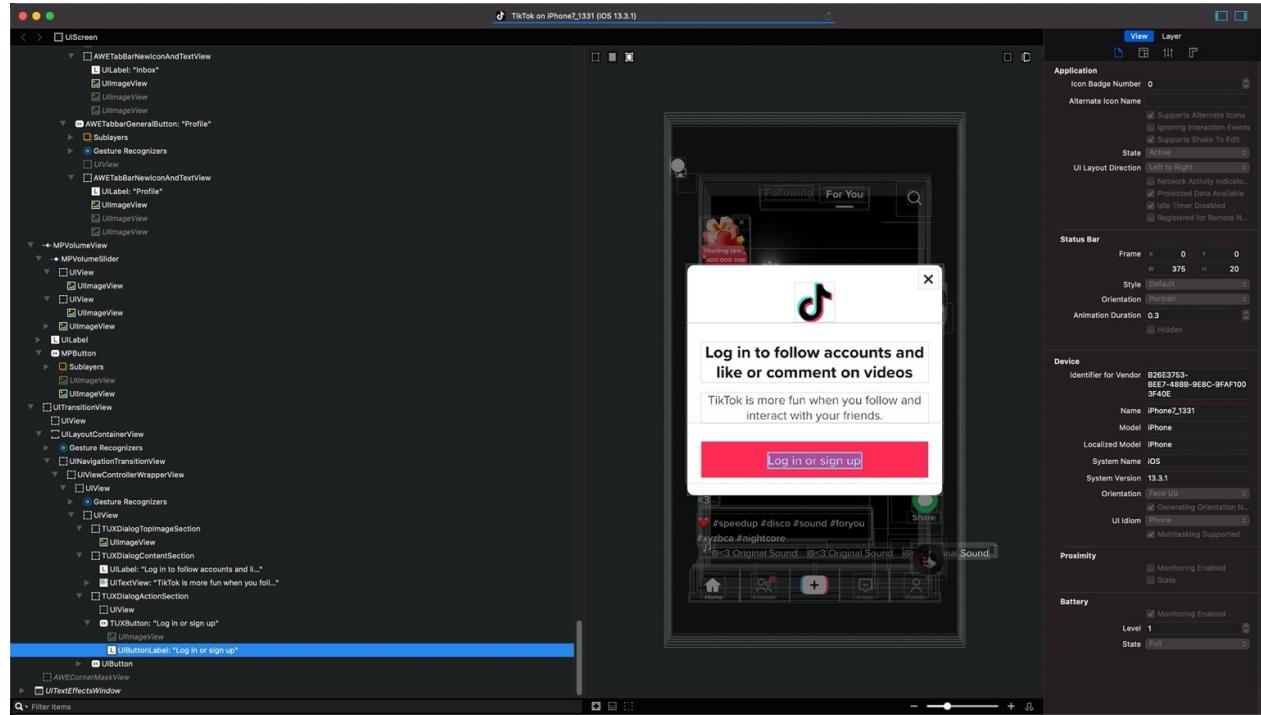
一般正常会出现2个按钮：

- Wifi
  - USB

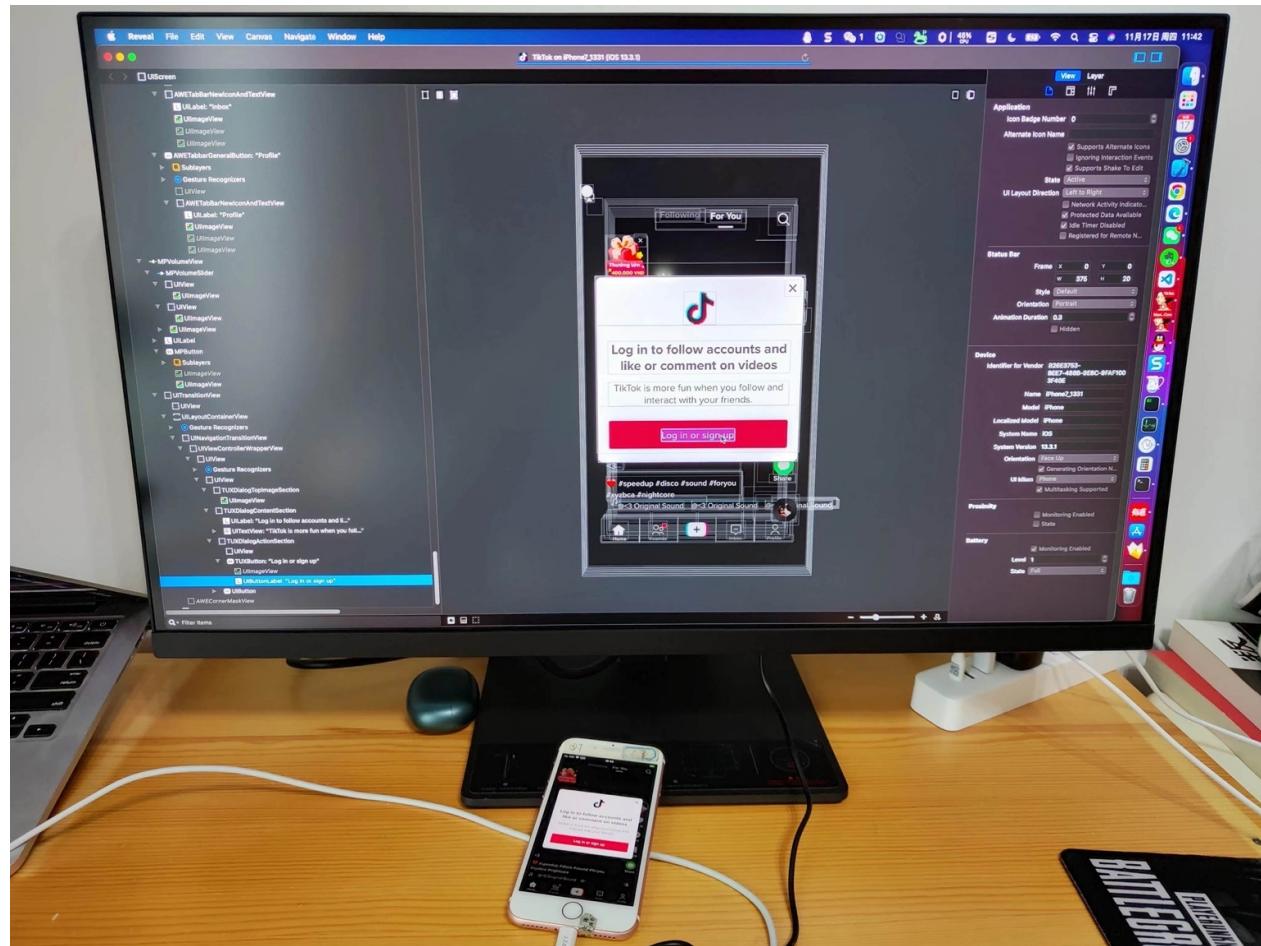
按道理USB的连接更稳定些，所以一般点击USB的



即可连接和正常调试app的UI界面元素了：

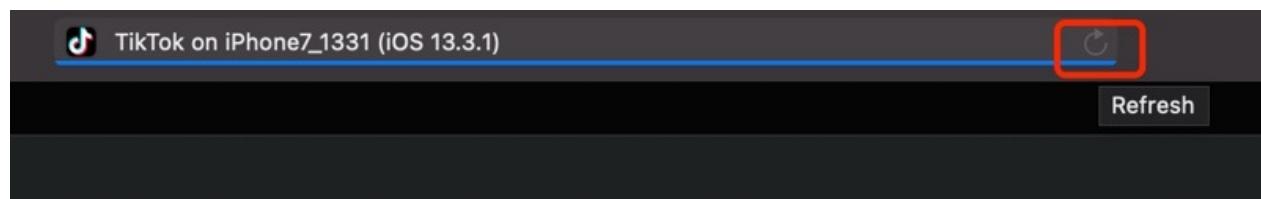


Mac电脑+iPhone手机的效果：



## 刷新页面

如果app端页面刷新了，可以点击Tab顶部的右上角的刷新按钮，即可刷新



注：

此处Refresh按钮是灰色的，原因是：此处Tiktok的app的UI界面元素内容太多，导致一直在加载，始终加载不能完全结束，所以无法刷新

不过一般无所谓，可以重新关掉窗口，重新点击连接设备，从而分析app上最新的界面元素的。

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：  
2023-07-10 09:43:11

# Cycript

TODO:

- 【部分解决】用Cycript查看抖音UI界面元素以寻找关注按钮所属元素
  - 【已解决】用MonkeyDev中Cycript去调试YouTube的UI页面的元素
- 

iOS逆向的调试界面元素的工具，也有命令行的： `cycript`

- Cycript
  - 官网
    - <http://www.cycript.org/>
    - 文档
      - <http://www.cycript.org/manual/>
  - 有用资料
    - [Cycript Tricks - iPhone Development Wiki](#)

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：  
2023-03-15 23:01:56

## Cycript初始化环境

- Cycript环境搭建=初始化
  - Sileo / Cydia 中安装 cycript 插件即可
    - 步骤
      - 如果没有，需要先添加软件源：
        - <https://strap.palera.in/>
      - 然后去搜索： cycript ，并安装，即可
    - 效果



**Cycript**

Jay Freeman (saurik)

更改

详情

更新日志

runtime execution server and disassembler

## 软件源



palera1n strap



## 已安装的软件包

版本

0.9.594-procursus1

显示软件包内容



crypt (0.9.594-procursus1)



精选



新闻



软件源



软件包



搜索

无 SIM 卡

下午 5:37



&lt; 返回

## 已安装的文件

▼ /

▼ usr/

▼ bin/

cycrypt

▼ lib/

▼ cycrypt0.9/

► com/

► org/

libcrypt.cy

libcrypt.db

libcrypt.dylib

libcrypt.0.dylib



精选



新闻



软件源



软件包



搜索

▪ 说明

- Sileo中安装cycrypt会自动找到并安装各种依赖
  - adv-cmds



**adv-cmds**

Procursus Team

更改

详情

更新日志

cap\_mkdb, colldef, finger, gencat, last, locale,  
lsvfs, tabs

## 软件源



palera1n strap >

## 已安装的软件包

版本

199.0.1

显示软件包内容 >

adv-cmds (199.0.1)



精选



新闻



软件源

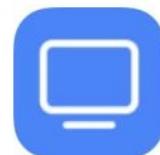


软件包



搜索

- Substitute



**Substitute**  
comex

更改

详情

更新日志

Substrate substitute for code substitution

## 软件源



palera1n



## 已安装的软件包

版本

2.3.1+9.g200ddd5

显示软件包内容



com.ex.substitute (2.3.1+9.g200ddd5)



精选



新闻



软件源



软件包



搜索

- 桌面图标: Substitute



■ Substrate Safe Mode



## Substrate Safe Mode

Jay Freeman (saurik)

更改

详情

更新日志

safe mode safety extension (safe)

查看介绍 >

### 软件源



palera1n

>

>

### 已安装的软件包

版本

0.9.6005

显示软件包内容 >

com.saurik.substrate.safemode (0.9.6005)



精选



新闻



软件源



软件包



搜索

## 安装Cycrypt后

可以找到对应二进制文件：

```
iPhone8-150:~ root# which cycrypt
```

```
/usr/bin/cycript
```

查看基本语法：

```
iPhone8-150:~ root# cycript --help
cycript: unrecognized option '--help'
usage: cycript [-c] [-p <pid name>] [-r <host:port>] [<script> [<arg>...]]
```

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：  
2023-03-15 22:57:08

## Cycript的基本用法

```
cycript -p PID_or_AppName
```

进入 cy# 开头的命令行界面，即表示注入成功，可以开始调试了

## Cycript中常用命令

### 调试ObjC对象的命令

```
[UIApplication sharedApplication]
UIApp.keyWindow.recursiveDescription().toString()

var topView = [[[UIApplication sharedApplication] keyWindow] subviews] lastObject
[topView recursiveDescription].toString()

var p = new Instance(0x157d1e200)
```

### 打印最顶层页面/窗口

背景知识是，iOS的ObjC的获取最顶层的窗口：

```
[[[[UIApplication sharedApplication] keyWindow] subviews] lastObject]
```

放到Cycript中：

```
[[[[UIApplication sharedApplication] keyWindow] subviews] lastObject] recursiveDescription].toString()
```

进一步优化：

写成变量，便于后续引用：

```
var topView = [[[UIApplication sharedApplication] keyWindow] subviews] lastObject
[topView recursiveDescription].toString()
```

### 打印页面详情

已有视图view：

```
cy [[[UIApplication sharedApplication] keyWindow] subviews] lastObject
#"<UITransitionView: 0x11f9059e0; frame = (0 0; 375 667); autoresize = W+H; layer = <CALayer: 0x280129300>"
```

去打印页面详情，以字符串输出，是：

- 先： recursiveDescription
- 再： toString

即：

```
var topView = [[[UIApplication sharedApplication] keyWindow] subviews] lastObject]
[topView recursiveDescription].toString()
```

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：  
2023-03-15 22:58:19

## Cycript使用心得

### 找按钮的响应函数=处理函数

- 背景

对于页面



取消

下一步

# Apple ID

使用 Apple ID 登录以使用 iCloud 和其他  
Apple 服务。

Apple ID

.com

密码

[没有或忘记 Apple ID?](#)



Apple ID 是您用于访问所有 Apple 服务的帐  
户。iCloud 使用无线数据。



您的 Apple ID 信息用于登录时启用 Apple 服务，其中包括 iCloud 云备  
份，该服务可自动备份设备上的数据，以便需要时进行替换或恢复。  
您的设备序列号可能被用于检查服务的使用资格。 [了解数据的管理方  
式....](#)

中右上角的 下一步 按钮

想要去找，点击之后所触发的对应的处理函数

- 核心思路
- 主要过程和结论

先搞清楚下一步按钮：

```
UIButtonLabel: 0x107dacb10; frame = (0 1; 52 20.5); text = '下一步'; opaque = NO; userInteractionEnabled = NO; layer = <UILabelLayer: 0x2823f0460>
```

的上2级的元素：

```
_UIButtonBarButton: 0x107d97600; frame = (0 0; 60 44); tintColor = <UIDynamicSystemColor: 0x2815a6dc0; name = systemBlueColor>; gestureRecognizers = <NSArray: 0x280fefdb0>; layer = <CALayer: 0x2801f8560>>
```

-》 对应的：

```
cy# var nextStepBtn2 = #0x107d97600
#"<_UIButtonBarButton: 0x107d97600; frame = (0 0; 60 44); tintColor = <UIDynamicSystemColor: 0x2815a6dc0; name = systemBlueColor>; gestureRecognizers = <NSArray: 0x280fefdb0>; layer = <CALayer: 0x2801f8560>>"

cy# [nextStepBtn2 allTargets]
[NSSet setWithArray:@[#<_UIButtonBar: 0x283fb7cf0> <_UIButtonBarStackView: 0x107da4c90
; frame = (307 6; 60 44); layer = <CALayer: 0x2800665e0>> buttonBar=0x283fb7cf0\nmetric
s=0x2815caccc0 layout=0x280f60270 groupLayouts=0x102b1aa70 views=0x280f9af70 guides=0x28
0f99bc0 activeConstraints=0x280f602d0 minimumInterItemSpace=8.000 minimumInterItemSpace
Anchor=0x2823c34d0 flexibleSpaceEqualSizeAnchor=0x2815c89c0 minimumInterGroupSpaceAncho
r=0x2823c1270\nbarButtonItemGroups={\n<UIBarButtonItemGroup: 0x2823c3070> barButtonItems={\n
\t<UIBarButtonItem: 0x107d75ef0> target=0x107d80750 action=_nextButtonSelected: title=
'\\xe4\\xb8\\x8b\\xe4\\xb8\\x80\\xe6\\xad\\xa5'\n} }\n} ,#"<_UIButtonBarTargetAction: 0x2801fbdc0
>"]]
```

中的：

- \_nextButtonSelected:

## Cycript常见问题

### 偶尔卡死

现象： cycript -p Preferences 卡死

原因： 偶尔的bug或者其他未知原因

解决办法： 多试几次。

包括但不限于：

- 确保设置页面处于前台
- 多运行几次命令
- 打开设置页面，点击进入子页面再返回等等操作

就可以了。

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：  
2023-03-15 23:01:36

## Cycript输出举例

Cycript命令行输出的内容，尤其是对于页面详情，往往输出内容很多。

此处举例说明输出内容大概长什么样：

## Cycript输出效果举例

当前页面：



取消

下一步

# Apple ID

使用 Apple ID 登录以使用 iCloud 和其他  
Apple 服务。

**Apple ID**

电子邮件或电话

[没有或忘记 Apple ID?](#)



Apple ID 是您用于访问所有 Apple 服务的帐  
户。iCloud 使用无线数据。



您的 Apple ID 信息用于登录时启用 Apple 服务，其中包括 iCloud 云备  
份，该服务可自动备份设备上的数据，以便需要时进行替换或恢复。  
您的设备序列号可能被用于检查服务的使用资格。 [了解数据的管理方  
式...](#)

输出结果：

```

cy# var topView = [[[UIApplication sharedApplication] keyWindow] subviews] lastObject]
#<UITransitionView: 0x11f9059e0; frame = (0 0; 375 667); autoresize = W+H; layer = <CALayer: 0x280129300>>

cy# [topView recursiveDescription].toString()
`-<UITransitionView: 0x11f9059e0; frame = (0 0; 375 667); autoresize = W+H; layer = <CALayer: 0x280129300>>
    | <UIDimmingView: 0x107eac170; frame = (-375 -667; 1125 2001); opaque = NO; gestureRecognizers = <NSArray: 0x280f2c960>; layer = <CALayer: 0x280129340>>
    | <UIDropShadowView: 0x107ed9e70; frame = (0 40; 375 627); gestureRecognizers = <NSArray: 0x280f2f810>; layer = <CALayer: 0x2801289c0>>
    |   | <_UIRoundedRectShadowView: 0x107e51ab0; frame = (-150 -150; 675 927); alpha = 0; opaque = NO; autoresize = W+H; userInteractionEnabled = NO; layer = <CALayer: 0x280129180>>
    |   |   | <UIView: 0x107e812f0; frame = (0 0; 375 627); clipsToBounds = YES; autoresize = W+H; layer = <CALayer: 0x280128160>>
    |   |   |   | <UIView: 0x107ed6ff0; frame = (0 0; 375 627); autoresize = W+H; layer = <CALayer: 0x280128c00>>
    |   |   |   |   | <UILayoutContainerView: 0x107ece570; frame = (0 0; 375 627); clipsToBounds = YES; autoresize = W+H; gestureRecognizers = <NSArray: 0x280f2f060>; layer = <CALayer: 0x280128940>>
    |   |   |   |   |   | <UINavigationTransitionView: 0x107e3fc00; frame = (0 0; 375 627); clipsToBounds = YES; autoresize = W+H; layer = <CALayer: 0x280128560>>
    |   |   |   |   |   | <UIViewControllerWrapperView: 0x107dd34a0; frame = (0 0; 375 627); layer = <CALayer: 0x2801f4200>>
    |   |   |   |   |   |   | <AAUIBuddyView: 0x107d85760; frame = (0 0; 375 627); layer = <CALayer: 0x2801f5a60>>
    |   |   |   |   |   |   | <UITableView: 0x1081a0600; frame = (0 0; 375 627); gestureRecognizers = <NSArray: 0x280fd3de0>; layer = <CALayer: 0x2801f6760>; contentOffset: {0, -56}; contentSize: {375, 524.5}; adjustedContentInset: {56, 0, 0, 0}; dataSource: <AAUISignInViewController: 0x107ec9ab0>
    |   |   |   |   |   |   |   | <UITableViewCell: 0x107d3ed10; frame = (0 155; 375 60); text = 'Apple ID'; autoresize = W; layer = <CALayer: 0x2801f45a0>>
    |   |   |   |   |   |   |   | <UISystemBackgroundView: 0x107dae70; frame = (0 0; 375 60); layer = <CALayer: 0x2801f5040>; configuration = <UIBackgroundConfiguration: 0x283593480; Base Style = List Grouped Cell; backgroundColor = <UIDynamicsSystemColor: 0x2815b4ac0; name = tableCellGroupedBackgroundColor>>
    |   |   |   |   |   |   |   | <UIView: 0x107daf030; frame = (0 0; 375 60); layer = <CALayer: 0x2801f4d80>>
    |   |   |   |   |   |   |   | <UITableViewCellContentView: 0x107dad80; frame = (0 0; 375 60); gestureRecognizers = <NSArray: 0x280fff2a0>; layer = <CALayer: 0x2801f5320>>
    |   |   |   |   |   |   |   | <UILabel: 0x107dadf20; frame = (16 19; 66.5 20.5); text = 'Apple ID'; userInteractionEnabled = NO; layer = <UILabelLayer: 0x2823db250>>
    |   |   |   |   |   |   |   | <UITextField: 0x10814f400; frame = (114 19; 245 22); gestureRecognizers = <NSArray: 0x280ffc630>; placeholder = '\u7535\u5b50\u90ae\u4ef6\u6216\u7535\u8bdd'; borderStyle = None; background = <UITextFieldNoBackgroundProvider: 0x2802b7ef0: textfield=<UITextField 0x10814f400>; layer = <CALayer: 0x2801f5300>>
    |   |   |   |   |   |   |   | <UITextFieldLabel: 0x107dad40; frame = (0 1; 245 20.5); text = '\u7535\u5b50\u90ae\u4ef6\u6216\u7535\u8bdd'; opaque = NO; userInteractionEnabled = NO; layer = <UILabelLayer: 0x2823db1b0>>

```

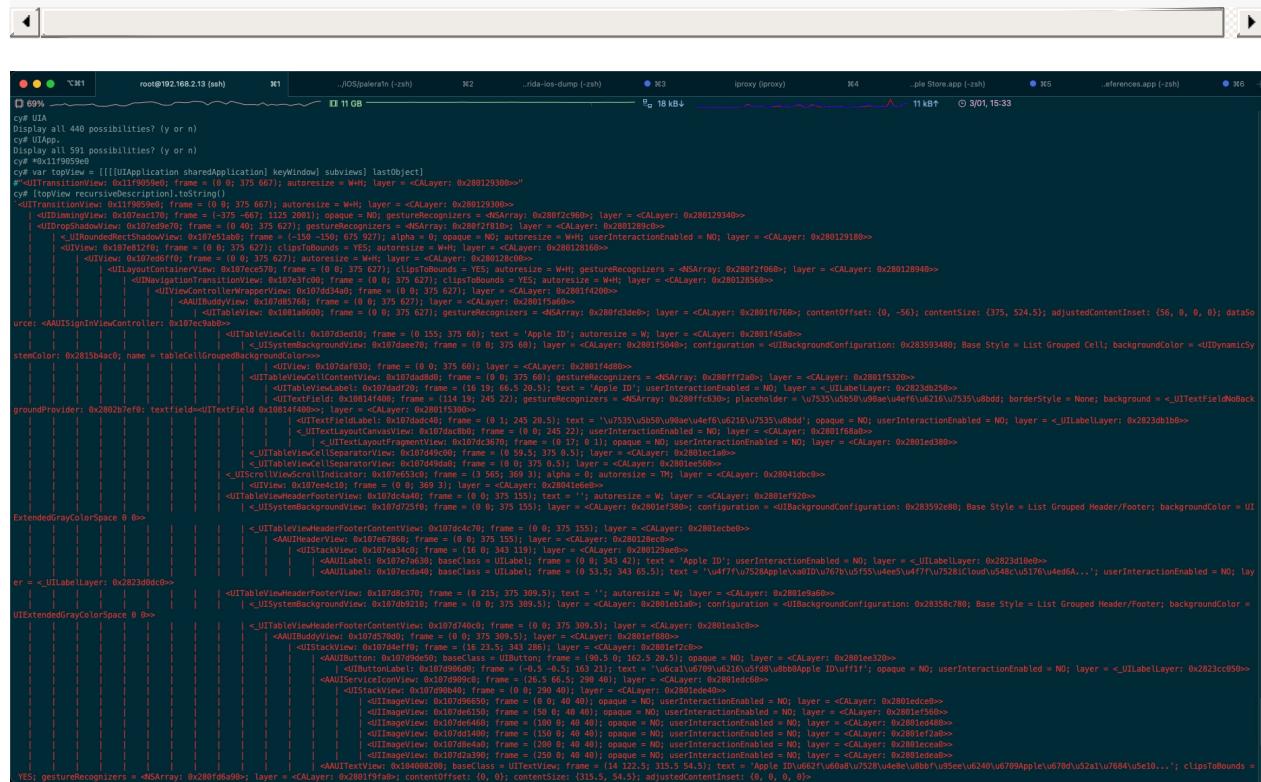


```

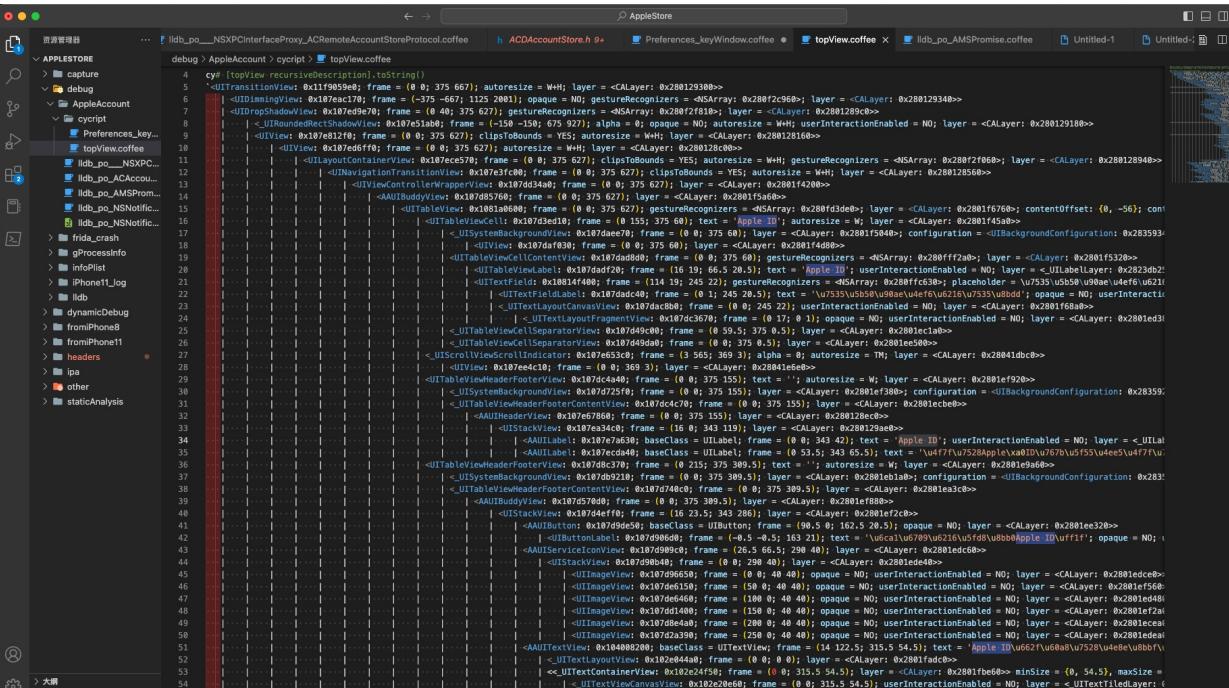
: 0x107d96650; frame = (0 0; 40 40); opaque = NO; userInteractionEnabled = NO; layer = <
CALayer: 0x2801edce0>>
: 0x107de6150; frame = (50 0; 40 40); opaque = NO; userInteractionEnabled = NO; layer = <
CALayer: 0x2801ef560>>
: 0x107de6460; frame = (100 0; 40 40); opaque = NO; userInteractionEnabled = NO; layer = <
CALayer: 0x2801ed480>>
: 0x107dd1400; frame = (150 0; 40 40); opaque = NO; userInteractionEnabled = NO; layer = <
CALayer: 0x2801ef2a0>>
: 0x107d8e4a0; frame = (200 0; 40 40); opaque = NO; userInteractionEnabled = NO; layer = <
CALayer: 0x2801ececa0>>
: 0x107d2a390; frame = (250 0; 40 40); opaque = NO; userInteractionEnabled = NO; layer = <
CALayer: 0x2801edea0>>
: 0x104008200; baseClass = UITextView; frame = (14 122.5; 315.5 54.5); text = 'Apple ID\u662f\u60a8\u7528\u4e8e\u8bbf\u95ee\u6240\u6709Apple\u670d\u52a1\u7684\u5e10...'; clipsToBounds = YES; gestureRecognizers = <NSArray: 0x280fd6a90 ; layer = <CALayer: 0x2801f9fa0>; contentOffset: {0, 0}; contentSize: {315.5, 54.5}; adjustedContentInset: {0, 0, 0, 0}>
<_UITextView: 0x102e044a0; frame = (0 0; 0 0); layer = <CALayer: 0x2801fadcc0>>
<_UITextContainer: 0x102e24f50; frame = (0 0; 315.5 54.5); layer = <CALayer: 0x2801fbe60>> minSize = {0, 54.5}, maxSize = {315.5, 54.5}, textContainer = <NSTextContainer: 0x283188000 size = (315.500000,38.500000); widthTracksTextView = YES; heightTracksTextView = YES>; exclusionPaths = 0x1f78db500; lineBreakMode = 0>
<_UITextView: 0x102e20e60; frame = (0 0; 315.5 54.5); userInteractionEnabled = NO; layer = <_UITextTiledLayer: 0x2832ae7f0>>
<UIScrollViewScrollIndicator: 0x107ec3260; frame = (3 48.5; 309.5 3); alpha = 0; autoresizingMask = TM; layer = <CALayer: 0x28041c3e0>>
<UIView: 0x107eb14b0; frame = (0 0; 309.5 3); layer = <CALayer: 0x28041d5e0>>
<UIScrollViewScrollIndicator: 0x107e518f0; frame = (309.5 3; 3 48.5); alpha = 0; autoresizingMask = LM; layer = <CALayer: 0x28041d600>>
<UIView: 0x107ead220; frame = (0 0; 3 48.5); layer = <CALayer: 0x28041c2a0>>
<OBPrivacyLinkButton: 0x107d6a250; baseClass = UIButton; frame = (0 193; 343 93); opaque = NO; layer = <CALayer: 0x2801e8a60>>
<UIView: 0x107d6a550; frame = (0 0; 343 93); userInteractionEnabled = NO; layer = <CALayer: 0x2801ea7c0>>
<UITextView: 0x108214800; frame = (1 29.5; 341.5 63.5); text = '\u60a8\u7684Apple\x01D\u4fe1\u606f\u7528\u4e8e\u767b\u5f55\u65f6\u542f\u7528Apple\u670d...'; clipsToBounds = YES; userInteractionEnabled = NO; gestureRecognizers = <NSArray: 0x280fe3e40 ; layer = <CALayer: 0x2801eaf60>; contentOffset: {0, 0}; contentSize: {341.5, 63.5}; adjustedContentInset: {0, 0, 0, 0}>
<_UITex

```





## 拷贝到VSCode中的显示效果



The screenshot shows the Xcode IDE with the project "AppleStore" open. The main editor window displays approximately 55 lines of assembly code, likely generated by the LLVM compiler for an Apple Store application. The assembly code is highly optimized and includes various memory management instructions, control flow, and data processing. The assembly is annotated with comments and labels, such as `__UITransitionView`, `__UITableCellSeparatorView`, and `__AUTUITextView`. The assembly code is presented in a standard assembly format with mnemonics like `add`, `sub`, `mul`, and `div`, along with their corresponding assembly opcodes and addresses.

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：  
2023-03-15 23:01:47

## LLDBTools

TODO:

- 【已解决】用MonkeyDev的LLDBTools去打印UI界面元素
- 

iOS逆向调试界面元素时，也可以用：[MonkeyDev](#)的 LLDBTools 的相关命令，输出界面元素信息。

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：  
2023-07-10 09:49:27

## chisel

chisel 本身是 [主流调试器: LLDB](#) 的插件，其中有部分命令，也可以用来，调试打印界面元素。

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:  
2023-07-13 21:58:38

# FLEX

TODO:

【整理】iOS的iPhone越狱和改机相关知识

---

iOS越狱插件 FLEX，可以用来辅助调试iOS的app的界面元素。

- FLEX
  - 效果
    - 当它加载时，会向目标程序上方添加一个悬浮的工具栏，通过这个工具栏可以查看和修改视图的层级结构、动态修改类的属性、动态调用实例和方法、动态查看类和框架以及动态修改UI等。
  - 截图
    -

## 教程



Word 文档

Excel 表格

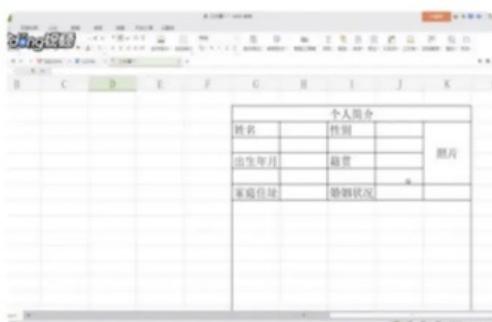
PPT 幻灯片

## 热门教程



将文档中的表格转化为...  
存档方便快捷

如何让你的图表动起来  
让你的工作报表更新颖



用表格也能制作简历  
快速晋升商务办公达人



01 语法：  
简单易懂不会忘，1个逻辑套用10个公式

震惊！PPT 究竟有多么...  
PPT 进阶





## View Hierarchy Tree

完成

Filter

## ● UIWindow

frame {(0, 0), (414, 736)}



## ● UITransitionView

frame {(0, 0), (414, 736)}



## ● UIDropShadowView

frame {(0, 0), (414, 736)}



## ● UILayoutContainerView (wordios.BaseTab)

frame {(0, 0), (414, 736)}



## ● UITransitionView

frame {(0, 0), (414, 736)}



## ● UIViewControllerWrapperView

frame {(0, 0), (414, 736)}



## ● UILayoutContainerView (wordios.BaseNav)

frame {(0, 0), (414, 736)}



## ● UINavigationTransitionView

frame {(0, 0), (414, 736)}



## ● UIViewControllerWrapperView

frame {(0, 0), (414, 736)}



## ● UIView (wordios.CourseVC)

frame {(0, 0), (414, 736)}



## ● UIView

frame {(0, 90), (414, 92)}

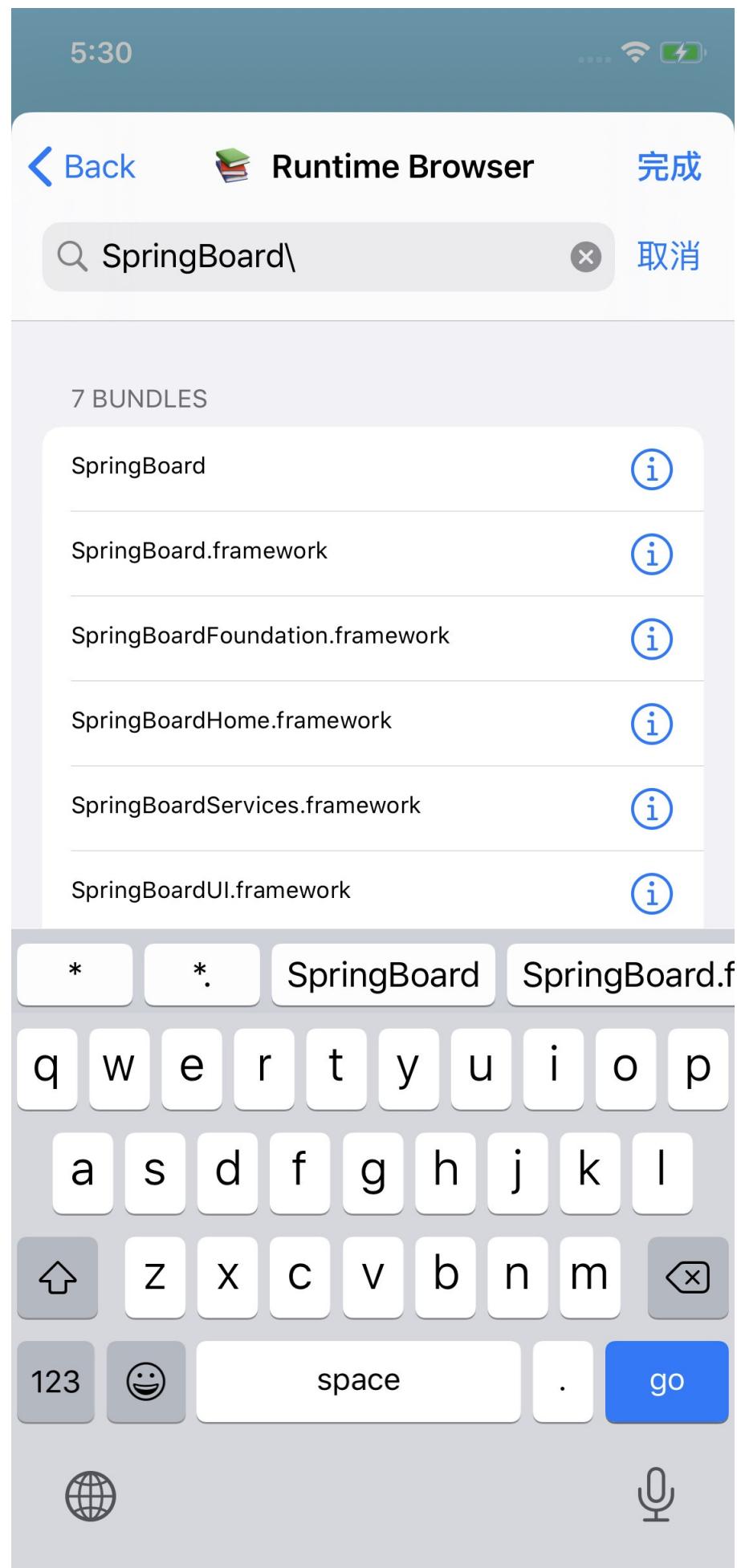


## ● UIView

frame {(0, 0), (138, 92)}



- 好像还可以擦好看类的定义





5:30

... WiFi 🔋

返回 NSBundle 完成

NSBundle NSObject

DESCRIPTION

NSBundle </System/Library/PrivateFrameworks/SpringBoard.framework> (loaded)

SHORTCUTS

Browse Bundle Directory >

Browse Bundle as Database... > •••

@property NSString \*bundleIdentifier  
com.apple.SpringBoardFramework > •••

@property Class principalClass  
nil > •••

@property NSDictionary \*infoDictionary  
{ BuildMachineOSBuild = 1... arm64 ); } >

@property NSString \*bundlePath  
/System/Library/PrivateFrameworks/SpringBoard.framework >

@property NSString \*executablePath  
/System/Library/PrivateFrameworks/SpringBoard.framework/SpringBoard >

@property BOOL loaded  
1 >

PROPERTIES (32)

CCUILayoutSize ccui\_prototypeModuleSize

...

↑

Bookmark

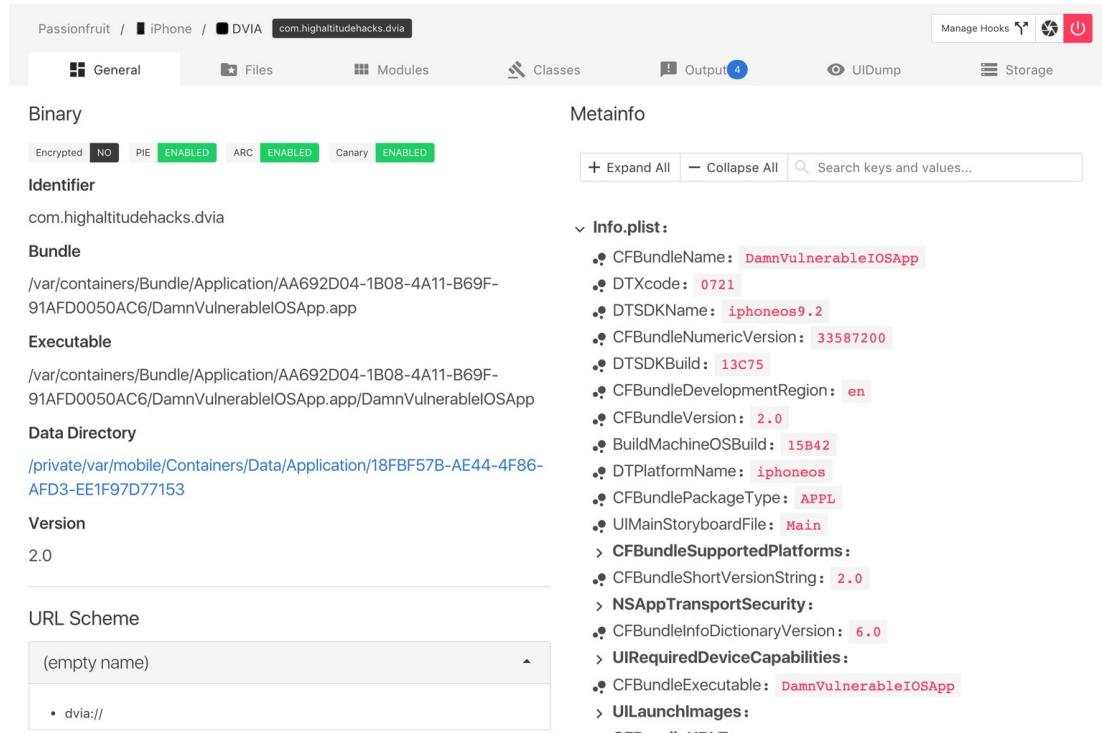
Share

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：  
2022-10-23 16:39:36

# Passionfruit

- Passionfruit
  - 概述:
    - Passionfruit 通过 frida 注入代码到目标应用实现功能，再通过 node.js 服务端消息代理与浏览器通信，用户通过访问网页即可对App实现常规的检测任务
    - Passionfruit 最大特点就是基于Web的图形界面，所以服务端支持跨平台的。
      - 在不少界面都添加了搜索功能，如模块列表、导出符号、Objective-C 类，甚至 Plist 这样的序列化数据

- 截图



- Github

- [chaitin/passionfruit: \[WIP\] Crappy iOS app analyzer](#)
  - Simple iOS app blackbox assessment tool. Powered by frida.re and vuejs.
  - 注：2021年停止维护了
  - [wiki](#)
    - [Screenshots · chaitin/passionfruit Wiki \(github.com\)](#)

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：  
2022-11-10 15:27:42

# 动态调试心得

TODO:

- 动态调试
  - 【已解决】iOS逆向心得：如何从对x8的adrp和ldr计算出对应的qword字符串值
  - 【整理】iOS逆向调试心得：bool等变量类型
  - 【整理】iOS逆向心得：变量类型是bool类型
- 类
  - 【整理】iOS逆向心得：类的属性字段偏移量计算要加上isa的父类
  - 【整理】iOS逆向心得：打印ObjC类的属性
  - 【已解决】iOS逆向：写hook代码时打印出类的私有属性变量值的类型
  - 【整理】iOS逆向调试心得：给类的属性去设置值以及如何计算类的属性的偏移量
  - 【整理】iOS逆向心得：通过查看类的地址保存的值找到值和属性字段的偏移量和对应关系
- 函数
  - 【整理】iOS逆向调试心得：bl函数调用和返回常见逻辑
  - 【整理】iOS逆向心得：ObjC函数调用时参数顺序和汇编代码中寄存器传递的参数顺序不一致
  - 【整理】iOS逆向lldb调试心得：iOS的ObjC的无名汇编跳板函数
    - 相关
      - 【已解决】clang中的\_\_cdecl和支持哪些调用规范
      - 【已解决】微软的调用规范的参数传递和命名规范
      - 【已解决】iOS中调用asm汇编关键字：asm **asm** **asm** 和volatile **volatile**
      - 【已解决】XCode的断点条件判断中如何获取iOS的ObjC函数的参数值
- 计算类的属性的偏移量
  - 【已解决】调试寻找HAMPlayerInternal的\_currentTime中字段的偏移量
  - 【整理】iOS逆向心得：通过查看类的地址保存的值找到值和属性字段的偏移量和对应关系
  - 【已解决】iOS逆向：写hook代码时打印出类的私有属性变量值的类型
  - 【整理】iOS逆向心得：类的属性字段偏移量计算要加上isa的父类
- C++
  - 【整理】iOS逆向心得：Cronet相关的C++的struct结构体类的属性和函数的偏移量计算逻辑
  - 【已解决】iOS逆向：IDA中如何逆向分析C++的vtable
  - 【整理】iOS逆向涉及内容：C++中的vtable
- 其他
  - 【已解决】iOS逆向：查看NSMutableURLRequest的HTTPBody的数据
  - 【已解决】Xcode中调试iOS的app再次报错：Thread EXC\_BREAKPOINT code 1 subcode 0x1bf09c598

iOS逆向的动态调试，有很多心得，整理如下。

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：  
2023-07-10 10:57:07

# iOS逆向动态调试代码逻辑的手段对比

- iOS逆向动态调试代码逻辑的手段对比
  - 图形界面的
    - 基于 Xcode 的
      - MonkeyDev (+ Xcode )
      - 优点
        - 整套环境，方便逆向调试
        - 集成了常用的工具，包括
          - 砸壳等
          - 反调试的相关代码
        - LLDBTools
      - 缺点
        - 对于原有复杂的 entitlement 权限的 app：调试运行起来，会有各种问题
          - 后来确认，是丢失了 entitlement 权限的问题
            - 由于其没完全开源，导致无法轻易解决和规避
          - 每次调试要重新安装 app，对于大的 app，比较耗时
      - iOSOpenDev (+ Xcode ) + 动态断点调试
        - 优点
          - 可以触发断点暂停去查看各种变量值
          - 可以完整利用 iOS 的 objc 底层的机制，实现调试的目的
            - 比如 <`0x281cf640`> 可以去查看po的描述 po [`0x281cf640 _shortMethodDescription`]，可以看到所有的属性
              - 对比： Frida 中，就难以方便的和完整的查看到这些属性
        - 缺点
          - 前提：要 app 或 进程 可调试
            - 注：关于app或进程可调试
              - iOS 15+
                - XinaA15：自带支持，很好
                - 但是只支持 arm64e 的 A12+ 的芯片（比如 iPhone11），而 arm64e 有 PAC，导致 ARM 代码模拟和反编译，会更困难
              - palera1n：的 rootful 越狱，暂时完美支持任意app和进程可调试
                - 但是可通过 codesign 重签名暂时实现单个的app或进程的可调试
            - 改动代码要重新编译安装插件，重新触发调试环境
        - 命令行的
          - iOSOpenDev (+ Xcode )+仅hook输出log日志
            - 优点
              - 只要hook代码写好，可以hook的范围比较广，几乎任何app或进程均可
            - 缺点
              - 只能从Console.app控制台中查看hook的log日志，无法动态调试
          - lldb (+ debugserver )

- 优点
  - 重签名后的 `debugserver`，支持调试任意app或进程
- 缺点
  - 只能用命令行的 `lldb`（无法像 `xcode` 中一样带图形界面使用）
- `Frida`
  - `Frida (+ js 脚本)`
    - 优点
      - 任意app和进程均可调试
      - 改动代码立刻生效
    - 缺点
      - 无法暂停去查看各种变量值
  - `frida-trace`
    - 优点
      - 可以输出带缩进的 `objc` 函数调用关系，且带颜色，易读
    - 缺点
      - 对于输出内容太长，虽然支持导出到日志，但是却丢失了缩进，不利于查看函数调用关系

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：  
2023-07-10 17:15:27

## Xcode相关

TODO:

- Xcode相关
  - 问题
    - 【已解决】XCode调试YouTube报错：Unable to install There was an internal API error
  - 心得
    - 【已解决】XCode+MonkeyDev调试YouTube：如何在广告页面停止供调试

---

## lldb

- 【整理】iOS逆向心得：lldb中打印d的d0寄存器不是double而是data
- 【整理】如何找到Xcode中lldb调试出的无名的函数对应的IDA的伪代码中是哪个函数

## EXC\_BREAKPOINT

- EXC\_BREAKPOINT
  - 【已解决】iOS逆向调试报错EXC\_BREAKPOINT: HAMNetworkRequestResponseEvent的initWithRequest
  - 【已解决】Xcode中调试iOS的app再次报错：Thread EXC\_BREAKPOINT code 1 subcode 0x1bf09c598
  - 【未解决】Xcode调试iOS的YouTube时objc\_msgSend崩溃：Thread EXC\_BREAKPOINT code 1 subcode 0x1bf09c598

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：  
2022-10-27 11:59:18

# ObjC

TODO:

- ObjC
  - 【整理】iOS逆向调试心得：ObjC或ARM中从偏移量中取值的不同写法
  - 【未解决】Xcode调试iOS的Objc时获取self的父类的实例
  - 【整理】iOS逆向心得：ObjC函数调用时参数顺序和汇编代码中寄存器传递的参数顺序不一致
  - 【整理】iOS逆向和IDA使用心得：调用objc\_msgSend时传递给MLPlayerItemQOEErrorEvent的initWithError:fatal:absoluteTime:的参数不够
  - 【整理】iOS逆向心得：打印ObjC类的属性

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：

2022-10-27 12:03:20

## objc\_msgSend

iOS的ObjC的函数调用，比如A函数调用B函数，底层都是通过 `objc_msgSend` 实现的。

所以iOS逆向期间，涉及到最多的，应该就属 `objc_msgSend` 了。

所以关于 `objc_msgSend` 也有很多心得，整理如下。

### 不带`lldb_unnamed_symbol`的无名的bl，往往是更重要的，我们所关注的objc\_msgSend

折腾：

【未解决】研究抖音关注逻辑：`__lldb_unnamed_symbol1588524$$AwemeCore`

期间，调试到目前的心得：

如果是带 `__lldb_unnamed_symbol` 的写法，往往不是主要的，我们所关心的 `objc_msgSend` 函数而无名的 `bl`，往往是重要的，我们所关注的：`objc_msgSend` 的相关调用

举例：

```
0x11427c2c8 <: 336 : bl      0x115ce58fc
```

其实就是：`objc_msgSend`

而其他很多其他的bl：

```
0x11427c2b0 <: 312 : bl      0x11427e920           ; __lldb_unnamed_symbol1588573  
 $$AwemeCore
```

只是个 `jmp_objc_retain`，不是我们关注的重点。

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：  
2022-10-26 16:42:30

## Runtime运行时

TODO:

- ObjC运行时
    - 【记录】iOS中的ObjC的函数: dispatch\_async
    - 【已解决】iOS逆向心得: OS\_dispatch\_data
    - 【已解决】iOS底层函数: objc\_enumerationMutation
- 

详见 子教程:

- [iOS逆向开发：ObjC运行时 \(crifan.org\)](#)

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：  
2023-07-10 09:55:12

## po

TODO:

- po失效
  - 【无需解决】Xcode中lldb调试iOS的ObjC的汇编代码时：偶尔po失效打印不出变量类型
  - 【未解决】Xcode中lldb的po再次失效尝试搞懂内部原因

[LLDB中的调试命令po](#)，也是iOS逆向期间，用的最多的命令：用于查看某个地址，具体是什么（iOS的ObjC的）类。

对于 `po`，也有很多经验和心得，整理如下。

## 类Class 对比 实例Instance

折腾：

【记录】XCode+MonkeyDev动态调试YouTube类：`YTWatchMiniBarController`

期间，可以通过hook代码：

```
hook YTWatchController

// - (void)playbackControllerDidLoadPlayerWithPlaybackData:(id)arg1;
- (void)playbackControllerDidLoadPlayerWithPlaybackData (id)arg1{
    iosLogInfo("arg1=%@", arg1);
    orig;
}

end
```

而输出log：

```
2022 03 27 17 16 50.049397 0800 YouTube[25245:2617390] hook_ youtubeDylib.xm YTWatchController$playbackControllerDidLoadPlayerWithPlaybackData$ arg1=YTPlaybackData: 0x282153e70>
```

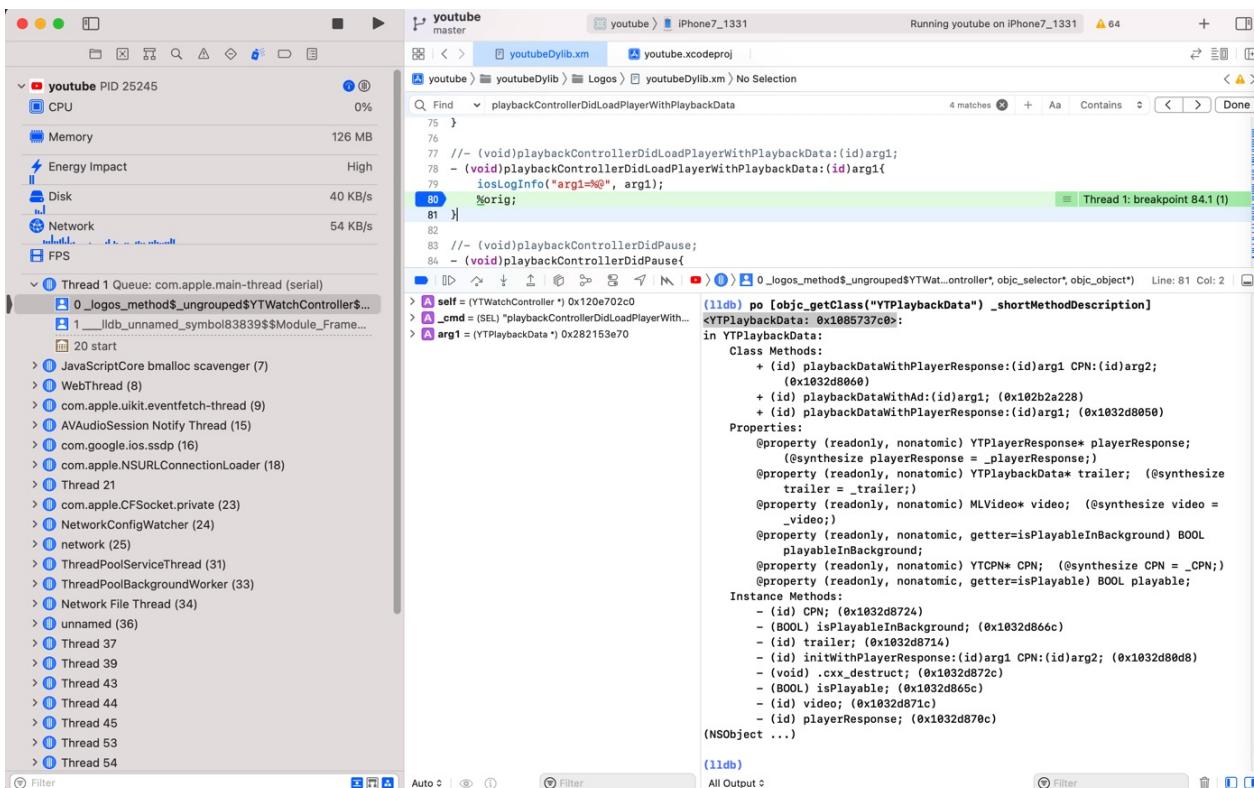
而另外可以通过 `po` 查看到类 `YTPlaybackData` 的信息

```
(lldb) po [objc_getClass("YTPlaybackData") _shortMethodDescription]
YTPlaybackData: 0x1085737c0>
in YTPlaybackData
  Class Methods
    + (id) playbackDataWithPlayerResponse (id)arg1 CPN (id)arg2; (0x1032d8060)
    + (id) playbackDataWithAd (id)arg1; (0x102b2a228)
    + (id) playbackDataWithPlayerResponse (id)arg1; (0x1032d8050)
  Properties
    @property (readonly, nonatomic) YTPlayerResponse playerResponse; (@synthesize pla
```

```

    _playerResponse = _playerResponse;
    @property (readonly, nonatomic) YTPPlaybackData trailer; (@synthesize trailer = _trailer)
    @property (readonly, nonatomic) MLVideo video; (@synthesize video = _video)
    @property (readonly, nonatomic, getter=isPlayableInBackground) BOOL playableInBackground;
    @property (readonly, nonatomic) YTCPN CPN; (@synthesize CPN = _CPN)
    @property (readonly, nonatomic, getter=isPlayable) BOOL playable;
    Instance Methods:
    - (id) CPN; (0x1032d8724)
    - (BOOL) playableInBackground; (0x1032d866c)
    - (id) trailer; (0x1032d8714)
    - (id) initWithPlayerResponse (id)arg1 CPN (id)arg2; (0x1032d80d8)
    - (void) cxx_destruct; (0x1032d872c)
    - (BOOL) playable; (0x1032d865c)
    - (id) video; (0x1032d871c)
    - (id) playerResponse; (0x1032d870c)
    (NSObject ...)

```



两者对比：

- Class = 类

- 无需遇到对应类的变量，任何时候，只要代码加载到内存了，即可查看具体的内容
- 查看类的信息的方式：

```
po [objc_getClass("YTPPlaybackData") _shortMethodDescription]
```

- 场景举例

- 比如给 YouTube 加了断点 `UIApplicationMain`，断点生效时，即可查看类 `YTPPlaybackData` 的信息，而无需实际调试找到 `YTPPlaybackData` 的实例变量

- `Instance = 实例`
  - 只有遇到对应的变量类型了，才能看到具体的值
    - 比如，此处是运行到函数playbackControllerDidLoadPlayerWithPlaybackData的内部，`YTPlaybackData`作为参数，所以才能看到具体的`Instance`实例的值
  - 查看示例变量值的方式
    - 举例

```
(lldb) po [(YTPlaybackData*) 0x282153e70 isPlayable]
true

(lldb) po [(YTPlaybackData*) 0x282153e70 isPlayableInBackground]
false

(lldb) po [(YTPlaybackData*) 0x282153e70 video]
MLVideo: 0x282149da0

(lldb) po [(YTPlaybackData*) 0x282153e70 trailer]
nil
```

- 即可看到，当前类 `YTPlaybackData` 的实例：`<0x282153e70>` 的各种属性值

## 用po打印Class类的属性Property或函数Method

- 打印Class的属性或函数值

```
po [ClassName classMethodOrProperty]
```

- 打印Class的Instance的属性或函数之

```
po [objc_getClass("ClassName") instanceMethodOrProperty]
```

举例：

```
(lldb) po [objc_getClass("TTMacroManager") _shortMethodDescription]
<TTMacroManager: 0x103b56e48>
in TTMacroManager
  Class Methods
    + (BOOL) isBDWEBIMAGE_APP_EXTENSIONS; (0x117cf1d50)
    + (BOOL) isDebug; (0x117cf1e28)
(NSSObject ...)
```

进一步的，对应着（导出抖音的）头文件：

```
#import <objc/NSObject.h>

@interface TTMacroManager : NSObject
{
```

```
+ (_Bool)isBDWEBIMAGE_APP_EXTENSIONS;
+ (_Bool)isDebug;

@end
```

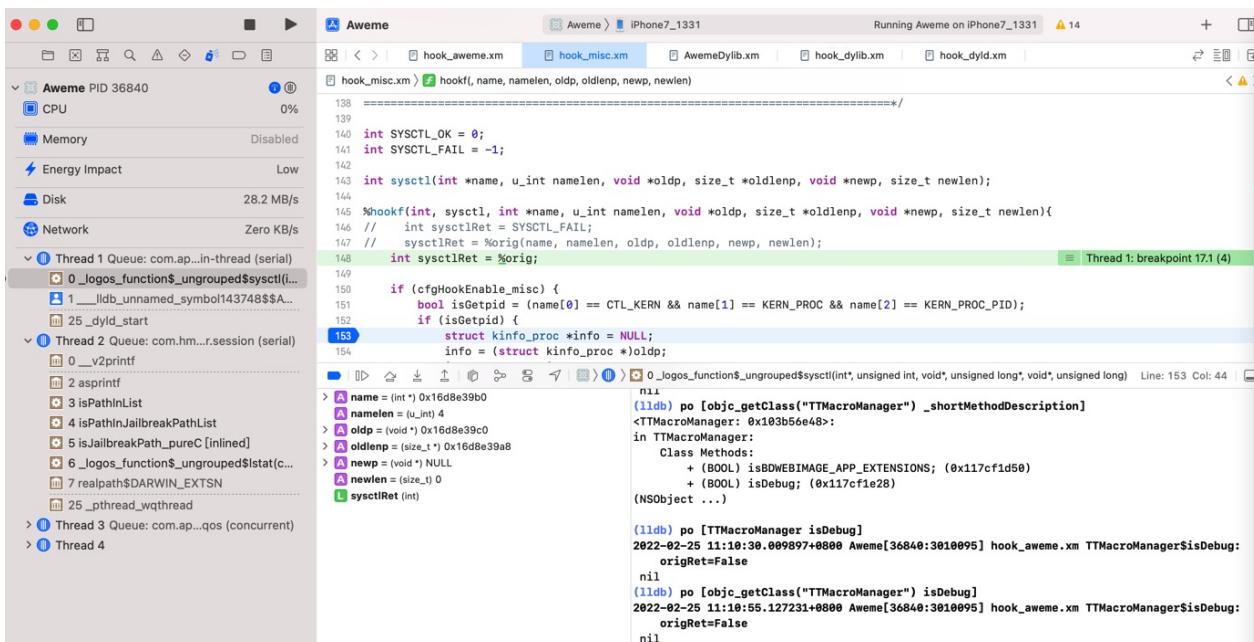
想要查看当前的Class的值，则是：

```
(lldb) po [TTMacroManager isDebug]
2022 02 25 11:10:30.009897 0800 Aweme[36840:3010095] hook_aweme.xm TTMacroManager$isDebug: origRet False
nil
```

注：此处输出的是被我加了hook了的代码的log

如果想要查看实例instance的值，则是：

```
(lldb) po [objc_getClass("TTMacroManager") isDebug]
2022 02 25 11:10:55.127231 0800 Aweme[36840:3010095] hook_aweme.xm TTMacroManager$isDebug: origRet False
nil
```



## po 失效时换用 object\_getClassName 查看是什么类

iOS逆向期间，正常的话，`po` 是可以打印出某个地址，具体是什么（ObjC的）类

比如：

```
(lldb) po 0x0000000137419800
AWEInviteSearchTableViewCell: 0x137419800; baseClass = UITableViewCell; frame = (0 152; 375 76); autoresize = W; layer = <CALayer: 0x28f694b00>
```

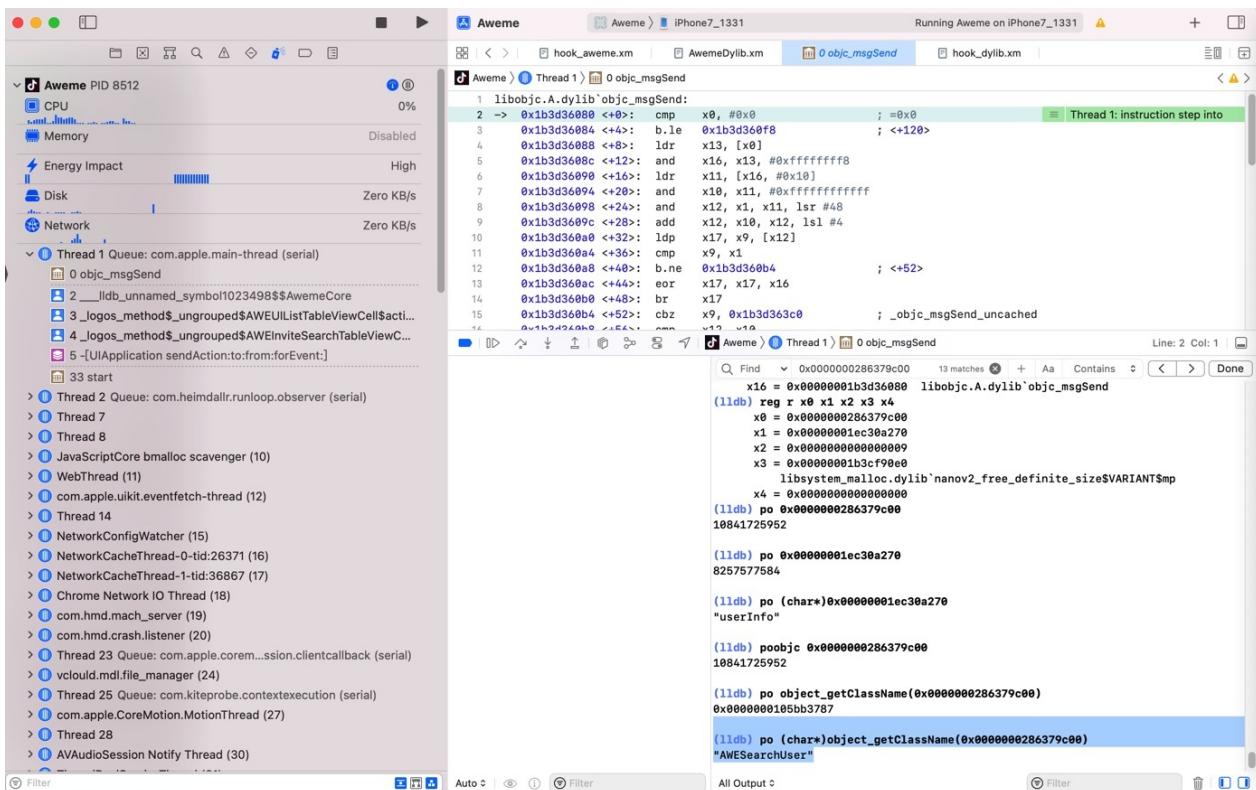
-> 从而通过调试搞懂代码的具体逻辑：调用了什么(ObjC的)类的什么函数。

而有时候，不知何故，`po` 失效，则打印不出来是什么类：

```
(lldb) po 0x00000000286379c00
10841725952
```

此时，可以换用：`object_getClassName`

```
(lldb) po (char*)object_getClassName(0x00000000286379c00)
"AWEsearchUser"
```



## po查看类的描述的同时可以看到父类的相关定义

比如抖音的：

```
Aweme_classDump/Aweme_17.8.0_header/Aweme/AWEPlayInteractionFollowSuccessElement.h
```

```
#import "AWEPlayInteractionBottomElement.h"

#import "AWEUserMessage-Protocol.h"

@class AWEAntiAddictedNoticeBarView, AWEHistoryPublicDataController, NSString;

@interface AWEPlayInteractionFollowSuccessElement : AWEPlayInteractionBottomElement <AW
EUserMessage>
{
    AWEAntiAddictedNoticeBarView *_antiAddictedNoticeBarView;
    AWEHistoryPublicDataController *_listDataController;
}
```

```

    long long _actionType;
}

- (void)cxx_destruct;
@property(nonatomic) long long actionType; // @synthesize actionType=_actionType;
@property(retain, nonatomic) AWEHistoryPublicDataController listDataController; // @synthesize listDataController=_listDataController;
@property(retain, nonatomic) AWEAntiAddictedNoticeBarView *antiAddictedNoticeBarView; // @synthesize antiAddictedNoticeBarView=_antiAddictedNoticeBarView;
- (void)didFinishUnFollowUser:(id)arg1 status:(long long)arg2 error:(id)arg3;
- (void)p_hideAntiAddictedNoticeBarView:(long long)arg1 animation:(CDUnknownBlockType)arg2;
- (void)p_showAntiAddictedNoticeBarViewWithCompletion:(CDUnknownBlockType)arg1;
- (void)noticeTapped;
- (void)showFollowSuccessNoticeBar:(id)arg1;
- (void)hideMutexTempElement:(CDUnknownBlockType)arg1;
- (void)dealloc;
- (void)viewDidDisposed;
- (void)reset;
- (void)viewDidLoad;
- (void)initializeElement;

// Remaining properties
@property(nonatomic, copy) NSString *debugDescription;
@property(nonatomic, copy) NSString *description;
@property(nonatomic) unsigned long long hash;
@property(nonatomic) Class superclass;

@end

```

去Xcode的lldb中动态调试抖音期间，通过：

```
po [objc_getClass("AWEPlayInteractionFollowSuccessElement") _shortMethodDescription]
```

- 不仅能看到：类 AWEPlayInteractionFollowSuccessElement 本身的信息
- 还能看到：父类 AWEPlayInteractionBottomElement
  - 父类的父类： AWEPlayInteractionBottomElement
    - 父类的父类的父类： AWEPlayInteractionBaseElement
      - 父类的父类的父类的父类： AWEBaseElement
        - 直到最后的根对象： NSObject

具体输出内容是：

```
(lldb) po [objc_getClass("AWEPlayInteractionFollowSuccessElement") _shortMethodDescription]
2022 04 02 13:36:53.128548 0800 Aweme[45939:3543378] hook_misc.xm NSBundle$bundlePath:
origBundlePath = /usr/lib
AWEPlayInteractionFollowSuccessElement 0x10629ab90
in AWEPlayInteractionFollowSuccessElement
Properties:
    @property (retain, nonatomic) AWEAntiAddictedNoticeBarView *antiAddictedNoticeBarView; // @synthesize antiAddictedNoticeBarView = _antiAddictedNoticeBarView;
    @property (retain, nonatomic) AWEHistoryPublicDataController listDataController;
```

```

;  (@synthesize listDataController = _listDataController;
     @property (nonatomic) long actionType;  (@synthesize actionType = _actionType;)
     @property (readonly) unsigned long hash;
     @property (readonly) Class superclass;
     @property (readonly, copy) NSString  description;
     @property (readonly, copy) NSString  debugDescription;
Instance Methods:
- (void) didFinishUnFollowUser:(id)arg1 status:(long)arg2 error:(id)arg3; (0x11
58b84b4)
- (void) viewDidDisposed; (0x1158b6a8c)
- (void) initializeElement; (0x1158b672c)
- (id) listDataController; (0x1158b86f0)
- (void) setListDataController:(id)arg1; (0x1158b8758)
- (void) noticeTapped; (0x1158b731c)
- (void) setAntiAddictedNoticeBarView:(id)arg1; (0x1158b8748)
- (id) antiAddictedNoticeBarView; (0x1158b859c)
- (void) hideMutexTempElement:(block)arg1; (0x1158b6b4c)
- (void) p_hideAntiAddictedNoticeBarView:(long)arg1 animation:(^block)arg2; (0x
1158b7f74)
- (void) p_showAntiAddictedNoticeBarViewWithCompletion:(^block)arg1; (0x1158b78
a4)
- (void) showFollowSuccessNoticeBar:(id)arg1; (0x1158b6e7c)
- (void) dealloc; (0x1158b6ad8)
- (void) cxx_destruct; (0x1158b8788)
- (void) reset; (0x1158b69fc)
- (void) viewDidLoad; (0x1158b67b4)
- (long) actionType; (0x1158b8768)
- (void) setActionType:(long)arg1; (0x1158b8778)
in AWEPlayInteractionBottomElement
Instance Methods:
- (void) configWithParamDict:(id)arg1; (0x10b014ce0)
- (id) bottomElementContainer; (0x10b014df8)
- (BOOL) elementAppearLowPriorityNeedAvoid; (0x115851c34)
- (void) updateNextElementAppearStatus; (0x1158519a0)
- (void) reset; (0x115851b1c)
in AWEPlayInteractionBaseElement
Properties:
@property (retain, nonatomic) AWEAwemeModel* model;  (@synthesize model = _model
;)
@property (nonatomic) unsigned long playerStatus;  (@synthesize playerStatus =
_playerStatus;)
@property (weak, nonatomic) NSPointerArray allElements;  (@synthesize allEleme
nts = _allElements;)
@property (readonly) unsigned long hash;
@property (readonly) Class superclass;
@property (readonly, copy) NSString  description;
@property (readonly, copy) NSString  debugDescription;
Instance Methods:
- (struct CGRect) viewFrame; (0x11584af68)
- (void) videoDidActivity; (0x10b0937f8)
- (BOOL) alertIfNotValidForAction:(long)arg1; (0x11584aae8)
- (id) elementFromAll:(id)arg1; (0x10b043ad8)
- (void) viewController_viewWillDisappear; (0x11584abc4)
- (void) viewController_viewDidDisappear; (0x11584abc8)
- (void) viewController_didEndDisplaying; (0x11584abcc)
- (void) viewController_willDisplay; (0x11584abb4)

```

```

- (void) viewControllerWillAppear; (0x11584abbc)
- (void) viewControllerDidAppear; (0x11584abc0)
- (void) hideAllElementExcepts (id)arg1; (0x11584a90c)
- (void) updateAllElement; (0x11584ab74)
- (void) setAllElements (id)arg1; (0x10b019980)
- (id) currentInfoForUnitWithIdentifier:(id)arg1; (0x11584aed8)
- (void) hideProgressSliderPopView; (0x11584ac74)
- (id) currentInfoForSubUnits; (0x11584adcc)
- (id) currentInfoForUnitWithClassName:(id)arg1; (0x11584ae5c)
- (void) dealloc; (0x11584abe4)
- (void) cxx_destruct; (0x11584b090)
- (void) pause; (0x11584ac54)
- (void) resume; (0x11584ac64)
- (void) setData:(id)arg1; (0x10b04aecc)
- (id) context; (0x10b00ef84)
- (void) reset; (0x11584abd0)
- (id) model; (0x10b016d44)
- (void) setModel:(id)arg1; (0x10b01cf58)
- (void) play; (0x11584abe0)
- (void) prepareForDisplay; (0x10b06b068)
- (BOOL) isShowing; (0x11584b074)
- (void) didEndDisplaying; (0x11584abb8)
- (id) currentInfo; (0x11584acc0)
- (unsigned long) playerStatus; (0x11584b080)
- (void) setHide:(BOOL)arg1; (0x10b0899c0)
- (void) setPlayerStatus:(unsigned long)arg1; (0x10b0431dc)
- (id) allElements; (0x10b043c60)

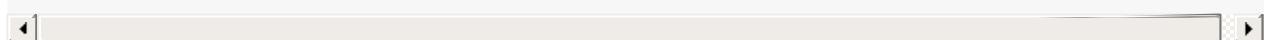
in AWBaseElement:
Properties:
@property (weak, nonatomic) AWElementContainer *elementContainer; (@dynamic elementContainer;)
@property (weak, nonatomic) UIView *boxView; (@synthesize boxView = _boxView;)
@property (weak, nonatomic) UIView *elementView; (@synthesize elementView = _elementView;)
@property (nonatomic) BOOL hascreateView; (@synthesize hascreateView = _hascreateView;)
@property (retain, nonatomic) AWPageContext *context; (@synthesize context = _context;)
@property (weak, nonatomic) AWElementContainer *elementContainer; (@synthesize elementContainer = _elementContainer;)
@property (retain, nonatomic) UIView *view; (@synthesize view = _view;)
@property (retain, nonatomic) id data; (@synthesize data = _data;)
@property (readonly, nonatomic, getter=isViewLoaded) BOOL viewLoaded;
@property (copy, nonatomic) NSString *identity; (@synthesize identity = _identity;)
@property (nonatomic) BOOL appear; (@synthesize appear = _appear;)
@property (readonly, weak, nonatomic) UIViewController *viewController; (@synthesize viewController = _viewController;)
@property (readonly) unsigned long hash;
@property (readonly) Class superclass;
@property (readonly, copy) NSString *description;
@property (readonly, copy) NSString *debugDescription;

Instance Methods:
- (void) configWithParamDict:(id)arg1; (0x11232b734)
- (void) viewDidDisposed; (0x11232b730)
- (void) processAppear:(BOOL)arg1; (0x10b010138)

```

```
- (id) elementContainer; (0x10b01036c)
- (void) setAppear (BOOL)arg1; (0x10b01d350)
- (void) initializeElement; (0x10b00f048)
- (void) setElementContainer:(id)arg1; (0x10b00ed6c)
- (id) boxView; (0x11232b8fc)
- (void) setBoxView (id arg1); (0x11232b914)
- (id) elementView; (0x11232b920)
- (void) setElementView (id arg1); (0x11232b938)
- (BOOL) hasCreateView; (0x11232b944)
- (void) setHascreateView (BOOL)arg1; (0x11232b94c)
- (void) addSubviewWithLayout (id)arg1 withEdgeInsets (struct UIEdgeInsets)arg2;
(0x11232b9ec)
- (void) addSubviewWithLayout (id)arg1 withEdgeInsets (struct UIEdgeInsets)arg2
withHeight:(double)arg3; (0x11232b9fc)
- (void) hide (BOOL)arg1 duration (double)arg2 animations:(^block)arg3; (0x1123
2bcc4)
- (void) hide (BOOL)arg1 duration (double)arg2 withTransform (struct CGAffineTransform)
arg3 animations:(^block)arg4; (0x11232bd48)
- (void) addSubviewWithLayout (id)arg1; (0x11232b9d0)
- (void) hide (BOOL)arg1 duration (double)arg2; (0x11232bcb4)
- (void) .cxx_destruct; (0x11232b968)
- (id) data; (0x11232b8e8)
- (void) setData (id)arg1; (0x11232b6cc)
- (id) context; (0x11232b954)
- (id) identity; (0x11232b8f0)
- (void) setContext (id)arg1; (0x11232b95c)
- (void) setIdentity (id)arg1; (0x10b014df0)
- (id) view; (0x10b0101fc)
- (void) setView (id)arg1; (0x11232b8dc)
- (void) loadView; (0x10b0102e0)
- (void) viewDidLoad; (0x11232b72c)
- (BOOL) isViewLoaded; (0x10b0101ec)
- (id) viewController; (0x10b012c08)
- (BOOL) appear; (0x10b00f010)

(NSObject ...)
```



Aweme > Thread 161 > 0 -[NSString stringByAppendingString:] Line: 2 Col: 1

```
(lldb) po [objc_getClass("AWEPlayInteractionFollowSuccessElement")
shortMethodDescription]
2022-04-02 13:36:53.128548+0800 Aweme[45939:3543378] hook_misc.xm NSBundle$bundlePath:
origBundlePath=/usr/lib
<AWEPlayInteractionFollowSuccessElement: 0x10629ab90>:
in AWEPlayInteractionFollowSuccessElement:
Properties:
@property (retain, nonatomic) AWEAntiAddictedNoticeBarView*
antiAddictedNoticeBarView; (@synthesize antiAddictedNoticeBarView =
_antiAddictedNoticeBarView;
@property (retain, nonatomic) AWEHistoryPublicDataController* listDataController;
(@synthesize listDataController = _listDataController;
@property (nonatomic) long actionType; (@synthesize actionType = _actionType;
@property (readonly) unsigned long hash;
@property (readonly) Class superclass;
@property (readonly, copy) NSString* description;
@property (readonly, copy) NSString* debugDescription;
Instance Methods:
- (void) didFinishUnFollowUser:(id)arg1 status:(long)arg2 error:(id)arg3;
(0x1158b84b4)
- (void) viewDidDisposed; (0x1158b6a8c)
- (void) initializeElement; (0x1158b672c)
- (id) listDataController; (0x1158b86f0)
- (void) setListDataController:(id)arg1; (0x1158b8758)
- (void) noticeTapped; (0x1158b731c)
- (void) setAntiAddictedNoticeBarView:(id)arg1; (0x1158b8748)
- (id) antiAddictedNoticeBarView; (0x1158b859c)
- (void) hideMutexTempElement:(^block)arg1; (0x1158b6b4c)
- (void) p_hideAntiAddictedNoticeBarView:(long)arg1 animation:(^block)arg2;
(0x1158b7f74)
- (void) p_showAntiAddictedNoticeBarViewWithCompletion:(^block)arg1; (0x1158b78a4)
- (void) showFollowSuccessNoticeBar:(id)arg1; (0x1158b6e7c)
- (void) dealloc; (0x1158b6ad8)
- (void) .cxx_destruct; (0x1158b8788)
- (void) reset; (0x1158b69fc)
- (void) viewDidLoad; (0x1158b67b4)
```

All Output

Filter



```

Aweme > Thread 161 > 0 -[NSString stringByAppendingString:]
Line: 2 Col: 1 | □
  instance methods:
    - (void) didFinishUnFollowUser:(id)arg1 status:(long)arg2 error:(id)arg3;
      (0x1158b84b4)
    - (void) viewDidDisposed; (0x1158b6a8c)
    - (void) initializeElement; (0x1158b672c)
    - (id) listDataController; (0x1158b86f0)
    - (void) setListDataController:(id)arg1; (0x1158b8758)
    - (void) noticeTapped; (0x1158b731c)
    - (void) setAntiAddictedNoticeBarView:(id)arg1; (0x1158b8748)
    - (id) antiAddictedNoticeBarView; (0x1158b859c)
    - (void) hideMutexTempElement:(^block)arg1; (0x1158b6b4c)
    - (void) p_hideAntiAddictedNoticeBarView:(long)arg1 animation:(^block)arg2;
      (0x1158b7f74)
    - (void) p_showAntiAddictedNoticeBarViewWithCompletion:(^block)arg1; (0x1158b78a4)
    - (void) showFollowSuccessNoticeBar:(id)arg1; (0x1158b6e7c)
    - (void) dealloc; (0x1158b6ad8)
    - (void) .cxx_destruct; (0x1158b8788)
    - (void) reset; (0x1158b69fc)
    - (void) viewDidLoad; (0x1158b67b4)
    - (long) actionType; (0x1158b8768)
    - (void) setActionType:(long)arg1; (0x1158b8778)
in AWEPlayInteractionBottomElement:
  Instance Methods:
    - (void) configWithParamDict:(id)arg1; (0x10b014ce0)
    - (id) bottomElementContainer; (0x10b014df8)
    - (BOOL) elementAppearLowPriorityNeedAvoid; (0x115851c34)
    - (void) updateNextElementAppearStatus; (0x1158519a0)
    - (void) reset; (0x115851b1c)
in AWEPlayInteractionBaseElement:
  Properties:
    @property (retain, nonatomic) AWEAwemeModel* model; (@synthesize model = _model;)
    @property (nonatomic) unsigned long playerStatus; (@synthesize playerStatus =
      _playerStatus;)
    @property (weak, nonatomic) NSPointerArray* allElements; (@synthesize
      allElements = _allElements;)
All Output ◁ Filter □ | □ | □

```

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:

2022-10-27 11:54:30

## 子教程

iOS逆向中动态调试相关的子教程：

- 动态调试
  - 工具
    - MonkeyDev
      - [iOS逆向开发：MonkeyDev调试](#)
    - Frida
      - [逆向调试利器：Frida](#)
    - LLDB
      - [主流调试器：LLDB](#)
    - IDA
      - [逆向利器：IDA](#)
    - Xcode
      - [Xcode开发：调试心得](#)
  - 子领域
    - 断点
      - [iOS逆向之动态调试：断点](#)
- [iOS逆向开发：iOS底层机制](#)
  - Block
    - [iOS逆向开发：Block匿名函数](#)
  - Runtime
    - [iOS逆向开发：ObjC运行时](#)

另外，还有相关代码：

- [iOSYouTubeAdsFilter](#)
  - [crifan/iOSYouTubeAdsFilter: MonkeyDev+Xcode项目，iOS逆向YouTube，尝试实现广告过滤功能](#)

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：

2023-07-13 21:59:30

## 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：  
2022-03-17 20:39:28

## 参考资料

- 【已解决】 Cycript中找点击按钮的响应函数处理函数
- 【已解决】 iPhone8中Cycript注入设置Preferences卡死
- 【已解决】 cycript中如何打印窗口视图的详情包括字符串
- 【已解决】 Cycript显示的页面元素不是最新的当前弹框页面
- 【已解决】 iOS的ObjC中如何获取到最顶层的窗口视图
- 【已解决】 iOS逆向： palera1n越狱后iPhone中初始化安装Cycript环境
- 【已解决】 iOS逆向Apple账号： 搭建Cycript调试系统设置UI界面
- 【已解决】 Mac中下载和安装Reveal
- 【已解决】 MonkeyDev中如何使用Reveal调试YouTube广告页面元素
- 【已解决】 Mac中Reveal中看不到MonkeyDev调试的iPhone设备
- 【未解决】 iOS逆向AppleStore： 用debugserver+lldb调试ipa
- 【已解决】 debugserver调试iOS抖音报错： Failed to get connection from a remote gdb process
- 【未解决】 iOS中debugserver调试报错： Failed to get connection from a remote gdb process
- 【已解决】 debugserver带日志运行报错： Failed to open log file for writing errno 1 Operation not permitted
- 【已解决】 用debugserver启动iPhone中抖音app
- 【已解决】 Mac中如何用lldb调试iPhone中的app
- 【未解决】 iOS中debugserver启动报错： failed to launch process debugserver Operation not permitted
- 【记录】 XCode+MonkeyDev动态调试YouTube类： YTWatchMiniBarViewController
- 【未解决】 iOS逆向iOS15的debugserver： 找其他版本debugserver看是否可用
- 【未解决】 iOS逆向iOS15的debugserver： Mac端找到的debugserver
- 【记录】 iOS越狱： XinaA15的rootless越狱后iPhone中的debugserver
- 【未解决】 iOS逆向iOS15的debugserver： 运行报错Killed 9
- 【记录】 iOS越狱： 如何写回改动后的debugserver到rootless越狱iPhone中
- 【已解决】 把增加了权限的debugserver拷贝到越狱iPhone中
- 【已解决】 debugserver启动iOS的app抖音报错： Segmentation fault 11
- 【已解决】 用debugserver和lldb去调试iOS的app
- 【记录】 iOS逆向WhatsApp： lldb+debugserver调试时加载的image镜像列表
- 【已解决】 Mac中lldb调试iOS的app抖音报错： Process exited with status 45
- 【记录】 分析iOS抖音入口函数\_awemeMain运行逻辑
- 【已解决】 抖音反反调试： 把二进制AwemeCore的svc 0x80指令替换成nop指令
- 【已解决】 Mac中用IDA实现抖音二进制AwemeCore的svc 0x80替换成nop指令
- 
- iOS逆向调试： debugserver+lldb
- 
- iOS逆向攻防实战 - 掘金 (juejin.cn)
- SpringBoard tweak 双击图标启动debugserver - 干货分享 - 睿论坛
- Frida & Passionfruit 安装记录. Frida是一个功能强大且可扩展的工具包，具有众多优势，非常适合测试和评估And... | by iOS Jailbreak Notes | Medium
- Reveal 24 (12917) 破解版 for Mac iOS界面UI开发调试神器 (macwk.com) 下载到
- Reveal24(12917)\_macwk.com.dmg\_\_macwk.com.dmg)

- Cycript Tricks - iPhone Development Wiki
- iOS攻防（六）：使用Cycript一窥运行程序的神秘面纱(入门篇) | 曹雪松de博客|CoderBoy's Blog (sevencho.github.io)
- iOS逆向之Cycript的使用 - 简书 (jianshu.com)
- Cycript的一些基础使用 - 简书 (jianshu.com)
- CoderMJLee/mjcript: 【越狱-逆向】基于Cycript实现的一些实用函数 (github.com)
- mjcript/mjcript.cy at master · CoderMJLee/mjcript (github.com)
- debugserver - iPhone Development Wiki
- c - lldb finding exit point of app - Stack Overflow
- errno.h (apple.com)
- Online ARM to HEX Converter (armconverter.com)
- 

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：  
2023-10-21 19:48:56