InfoBeat 智能数据平台

数据采集 iOS SDK 接入文档

（2019 年）

文件编号	Ijiami-技术-2019-i005		
编写人	谭振球	编写日期	20190712
审核人		审核日期	
批准人		批准日期	
发布时间		生效日期	
密级	完全公开	完全公开/内部使用/涉密	

■ **版本变更记录**

时间	版本	说明	修改人
2019.07.12	1.0.0	全新的威胁感知 SDK，更稳定，更好用	
2019.12	1.2.0	加入越狱检测	

■ 版权声明

■ 适用性说明

本文档主要面向需要接入数据采集 SDK 的 iOS 开发人员。本文只涉及教授数据采集 SDK 的集成方法，默认读者已经熟悉 Xcode 开发工具的基本使用方法，以及具有一定的编程知识基础等

目录

1、准备

接入前期准备工作包含:获取 AppKey 以及 SDK 文件（已完成用户可略过此步骤）

1.1、获取 appkey

AppKey 为接入 SDK 的必要参数，参数值需要到管理平台去创建应用获取；对于本地化部署，则需要在本地服务器上进行登陆管理页面进行分配 Appkey 参数。具体操作步骤如下：登入管理平台，在左侧菜单【系统配置】选项下选择【应用管理】

1. 打开应用管理模块，在右侧点击上传应用。见图 1-1-1

图 1-1-1

2. 提交完成，应用列表点击查看 AppKey。见图 1-1-2

图 1-1-2

1.2、目录结构

登入管理平台，在左侧菜单【系统配置】选项下选择【SDK 管理】，选择数据采集 SDK 下载 iOS 版本。（文件夹目录如下图 1-2-1）

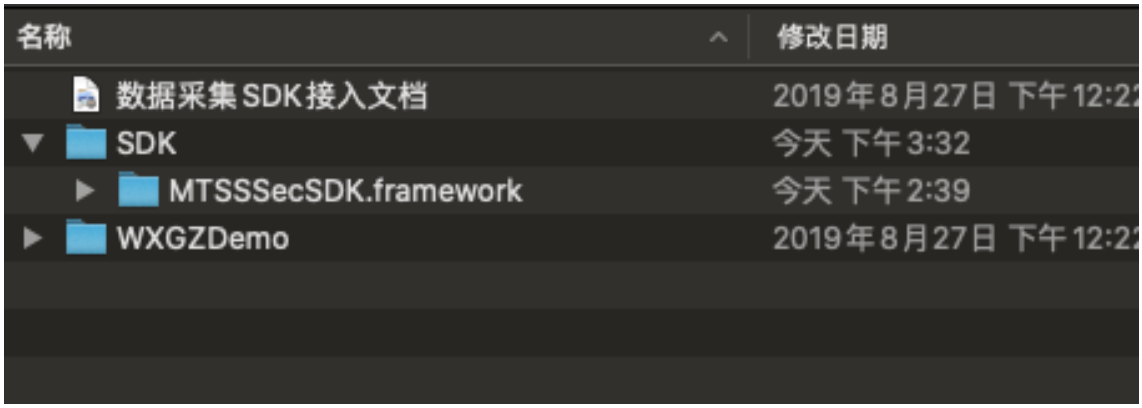

图 1-2-1

2、集成

- 自动集成 SDK 方式。使用 CocoaPods 的用户可以通过如下操作：

```
pod 'MTSSSDK', '~> 1.2.0'
```

注意（搜索之前在终端更新下 pod）

```
rm ~/Library/Caches/CocoaPods/search_index.json
Pod setup
```

- 手动集成方式。把下载的 SDK 包拖进工程内（包含：MTSSSecSDK.framework、MTSSSecSDK.bundle）

2.1、集成 SDK

在项目 build phases 如下图 2-1-1 所示配置

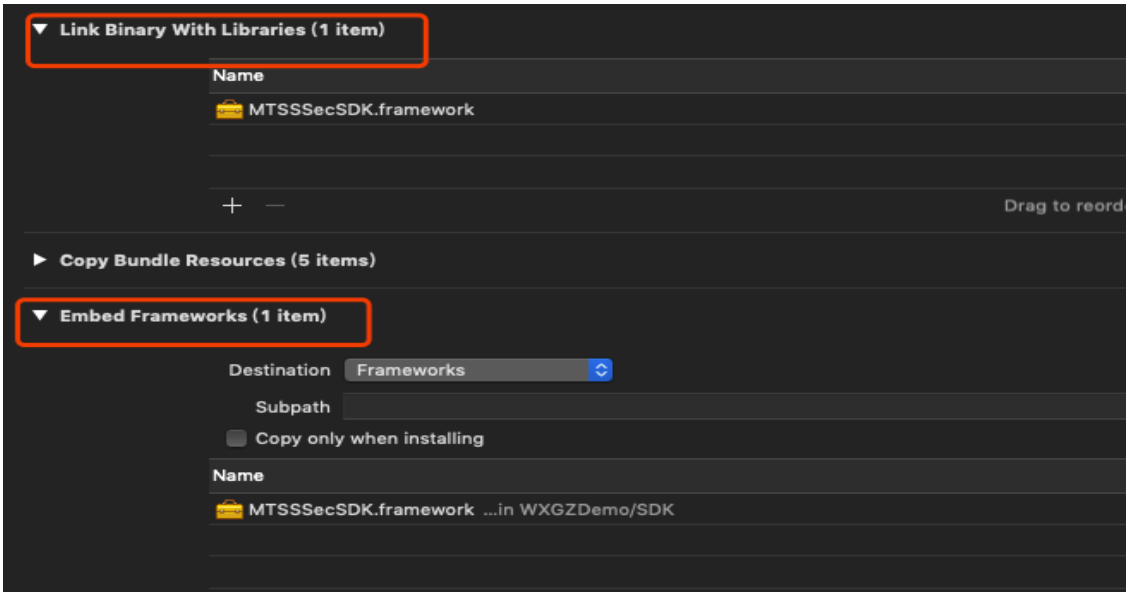

图 2-1-1

2.2、工程配置

1、在工程文件中选择 Build Setting，在"Other Linker Flags"中加入"-ObjC"（注意区分大小写）。见图 2-2-1

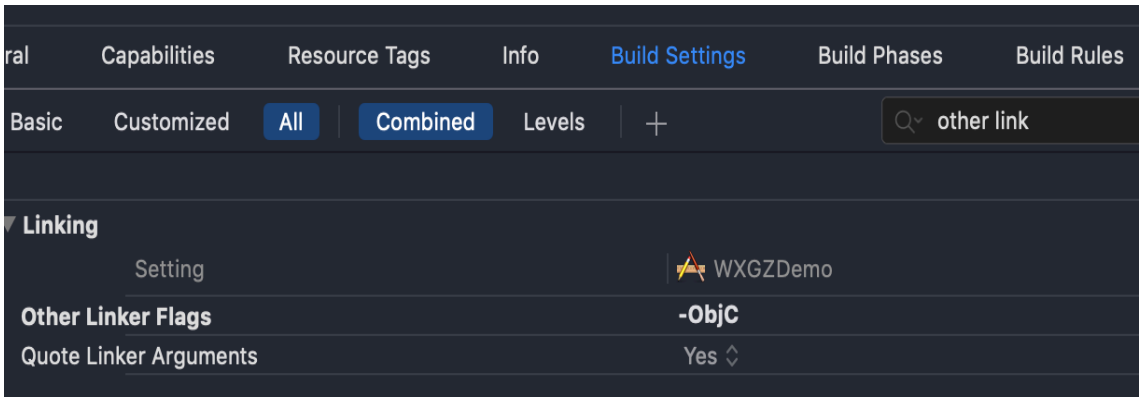

图 2-2-1

2、在你的工程文件中选择 Build Setting，在"Enable Bitcode"栏选择 NO。见图 2-2-2

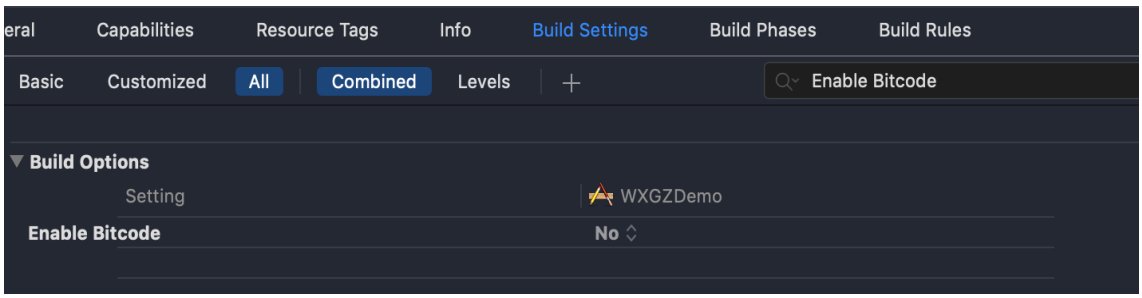

图 2-2-2

2.3、数据采集 SDK 对接

1、在 AppDelegate.m 文件里面导入 SDK 头文件

#import <MTSSSecSDK/MTSSSecSDK.h>

2、初始化 SDK

在 AppDelegate.m 文件 didFinishLaunchingWithOptions 方法中初始化 SDK

```
//1、初始化SDK环境
[MTSSSec  startWithReportedURL:@"http://172.10.4.94:8889"
                        appKey:@"18c2c8ac792d63abacfaff5e"];
//2、配置是不是每次检测定位授权
[MTSSSec setDetecteAndAuthorizedEveryLauchTime:YES
                             withAuthorizedType:MTSSAuthorizationTypeLocal];
//3、日志打印开关
[MTSSSec setLogEnable:NO];
```

图 2-3-1

3、FAQ

3.1、网络请求失败

如果你的 App 基于 iOS 9.0 编译，那么为了适配 iOS 9.0 中的 App Transport Security(ATS)对 HTTP 的限制，在 App 对应的 Info.list 中添加如下配置。（如果采用 https 方式则无需进行配置）

<key>NSAppTransportSecurity</key>

<dict>

<key>NSAllowsArbitraryLoads</key><true/>

</dict>

3.2、授权失败处理

首次启动用户拒绝推送或定位授权后，在 AppDelegate.m 的 didFinishLaunchingWithOptions 方法内加入 SDK 提供接口方法，可设置每次启动 App 时开启检测并提示去设置的授权弹框。（注：默认授权弹窗只在首次运行 App 弹出一次，特定的授权类型在设置 YES 之后，如果检测到未授权，每次启动 App 都会弹窗提示用户去设置）

具体设置如下图 3-2-1 所示：

```
//2、配置是不是每次检测定位授权
[MTSSSec setDetecteAndAuthorizedEveryLauchTime:YES withAuthorizedType:MTSSAuthorizationTypeLocal];
//3、日志打印开关
```

图 3-2-1

3.3、崩溃分析日志异常

发生崩溃以后，在管理中心并没有找到崩溃日志。可能是由于以下 4 种情况造成：

1. 崩溃以后没有重新启动 app，因为崩溃的时候来不及发送崩溃日志给服务器。所以需要重新启动 app 来进行网络请求发送日志到服务器。

2. iOS 捕获异常的函数 NSSetUncaughtExceptionHandler()被其他的崩溃收集类覆盖。导致威胁感知 SDK 捕获不到 NSSetUncaughtExceptionHandler 的回调消息。解决的办法是找出其他的崩溃收集类，再其捕获异常的时候先判断是否已经有其他的 handle 已经设置。可以先将该 handle 保存，然后将在收集完成异常以后回调该 handle。如图 3-3-2、3-3-3 所示

```
27
28  static  NSUncaughtExceptionHandler * temptempHand; 保存先前的handle
29
30  + (void)installCrashReportHandler
31  {
32
33      // OC崩溃
34      NSUncaughtExceptionHandler *tempHand = NSGetUncaughtExceptionHandler();
35      if (tempHand != NULL)
36      {
37          NSLog(@"检测到第三方已经监听了异常捕获");
38          temptempHand = tempHand;
39      }
40      NSSetUncaughtExceptionHandler(&MTSSuncaughtExceptionHandler);
41
42  }
43
44
```

图 3-3-2

```
3       if (temptempHand) {
4           NSLog(@"回调给第三方奔溃统计");
5           temptempHand(exception);
6       }
7
```

图 3-3-3

3. 没有连接网络，导致日志没法上报。

4. 发生崩溃的时候 app 还是断点状态，导致无法保存崩溃信息，放开断点即可完成崩溃信息收集。