符号表工具iOS版-使用指南

- 符号表工具iOS版-使用指南
  - 1. 介绍
    - 1.1 环境要求
    - 1.2 符号表提取要求
    - 1.3 配置文件
    - 1.4 上传功能
    - 1.5 如何获取App ID和App Key
  - 2. 提取符号表文件的方法
    - 2.1 工具使用方法
    - 2.2 工具选项
    - 2.3 例子
      - 2.3.1 dSYM的符号表生成和上传
      - 2.3.2 直接上传dSYM文件
  - 3. dSYM文件
    - 3.1 什么是dSYM文件?
    - 3.2 如何定位dSYM文件?
    - 3.3 XCode编译后没有生成dSYM文件?
    - 3.4 如何判断dSYM文件是否与Crash的UUID匹配?
    - 3.5 如何查看dSYM文件的UUID?
      - 3.5.1 通过命令查看UUID
      - 3.5.2 通过符号表工具查看UUID
    - 3.6 如何找回已发布到App Store的App对应的dSYM文件?
      - 3.6.1 通过Xcode的Archive找回
      - 3.6.2 通过mdfind工具找回
    - 3.7 找不到Crash对应的dSYM文件?

---

1. 介绍

为了能快速并准确地定位用户App发生**Crash的代码位置**，Bugly使用**符号表文件**对App发生Crash的程序**堆栈**进行**解析**和**还原**。

举一个实例：

还原前堆栈	还原后堆栈
0 Test 0x00b122e4 0x00004000 + 11592420	0 Test 0x00b122e4 -[FXLabel initWithFrame:] (FXLabel.m:71)
1 Test 0x0062f37c 0x00004000 + 6468476	1 Test 0x0062f37c -[BTBannerView resetUI] (BTBannerView.m:312)
2 Test 0x0062fe40 0x00004000 + 6471232	2 Test 0x0062fe40 -[BTNavView init] (BTNavView.m:76)

而符号表工具，正是Bugly提供给开发者提取符号表文件（.symbol）的工具。另外，Bugly提供了自动上传符号表文件的方法，请参考《Bugly iOS 符号表配置》，建议使用自动上传的方式。

`Bugly iOS符号表工具2.5.0及以上版本支持使用脚本直接上传dSYM文件，上传方法请参考“2.3.2 直接上传dSYM文件”。`

1.1 环境要求

符号表工具的运行需要Java运行环境（JRE或JDK版本需要>=1.6）。

1.2 符号表提取要求

提取符号表需要符号表工具和**dSYM文件**（具有调试信息的目标文件，可参考下文的第三部分：“**3. dSYM文件**”）。

1.3 配置文件

Bugly iOS符号表工具2.3.0及以上版本增加了配置文件的解析功能，工具包中提供了一个与工具JAR同目录的默认配置文件（settings.txt）。

可以通过配置文件设置以下信息：

- Debug：调试模式开关（打印更多Log）
- Upload：上传开关
- ID：Bugly平台的App ID
- Key：Bugly平台的App key

1.4 上传功能

Bugly iOS符号表工具支持上传功能，使用上传功能时，必须要指定以下信息：

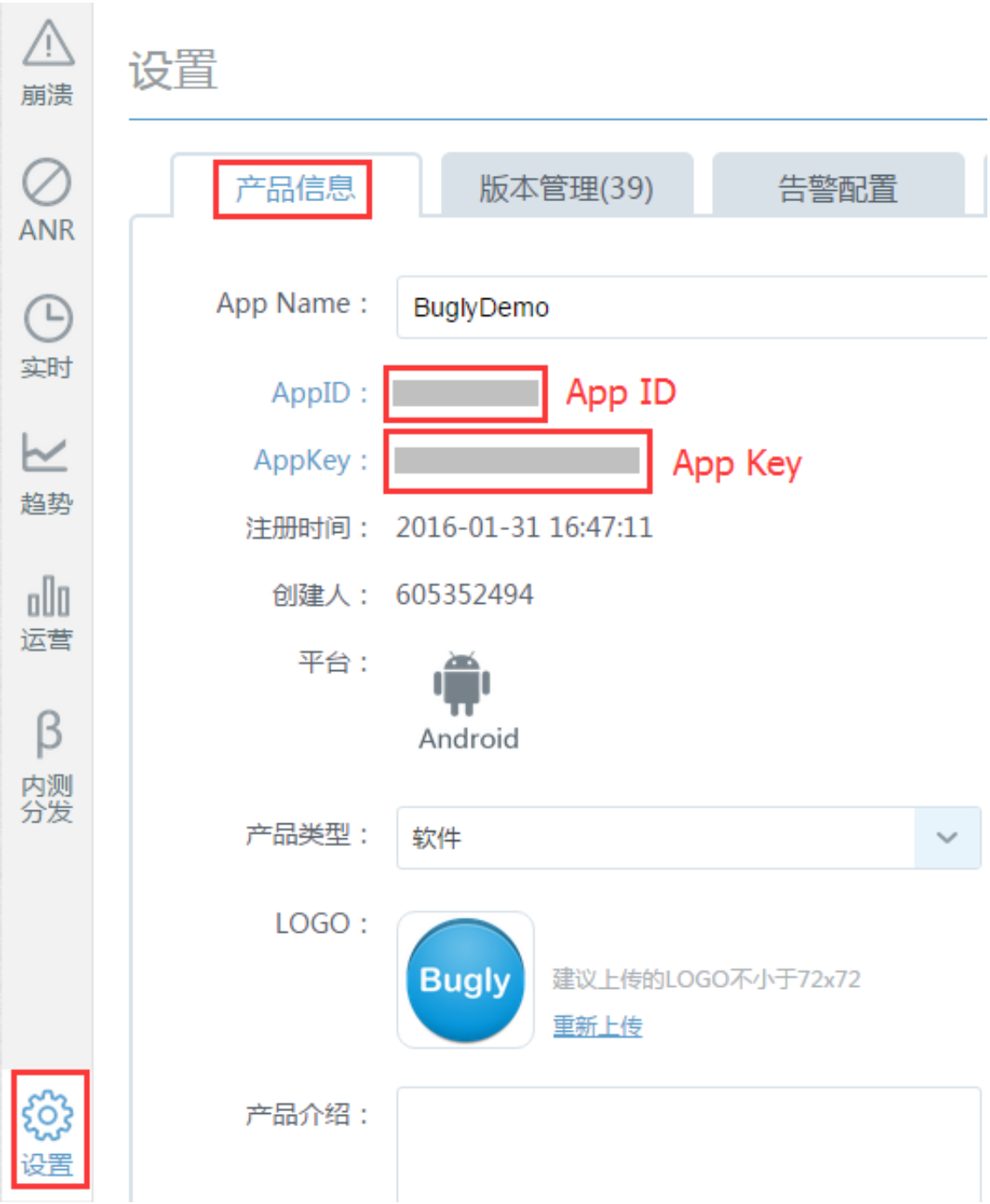
- App ID（可通过配置文件指定）
- App key（可通过配置文件指定）

- App版本
- App包名

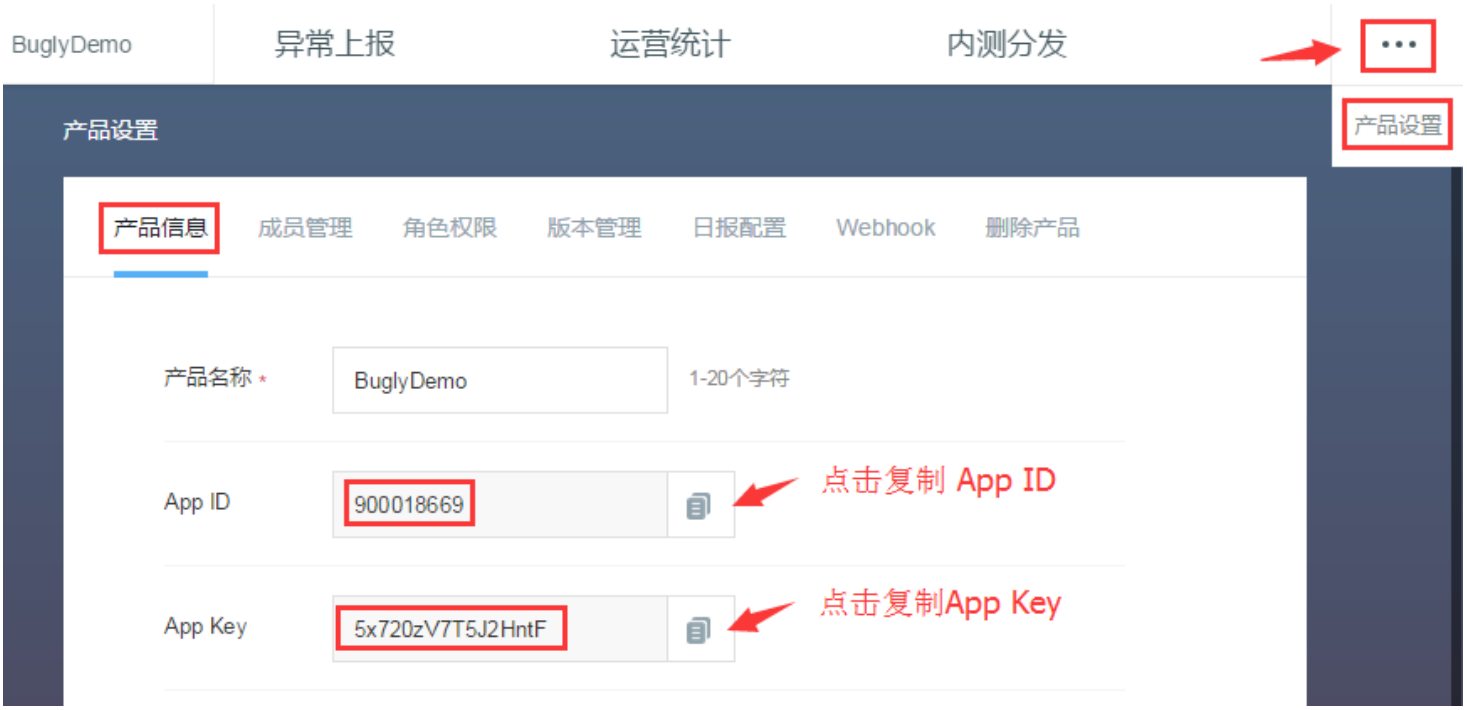
目前脚本不支持以上信息的指定，因此需要通过直接执行JAR包来使用上传功能。

1.5 如何获取App ID和App Key

- **Bugly 1.0**



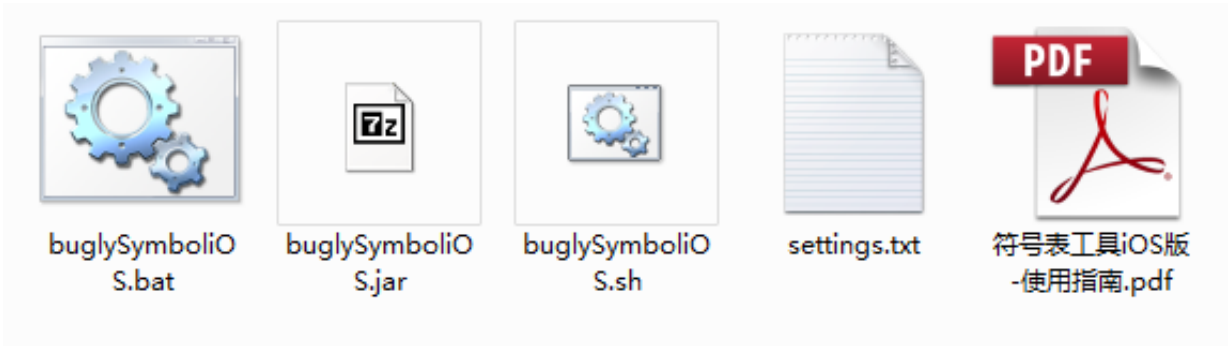
- **Bugly 2.0**



2. 提取符号表文件的方法

iOS版符号表工具支持Windows、Linux和Mac三个平台，同时提供了JAR包、各平台脚本和符号表配置文件：

- 符号表工具JAR包（buglySymboliOS.jar）
- Bat脚本（buglySymboliOS.bat）
- Shell脚本（buglySymboliOS.sh）
- 默认符号表配置文件（settings.txt）



使用脚本时，请保证脚本和jar包在同个目录下!

2.1 工具使用方法

```
java -jar <JAR包> <选项>

<Bat脚本> <选项>
```

```
4
5 <Shell脚本> <选项>
```

2.2 工具选项

选项	说明
-i	指定文件路径，可指定目录（必选）
-o	输出的符号表zip文件的路径，必须是zip文件
-d	调试模式开关（默认关闭）
-s	指定配置文件（默认读取JAR目录下的“settings.txt”文件）
-u	上传开关
-id	App ID
-key	App key
-package	App包名
-version	App版本
-symbol	生成Symbol文件

> 注意该版本的符号表工具默认不会生成Symbol文件，如果需要请指定“-symbol”参数运行符号表工具。

2.3 例子

注意事项

不要直接复制例子中的命令运行，需要根据自己的具体情况更改下命令。

环境和用户信息

- 系统：**Mac OS**
- 用户目录：**/home/batman**
- 符号表工具（已解压）所在目录：**/Users/batman/Downloads/buglySymboliOS**
- dSYM所在目录：**/Users/batman/Desktop/test.app.dSYM**
- APP ID：**900012345**

- APP key：**abcdefghijk**
- APP包名：**com.batman.demo**
- APP版本：**2.3.1**

2.3.1 dSYM的符号表生成和上传

生成符号表文件

使用符号表工具的JAR包生成符号表文件的命令如下：

```
cd /Users/batman/Downloads/buglySymboliOS

java -jar buglySymboliOS.jar -i /Users/batman/Desktop/test.app.dSYM
```

生成的符号表文位于：/Users/batman/Desktop/

生成符号表文件并自动上传

使用符号表工具的JAR包生成符号表文件，并自动上传的命令如下：

```
cd /Users/batman/Downloads/buglySymboliOS

java -jar buglySymboliOS.jar -i /Users/batman/Desktop/test.app.dSYM -u -i
d 900012345 -key abcdefghijk -package com.batman.demo -version 2.3.1
```

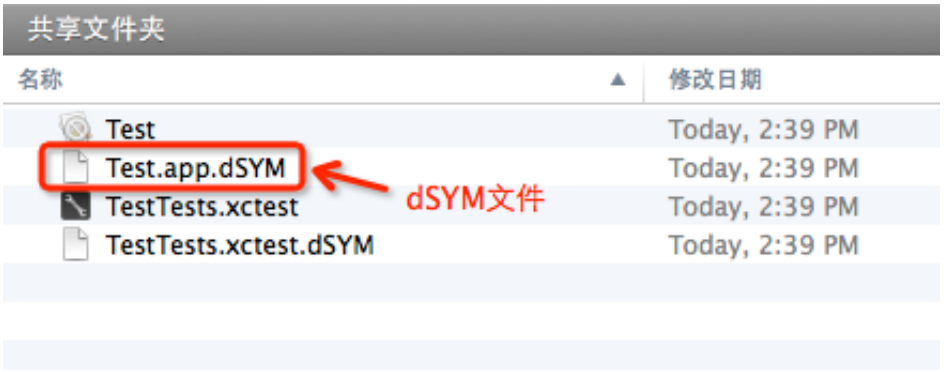
2.3.2 直接上传dSYM文件

注意，Bugly已不再支持直接上传dSYM文件。

3. dSYM文件

3.1 什么是dSYM文件?

iOS平台中，dSYM文件是指具有调试信息的目标文件，文件名通常为：**xxx.app.dSYM**。如下图所示：



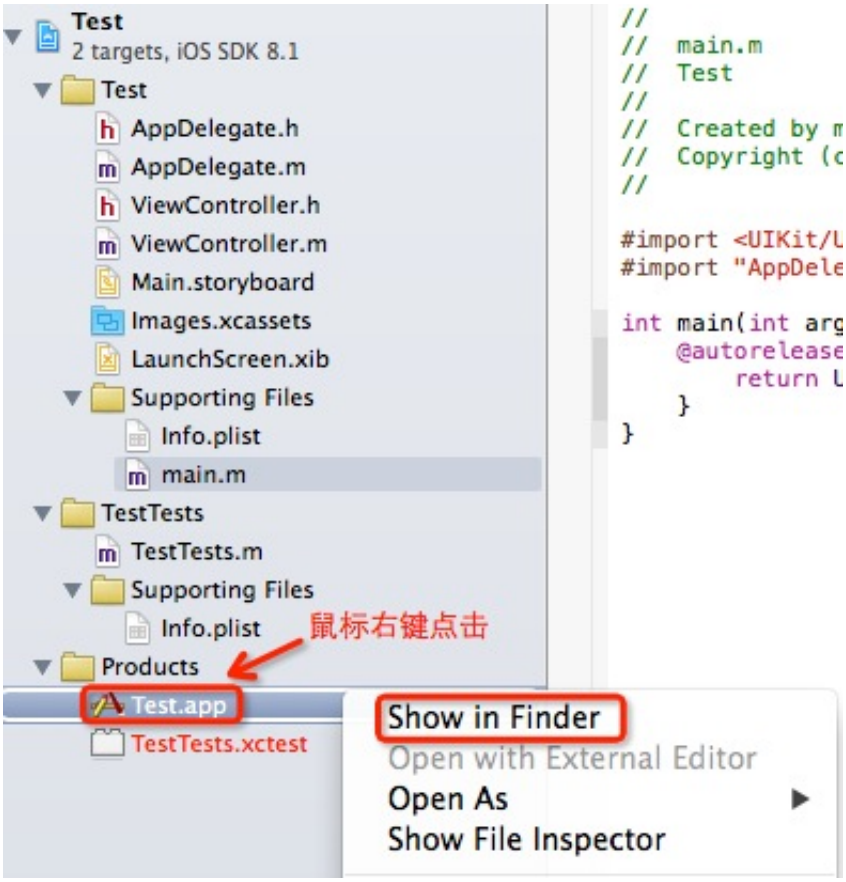
为了方便找回Crash对应的dSYM文件和还原堆栈，建议每次构建或者发布APP版本的时候，备份好dSYM文件。

3.2 如何定位dSYM文件?

一般情况下，项目编译完dSYM文件跟app文件在同一个目录下，下面以XCode作为IDE详细说明定位dSYM文件。

-> 进入XCode;
-> 打开工程（已编译过）;
-> 在左栏找到“Product”项;
-> 鼠标右键点击编译生成的“xxx.app”;
-> 点击“Show in Finder”;

如下图所示:



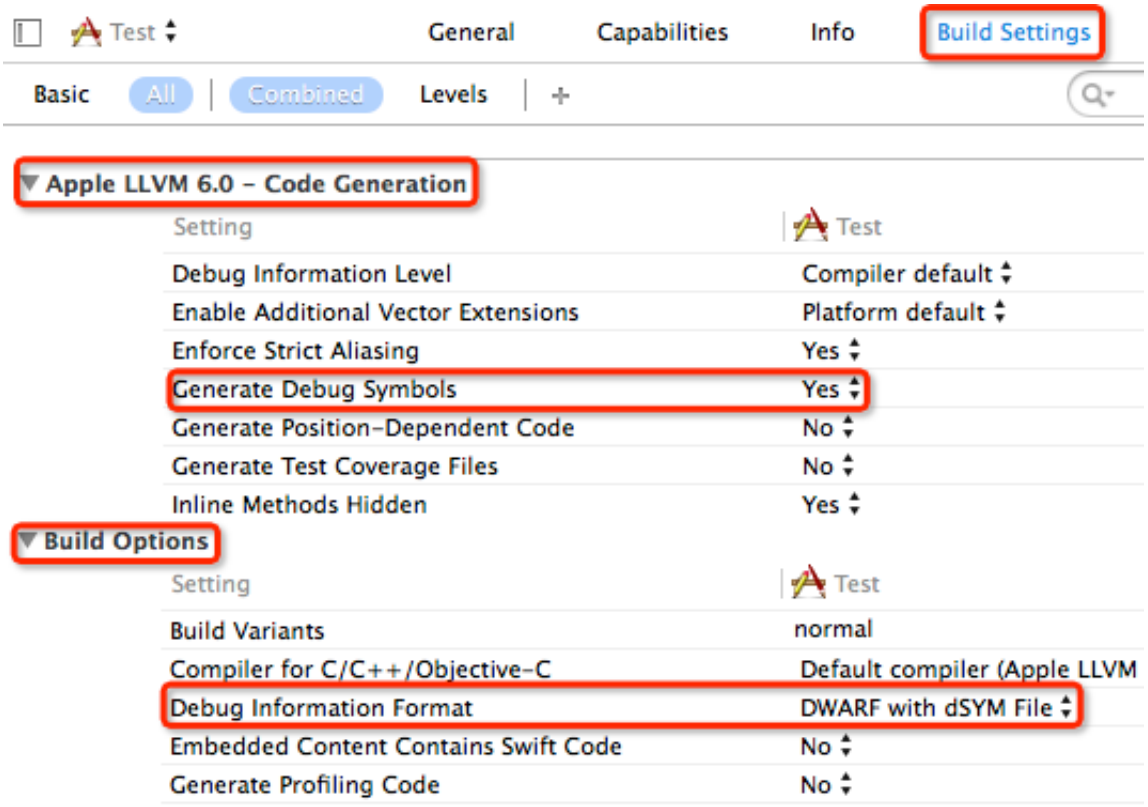
如果有多个dSYM文件，可以在使用工具时指定输入为dSYM文件所在的目录或者工程目录。

3.3 XCode编译后没有生成dSYM文件?

XCode Release编译默认会生成dSYM文件，而Debug编译默认不会生成，对应的Xcode配置如下：

```
XCode -> Build Settings -> Code Generation -> Generate Debug Symbols -> Yes
```

```
XCode -> Build Settings -> Build Option -> Debug Information Format -> DWARF with dSYM File
```



3.4 如何判断dSYM文件是否与Crash的UUID匹配?

Bugly还原Crash堆栈时，需要根据UUID来匹配符号表文件，因此只有上传的符号表文件的UUID与Crash对应APP的UUID一致时，才能准确地对堆栈进行还原。

- 查看符号表文件的UUID（“3.5 如何查看dSYM文件的UUID? ”）
- 查看Crash对应的APP的UUID

Bugly v1.0页面

崩溃 ---> Crash issue ---> dSYM UUID

Bugly v2.0页面

崩溃分析 ---> Crash issue ---> 符号表 ---> UUID

3.5 如何查看dSYM文件的UUID?

3.5.1 通过命令查看UUID

```
xcrun dwarfdump --uuid <dSYM文件>
```

3.5.2 通过符号表工具查看UUID

```
java -jar buglySymboliOS.jar -uuid -i <dSYM文件>
```

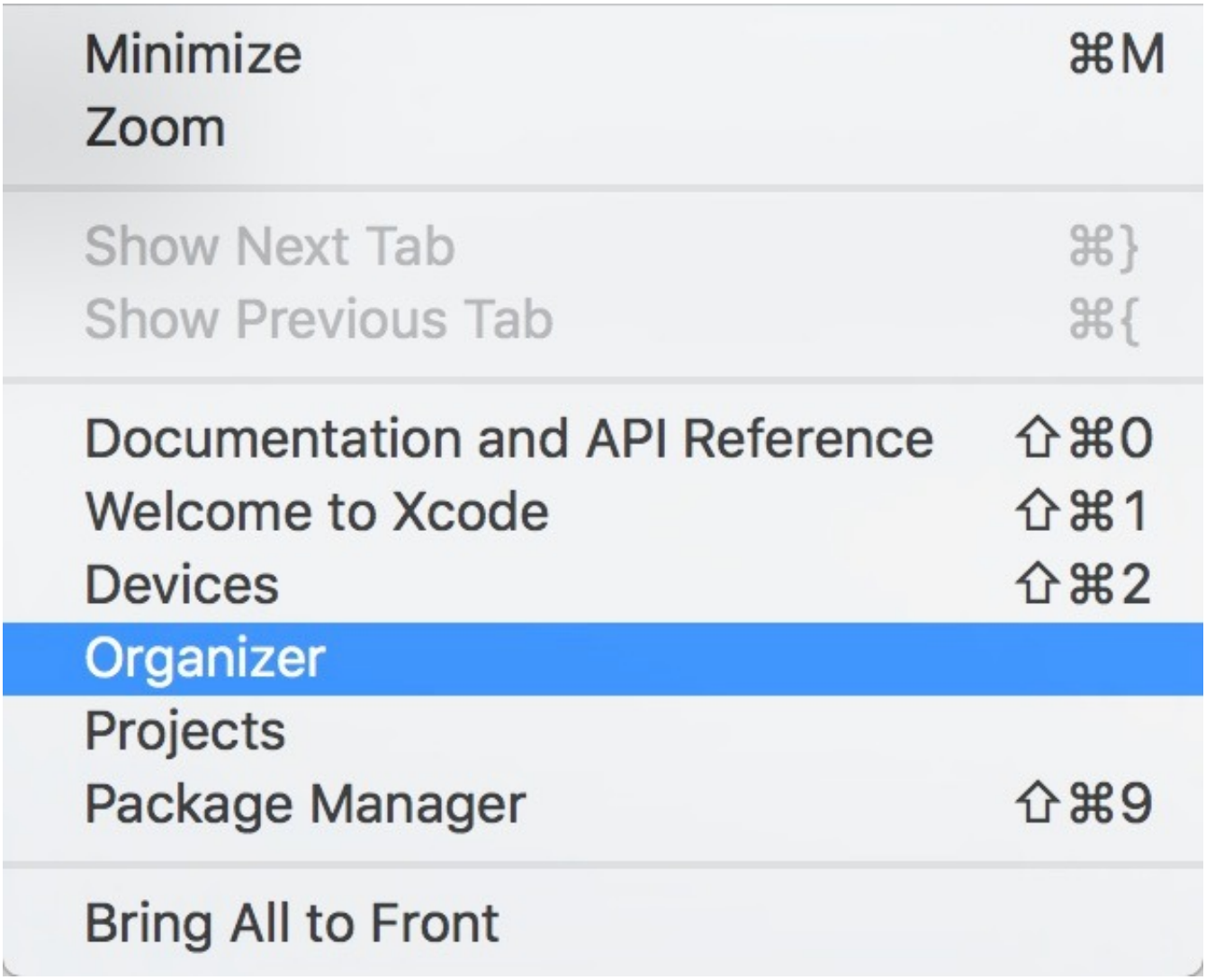
例子：

```
➜ Downloads java -jar buglySymboliOS.jar -uuid -i test.app.dSYM
/Users/jalen/Downloads/test.app.dSYM/Contents/Resources/DWARF/MD5Digest [armv7] cfd79653d5f133de864a015fb32fea7d
/Users/jalen/Downloads/test.app.dSYM/Contents/Resources/DWARF/MD5Digest [arm64] 3018e59f3d8836f08eee5f3aa02285ea
```

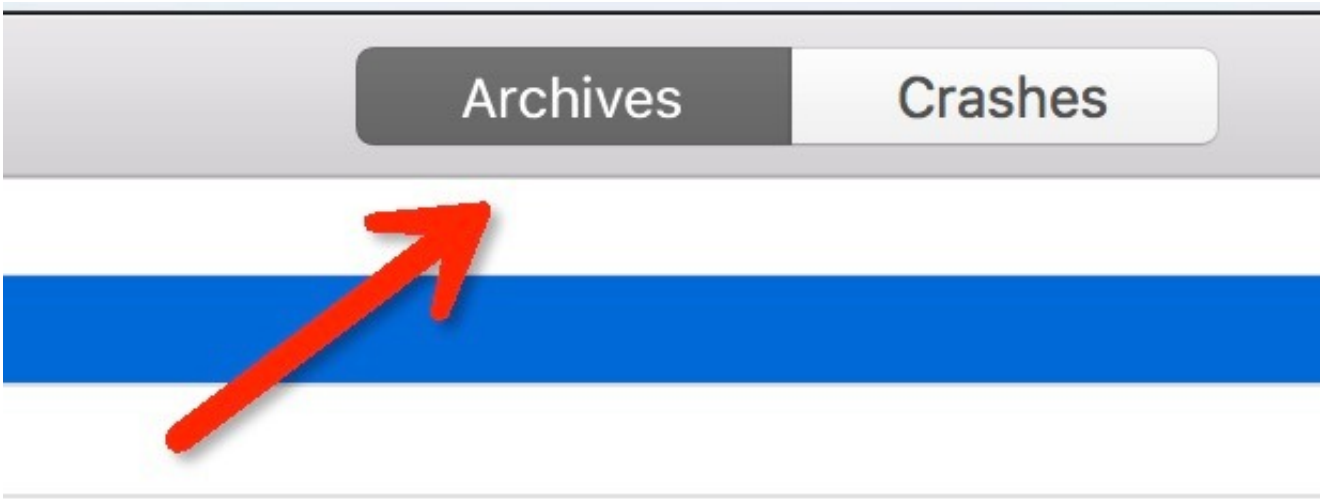
3.6 如何找回已发布到App Store的App对应的dSYM文件?

3.6.1 通过Xcode的Archive找回

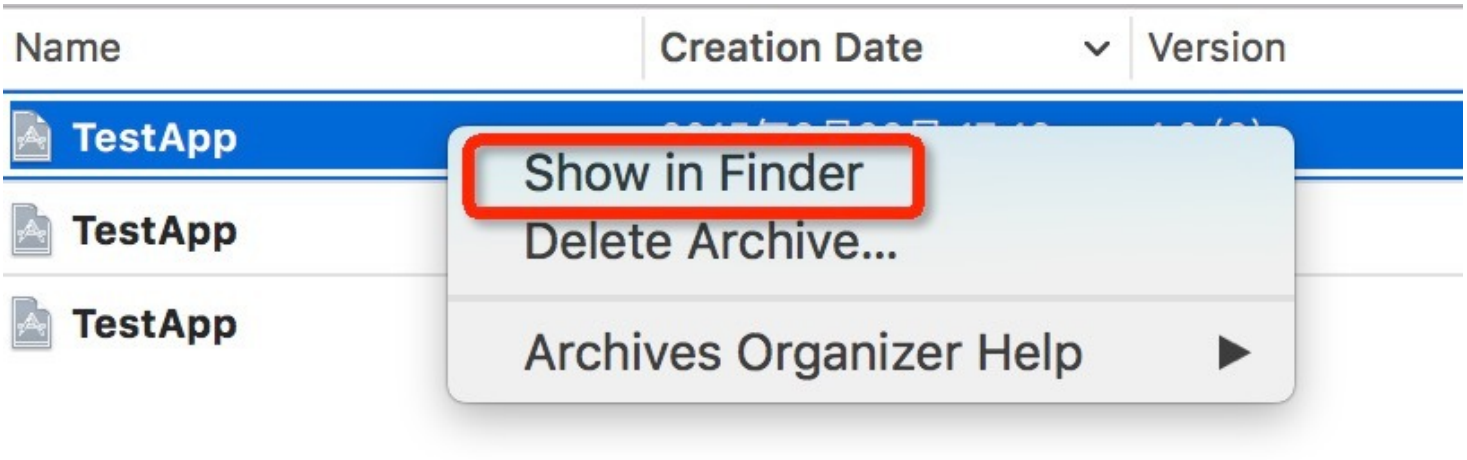
打开 Xcode 顶部菜单栏 -> Window -> Organizer 窗口：



打开 Xcode 顶部菜单栏，选择 Archive 标签：

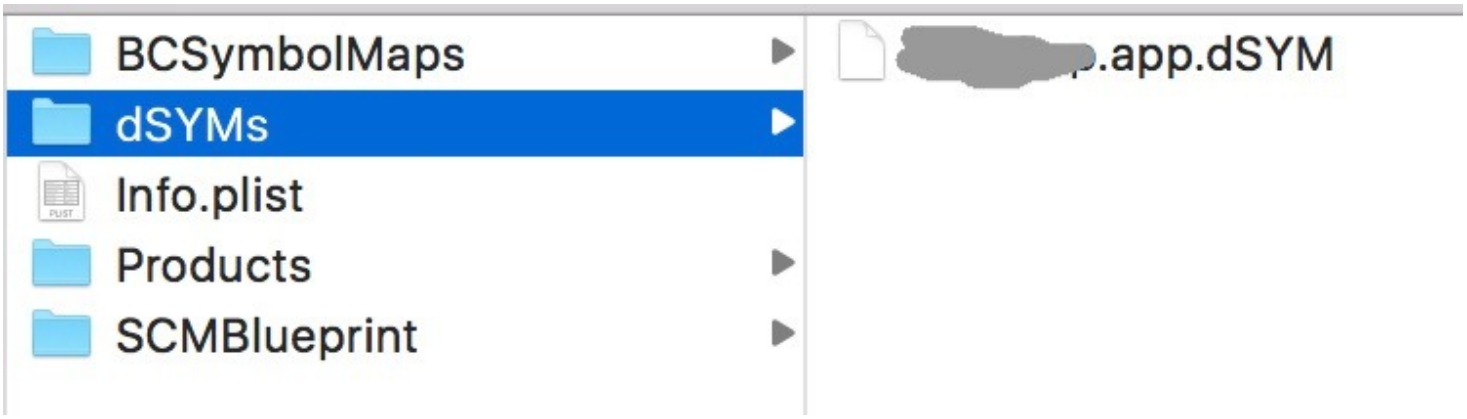


找到发布的归档包，右键点击对应归档包，选择Show in Finder操作：



右键选择定位到的归档文件，选择显示包内容操作：

选择dSYMs目录，目录内即为下载到的 dSYM 文件：



3.6.2 通过mdfind工具找回

在Bugly的issue页面查询到crash对应的UUID：

然后在Mac的Shell中，用mdfind命令定位dSYM文件：

```
mdfind "com_apple_xcode_dsym_uuids == <UUID>"
```

> 注意，使用mdfind时，UUID需要格式转换（增加“-”）：
> 12345678-1234-1234-1234-xxxxxxxxxxxx

例如，要定位的dSYM的UUID为：E30FC309DF7B3C9F8AC57F0F6047D65F
则定位dSYM文件的命令如下：

```
mdfind "com_apple_xcode_dsym_uuids == E30FC309-DF7B-3C9F-8AC5-7F0F6047D65F"
                                    |12345678-1234-1234-1234-xxxxxxxxxxxx|
```

3.7 找不到Crash对应的dSYM文件?

如果本地已经无法找到Crash对应的符号表文件或者dSYM文件了，但还能找回Crash对应的APP版本的工程代码，建议尝试重新用Xcode编译出dSYM文件并用符号表工具生成符号表文件。

如果连工程代码也无法找回了，那就真的无法还原这个Crash堆栈了。

为了防止出现这种情况，建议每次构建或者发布APP版本的时候，一定要备份好dSYM文件！