

iOS 广告 SDK 接入说明

| 文档版本 | 修订日期 | 修订说明 |
|--------|------------|---|
| v2.6.9 | 2023-05-04 | 优化SDK视频渲染，图片渲染，模板样式 |
| v2.6.8 | 2023-04-24 | 优化Draw信息流，根据最新苹果政策要求，该版本及以后将基于Xcode14.1打包 |
| v2.6.7 | 2023-04-18 | 新增自渲染banner，完善上报 |
| v2.6.6 | 2023-03-24 | 优化聚合广告展示逻辑 |
| v2.6.5 | 2023-03-20 | 优化上报，竞价广告支持缓存控制 |
| v2.6.4 | 2023-03-20 | 新增预加载广告缓存功能 |
| v2.6.3 | 2023-01-06 | 修复已知问题，优化上报 |
| v2.6.2 | 2022-12-12 | 优化广告请求展示逻辑，优化缓存逻辑 |
| v2.6.1 | 2022-11-18 | 支持快手server-bidding，增加广告填充失败的code上报 |
| v2.6.0 | 2022-11-04 | 优化广告链路转化，提升广告收益 |
| v2.5.9 | 2022-10-28 | 优化广告上报 |
| v2.5.8 | 2022-09-22 | 修复一些已知问题，优化广告请求 |
| v2.5.7 | 2022-08-26 | 增加SDK错误统计 |
| v2.5.6 | 2022-08-12 | 新增快速请求 |
| v2.5.5 | 2022-07-28 | 适配京准通SDK新版本 |
| v2.5.3 | 2022-07-17 | 增加自渲染插屏样式,支持pod接入 |
| v2.5.2 | 2022-07-07 | 增加百度、优量汇SDK—bidding，增加激励视频奖励透传 |
| v2.4.9 | 2022-06-21 | 增加兜底广告 |
| v2.4.7 | 2022-06-11 | 增加自定义广告Adapter |
| v2.4.6 | 2022-06-01 | 增加广告缓存 |
| v2.4.5 | 2022-05-19 | 优化上报，增加优量汇竞价功能，增加配置缓存功能 |
| v2.4.1 | 2022-04-25 | 优化开屏展示 |
| v2.4.0 | 2022-04-19 | 增加视频上报 |
| v2.3.7 | 2022-04-12 | naviveAd支持自定义视频播放器 |
| v2.3.6 | 2022-04-01 | 修复已知问题 |

| | | |
|--------|------------|--|
| v2.3.5 | 2022-03-25 | 适配API广告参数 |
| v2.3.0 | 2022-02-28 | 支持server-bidding |
| v2.2.0 | 2022-01-07 | 优化广告展示性能 |
| v2.1.0 | 2021-11-25 | 修复已知问题，提升广告性能 |
| v2.0.5 | 2021-11-10 | 升级联盟SDK至最新 |
| v2.0.3 | 2021-10-30 | bug修复，性能优化 |
| v2.0.2 | 2021-10-20 | 内部版本测试 |
| v2.0.1 | 2021-10-15 | 支持全屏视频广告，文档Demo优化，其他问题修复 |
| v2.0.0 | 2021-10-05 | 支持Sigmob SDK |
| v1.0.5 | 2021-09-20 | 支持百度联盟，京准通SDK，优化已知问题 |
| v1.0.0 | 2021-09-06 | 创建文档，支持穿山甲，优量汇，快手，支持开屏，模板，横幅，原生自渲染，插屏，激励视频广告样式 |

前置说明

使用本SDK前必须满足以下条件。具体详情请联系官方指定负责人或到指定平台申请！

参数申请

目前聚合广告SDK需要的参数为APPID和各广告位对应ID，请在官方指定平台申请或联系具体负责人！

支持的广告联盟和广告类型

| 广告联盟 | 开屏广告 | 原生广告 | 模板广告 | 横幅广告 | 插屏广告 | 全屏视频广告 | 激励视频广告 |
|-----------|------|------|------|------|------|--------|--------|
| 穿山甲 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 优量汇 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 快手 | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ |
| 京准通 | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ |
| 百度联盟 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Sigmob | ✓ | ✓ | - | ✗ | ✓ | ✓ | ✓ |
| Mintegral | ✓ | ✓ | - | ✗ | ✓ | ✓ | ✓ |

平台配置

接入方需事先在各三方SDK平台申请相关广告参数，然后在聚合平台进行配置。在聚合平台配置之后，才能正常使用本SDK的聚合功能。

SDK包体影响

| 接入模块 | 包体影响(M) |
|-----------------|---------|
| MSaas.framework | 0.7 |

三方SDK导入

聚合SDK依赖于被聚合的三方SDK，故在接入本SDK之前请确保项目已经接入了参与聚合的第三方广告联盟SDK。

| 三方联盟SDK | 需要导入的联盟包 | 需要导入的广告桥接包 |
|-----------|--|-------------------------------|
| 穿山甲 | CSJAdSDK.framework CSJAdSDK.bundle BUAdSDK.framework BURelyFoundation.framework | SFAdPangolinAdapter.framework |
| 优量汇 | libGDTMobSDK.a | SFAdGdtAdapter.framework |
| 快手 | KSAdSDK.xcframework | SFAdKsAdapter.framework |
| 京准通 | JADYun.framework | SFAdJztAdapter.framework |
| 百度联盟 | BaiduMobAdSDK.framework baidumobadsdk.bundle | SFAdBaiduAdapter.framework |
| Sigmob | WindFoundation.xcframework WindSDK.xcframework | SFAdSigmobAdapter.framework |
| Mintegral | MTGSDK.xcframework MTGSDKBanner.xcframework MTGSDKBidding.xcframework MTGSDKInterstitial.xcframework MTGSDKInterstitialVideo.xcframework MTGSDKNativeAdvanced.xcframework MTGSDKNewInterstitial.xcframework MTGSDKReward.xcframework MTGSDKSplash.xcframework | SFAdMtgAdapter.framework |

SDK接入

展示广告接入

1.1 申请应用的应用ID 和 广告位ID

开发者需在平台创建应用和广告位，生成对应的应用ID和广告位ID。

1.2 导入framework

1.2.1 SDK集成

方法1 pod接入

```
# MSaas主包
pod 'MSaas', :git => 'https://github.com/xiaofu666/MSaas'

# Adapter插件，根据实际按需拖入：
# 穿山甲广告主适配包
pod 'SFAdPangolinAdapter'
# 广点通广告主适配包
pod 'SFAdGdtAdapter'
```

```
# 快手广告主适配包
pod 'SFAdKsAdapter'
# 京准通广告主适配包
pod 'SFAdJztAdapter'
# 百度广告主适配包
pod 'SFAdBaiduAdapter'
# sigmob广告主适配包
pod 'SFAdSigmobAdapter'
# Mintegral广告主适配包
pod 'SFAdMtgAdapter'
```

方法2 工程设置导入framework

获取相应版本的framework库，导入项目工程即可。

聚合SDK的framework库结构如下：

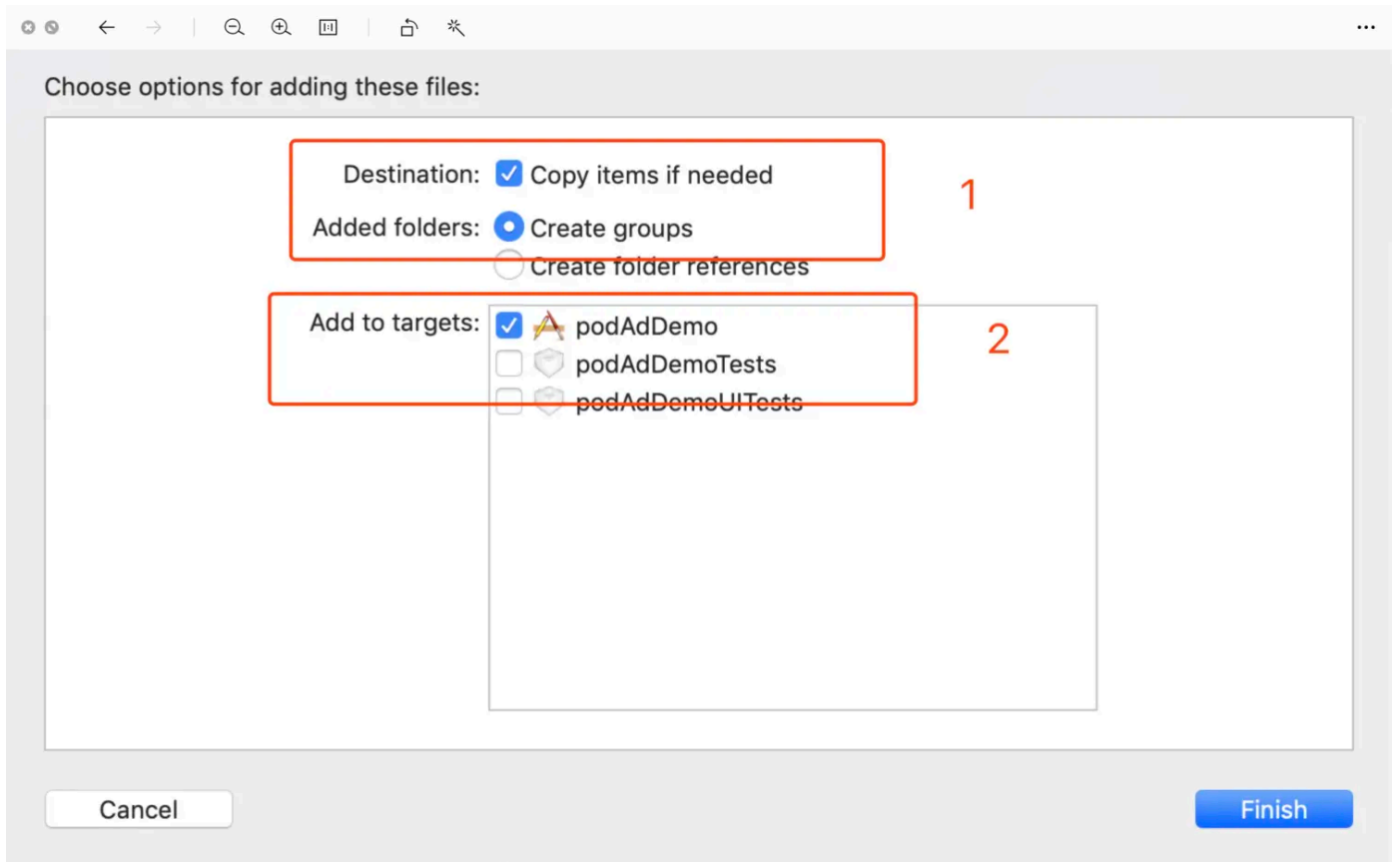
聚合SDK:

- MSaas.framework

Adapter插件，根据实际按需拖入：

- SFAdPangolinAdapter.framework
- SFAdGdtAdapter.framework
- SFAdKsAdapter.framework
- SFAdJztAdapter.framework
- SFAdBaiduAdapter.framework
- SFAdSigmobAdapter.framework
- SFAdMtgAdapter.framework

拖入时请按以下方式选择：



1.2.2 Xcode编译选项设置

1.2.2.1 添加权限

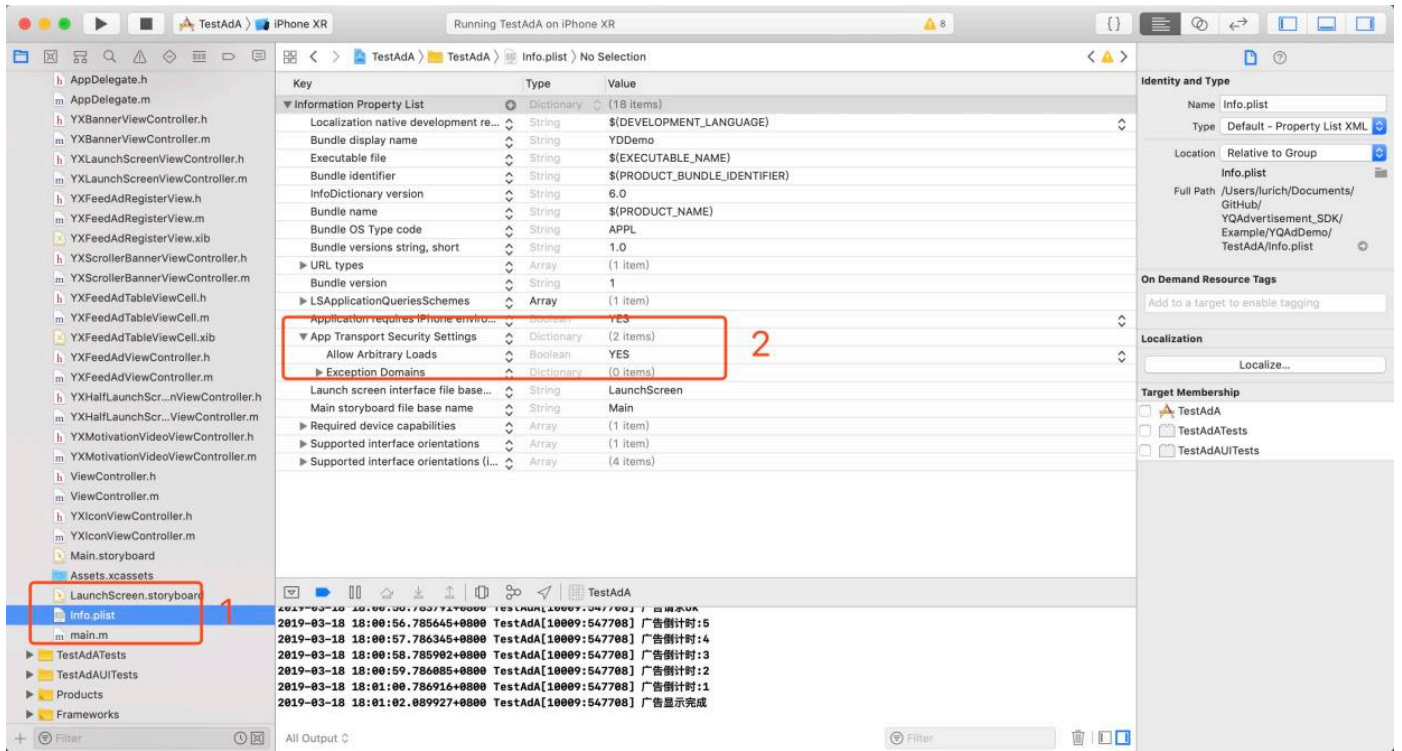
注意要添加的系统库

- 工程plist文件设置，点击右边的information Property List后边的 "+" 展开

添加 App Transport Security Settings，先点击左侧展开箭头，再点右侧加号，Allow Arbitrary Loads 选项自动加入，修改值为 YES。SDK API 已经全部支持HTTPS，但是广告主素材存在非HTTPS情况。

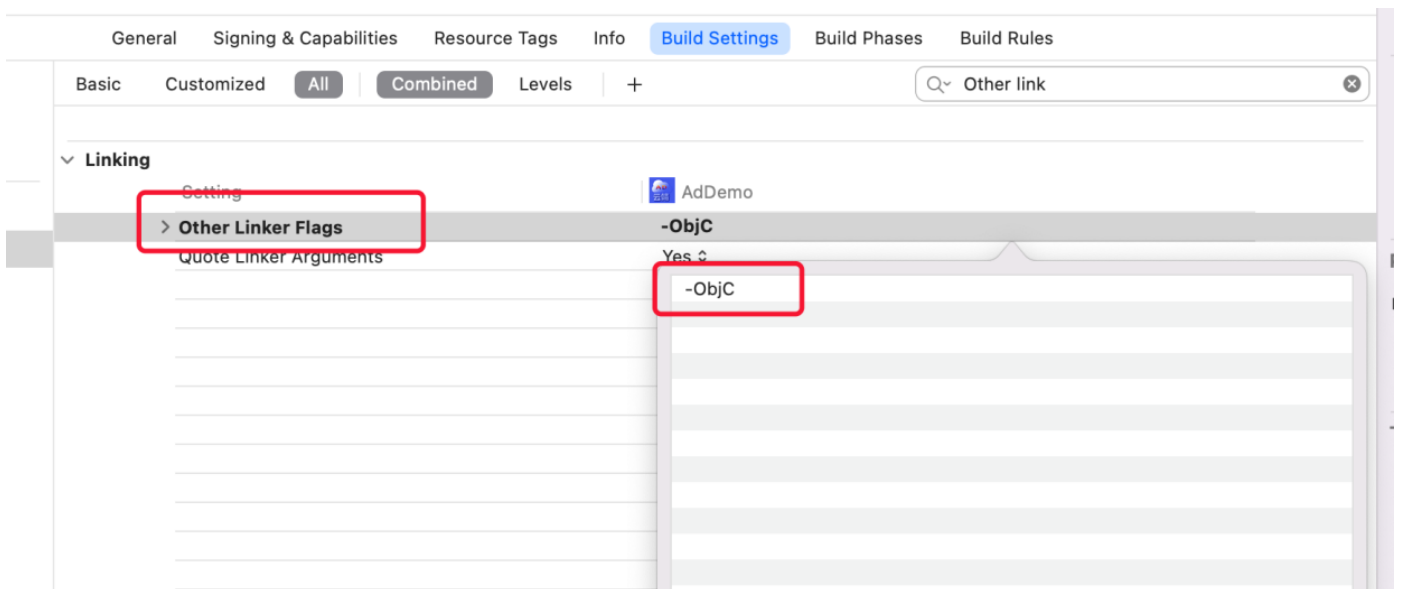
```
<key>NSAppTransportSecurity</key>
  <dict>
    <key>NSAllowsArbitraryLoads</key>
    <true/>
  </dict>
```

具体操作如图：

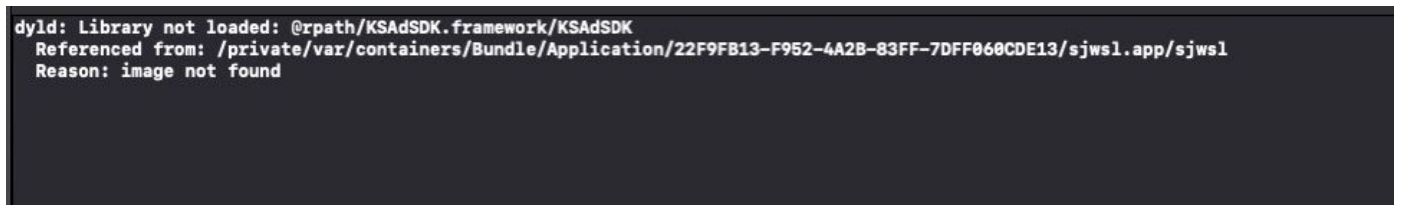


- Build Settings中Other Linker Flags 增加参数-ObjC

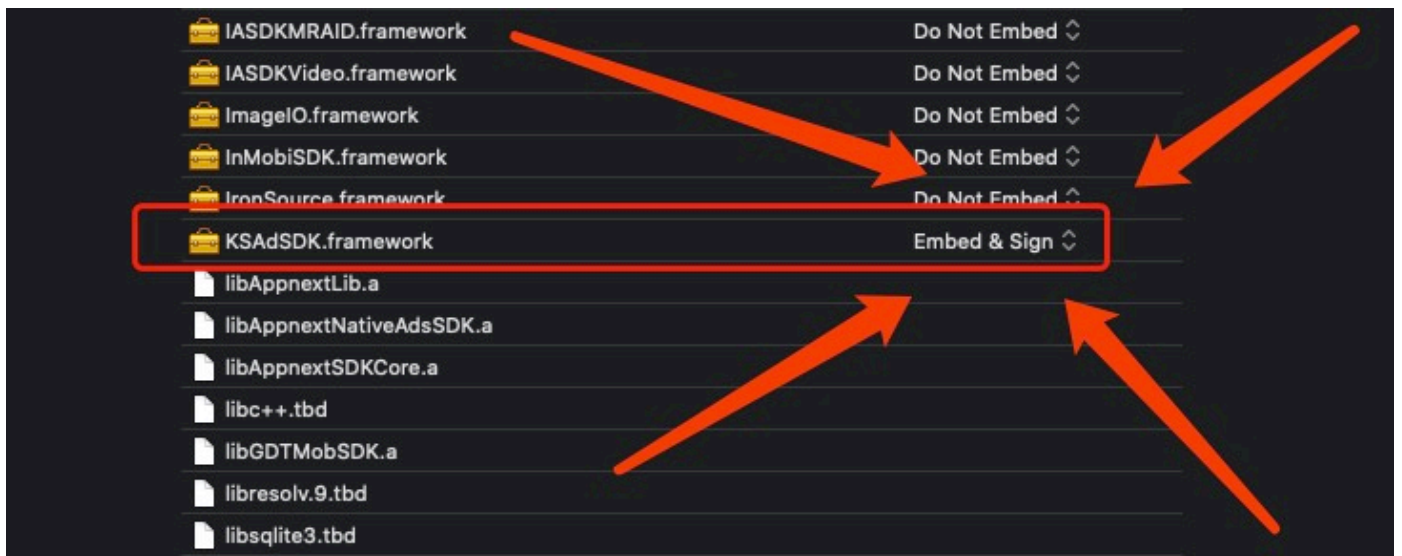
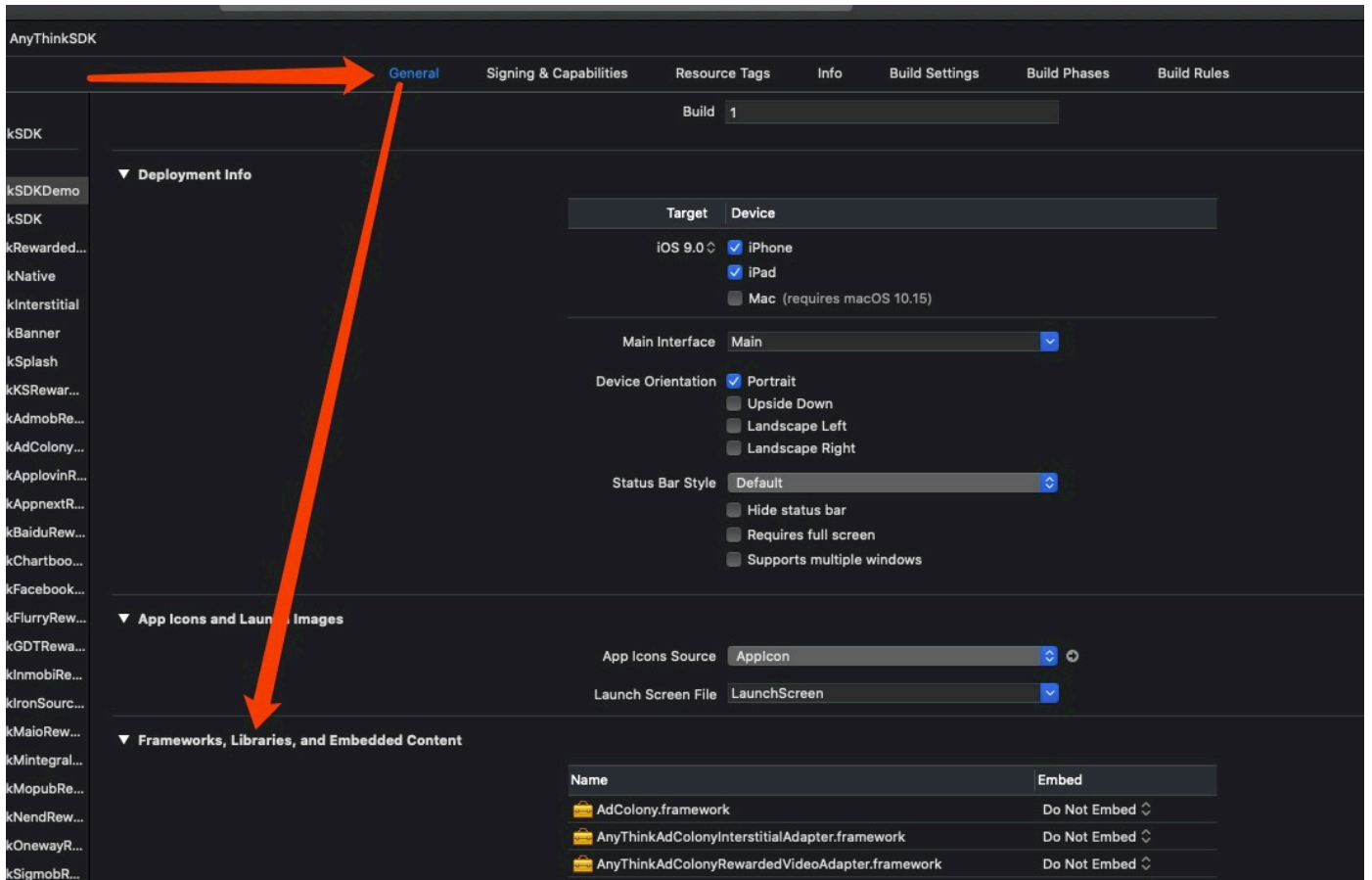
具体操作如图：



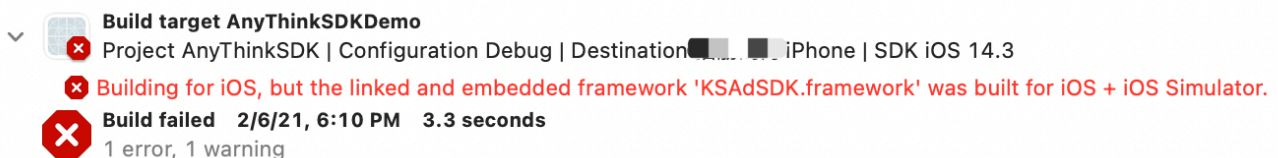
引入快手SDK编译崩溃怎么办？



您遇到这个崩溃是因为快手sdk为动态库，需要将KSAdSDK.framework的Embed修改为Embed&Sign即可



在修改完Embed之后，若是编译还是报错，如下：



解决方法：Build Settings中的Validate Workspace修改为 YES

- 京准通SDK为Swift和OC混编，所以集成时需要配置混编环境！
 1. 如果是OC项目，新建一个.swift文件，Xcode会提示创建桥接文件，点击确认即可
 2. 如果是Swift项目，新建一个OC文件，Xcode会提示创建桥接文件，点击确认即可

- SDK中包含获取IDFA的权限,所以需要在info.plist中添加IDFA权限,如图所示:

```
<key>NSUserTrackingUsageDescription</key>
<string>该标识符将用于向您投放个性化广告</string>
```

| | | |
|--------------------------------------|--------|---------------------------------|
| Privacy - Tracking Usage Description | String | 请允许获取并使用您的IDFA，该标识符将用于向您投放个性化广告 |
|--------------------------------------|--------|---------------------------------|

- 将联盟SDK的 SKAdNetwork ID 添加到 info.plist 中，以保证 SKAdNetwork 的正确运行（可自行添加对应联盟SDK的SKAdNetwork）,如图所示:

```
<key>SKAdNetworkItems</key>
<array>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>r3y5dwb26t.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>238da6jt44.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>x2jnk7ly8j.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>22mmun2rn5.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>f7s53z58qe.skadnetwork</string>
  </dict>
</array>
```

| | | |
|--------------------------------------|------------|------------------------|
| Privacy - Tracking Usage Description | String | 该标识符将用于向您投放个性化广告 |
| SKAdNetworkItems | Array | (5 items) |
| Item 0 | Dictionary | (1 item) |
| SKAdNetworkIdentifier | String | r3y5dwb26t.skadnetwork |
| Item 1 | Dictionary | (1 item) |
| SKAdNetworkIdentifier | String | 238da6jt44.skadnetwork |
| Item 2 | Dictionary | (1 item) |
| SKAdNetworkIdentifier | String | x2jnk7ly8j.skadnetwork |
| Item 3 | Dictionary | (1 item) |
| SKAdNetworkIdentifier | String | 22mmun2rn5.skadnetwork |
| Item 4 | Dictionary | (1 item) |

- 为了提高广告 ADX的收益效果，建议您复制以下代码到您App的info.plist文件:

```
<key>LSApplicationQueriesSchemes</key>
<array>
```



```
<string>tbopen</string>
<string>openapp.jdmobile</string>
<string>imeituan</string>
<string>pinduoduo</string>
<string>youku</string>
<string>iqiyi</string>
<string>alipays</string>
<string>vipshop</string>
<string>tmall</string>
<string>sinaweibo</string>
<string>zhihu</string>
<string>jdmobile</string>
<string>autohome</string>
<string>taobaolite</string>
<string>taobaoliveshare</string>
<string>eleme</string>
</array>
```

1.2.2.2 运行环境配置

- 支持系统 iOS 9.0 及以上;
- SDK编译环境 Xcode 9.3;
- 支持架构: i386, x86-64, armv7, arm64

1.2.2.3 添加依赖库(pod接入可忽略此操作)

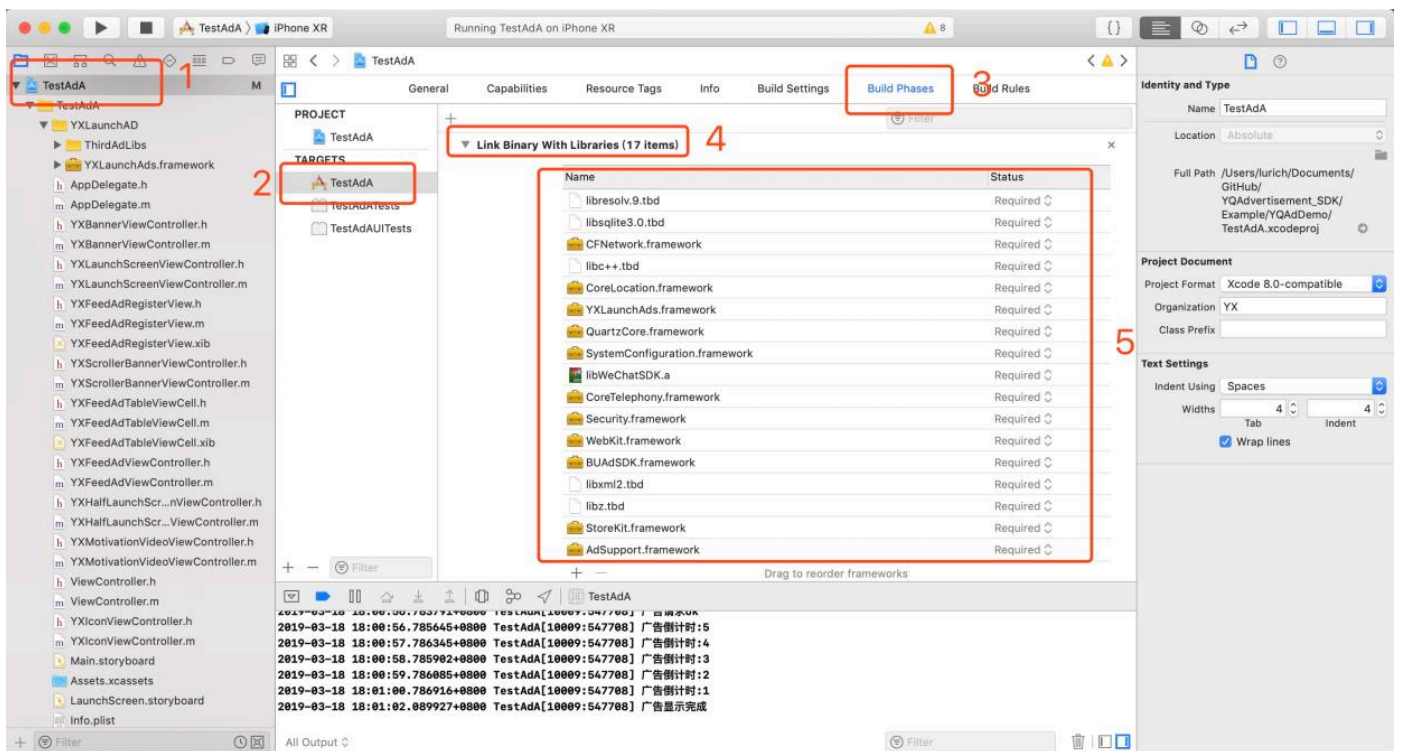
以下依赖库已整合所有联盟广告主所需的依赖库

工程需要在TARGETS -> Build Phases中找到Link Binary With Libraries, 点击"+", 依次添加下列依赖库

- libresolv.9.tbd
- libc++.tbd
- libc++abi.tbd
- libz.tbd
- libbz2.tbd
- libxml2.tbd
- libiconv.tbd
- libsqlite3.tbd
- StoreKit.framework
- MobileCoreServices.framework
- WebKit.framework
- MediaPlayer.framework
- CoreMedia.framework
- AVFoundation.framework
- CoreLocation.framework
- CoreTelephony.framework
- SystemConfiguration.framework
- AdSupport.framework
- CoreMotion.framework
- Security.framework

- QuartzCore.framework
- CoreGraphics.framework
- SafariServices.framework
- UIKit.framework
- Foundation.framework
- JavaScriptCore.framework
- MapKit.framework
- AssetsLibrary.framework
- AppTrackingTransparency.framework
- MessageUI.framework
- DeviceCheck.framework

具体操作如图所示：



1.2.3. 获取广告标识IDFA

从iOS 14开始，只有在获得用户明确许可的前提下，应用才可以访问用户的IDFA数据并向用户投放定向广告。在应用程序调用 [App Tracking Transparency](#) 框架向最终用户提出应用程序跟踪授权请求之前，IDFA将不可用。如果某个应用未提出此请求，则读取到的IDFA将返回全为0的字符串。

1.2.3.1 在info.plist文件里添加跟踪权限请求描述说明

以下提供中英文描述说明示例，开发者可参考添加其中一种即可：

- 中文版：

```
<key>NSUserTrackingUsageDescription</key>
<string>该标识符将用于向您投放个性化广告。</string>
```

| Key | Type | Value |
|--------------------------------------|------------|--|
| ▼ Information Property List | Dictionary | (21 items) |
| Privacy - Tracking Usage Description | String | This identifier will be used to deliver personalized ads to you. |

- 英文版：

```
<key>NSUserTrackingUsageDescription</key>
<string>This identifier will be used to deliver personalized ads to you.</string>
```

| Key | Type | Value |
|--------------------------------------|------------|--|
| ▼ Information Property List | Dictionary | (21 items) |
| Privacy - Tracking Usage Description | String | This identifier will be used to deliver personalized ads to you. |

1.2.3.2 获取App Tracking Transparency权限

想要获取授权，需要使用[requestTrackingAuthorizationWithCompletionHandler](#)。我们建议您在初始化 Mediatom SDK之前获取授权，以便如果用户授予允许跟踪权限，Mediatom SDK则可以在广告请求中使用IDFA。

```
#import <AppTrackingTransparency/AppTrackingTransparency.h>

if (@available(iOS 14, *)) {
    //iOS 14
    [ATTrackingManager
requestTrackingAuthorizationWithCompletionHandler:^(ATTrackingManagerAuthorizationStatus status) {
        //to do something, like preloading
        // load AD
    }];
} else {
    // load AD
}
```

描述说明将会显示在App Tracking Transparency授权对话框中，如下：

获取 IDFA

允许“Demo”跟踪你在其他公司的 App 和网站上的活动吗？
该 ID 将用于向您推送个性化广告

要求 App 不跟踪

允许

注意：

1. 如果有预加载广告需求的，强烈建议把预加载写在 `requestTrackingAuthorizationWithCompletionHandler` 的 block 回调中。
2. 该权限只有 Xcode 12 及以上版本才有，需要大家更新 Xcode 12 版本来进行测试使用。

1.2.3.3 iOS 15 获取 App Tracking Transparency 权限说明

iOS 15 对 **AppTrackingTransparency** 框架做了调整，需要满足以下 2 点：

- 应用当前的状态为 `UIApplicationStateActive`。
- 当前没有其他的授权弹窗。

当满足以上 2 点后，再检查 `ATTrackingManagerAuthorizationStatus` 和请求授权。因此，可以在以下的选项中选择一个作为调整方案：

- 启动应用后，延时 1~2 秒去检查和申请权限。
- 在主控制器的 “`ViewDidAppear`” 方法中检查和申请权限。

参考链接：<https://developer.apple.com/forums/thread/690607>

1.3 SDK 接口类介绍与广告接入

1.3.0 全局设置

SDK的开屏广告建议在 AppDelegate 的方法 `- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions` 里

当window指定root控制器并显示之后最先进行初始化

```
[SFAdSDKManager registerAppId:@"从后台获取的应用ID"];
```

1.3.1 开屏广告

- **类型说明：**开屏广告主要是 APP 启动时展示的全屏广告视图，开发只要按照接入标准就能够展示设计好的视图(注意：开屏接入代码必须放在window指定rootViewController之后)。具体可参考Demo中 LaunchScreenViewController部分示例代码

1. 导入

```
#import <MSaas/MSaas.h>
```

2. 遵循代理

```
<SFSplashDelegate>
```

3. 广告页面呈现在keywindow上，建议开屏广告的初始化放在window指定root控制器并makeKeyAndVisible之后。

- 全屏示例：

1. 请求开屏广告

```
_manager = [SFSplashManager new];  
_manager.delegate = self;  
_manager.mediaId = @"获取的广告位";  
_manager.showAdController = self.window.rootViewController;  
[_manager loadAdData];
```

2. 展示开屏广告

```
[self.manager showSplashAdWithWindow:[UIApplication sharedApplication].keyWindow];
```

- 非全屏示例：

1. 请求开屏广告

```
//bottom 为含logo的view  
UILabel *bottom = [[UILabel alloc] initWithFrame:CGRectMake(0, 0, [UIScreen  
mainScreen].bounds.size.width, [UIScreen mainScreen].bounds.size.height * 0.2)];
```

```

bottom.text = @"AD Demo";
bottom.textAlignment = NSTextAlignmentCenter;
bottom.font = [UIFont systemFontOfSize:35];
bottom.userInteractionEnabled = YES;
bottom.backgroundColor = [UIColor whiteColor];

_manager = [SFSplashManager new];
_manager.delegate = self;
_manager.mediaId = @"获取的广告位";
_manager.bottomView = bottom;
_manager.showAdController = self.window.rootViewController;
[_manager loadAdData];

```

2. 展示开屏广告

```
[self.manager showSplashAdWithWindow:[UIApplication sharedApplication].keyWindow];
```

建议等待时间设置为5秒，展示时间设置为5秒。
App在从后台5分钟后到前台时 建议也加上开屏广告。

1.3.2 原生信息流广告

- **类型说明：** 原生广告即一般广告样式，形式分为图文和视频。
- **使用说明：** 在SDK里只需要使用 SFFeedManager 就可以获取原生广告，SFFeedManager 类提供了原生广告的数据类型等各种信息，在数据获取后可以在 datas（SFFeedAdData）里面获取广告数据信息。具体可参考 Demo中FeedAdViewController部分示例代码。

1. 导入

```
#import <MSaas/MSaas.h>
```

2. 遵循代理

```
<SFFeedDelegate>
```

3. 请求广告

```

self.feedManager = [[SFFeedManager alloc] init];
self.feedManager.mediaId = @"获取的广告位";
self.feedManager.adCount = 1;
self.feedManager.showAdController = self;
self.feedManager.delegate = self;
[self.feedManager loadAdData];

```

- 广告的代理回调

```

/**
 * 广告数据：加载成功
 */
- (void)feedAdDidLoadDatas:(NSArray<__kindof SFFeedAdData *> *)datas;
/**
 * 广告数据：加载失败
 * @param error : 错误信息
 */
- (void)feedAdDidFailed:(NSError *)error;
/**
 * 广告视图：展示
 */
- (void)feedAdDidVisible;
/**
 * 广告视图：点击
 */
- (void)feedAdDidClicked;
/**
 * 落地页或者appstoe返回事件
 */
- (void)feedAdDidCloseOtherController;
/**
 * 视频广告播放状态更改回调
 * @param status 视频广告播放状态
 */
- (void)feedAdViewPlayerStatusChanged:(SFMediaPlayerStatus)status;

```

其中 `SFMediaPlayerStatus` 为枚举类型，如下所示：

```

typedef NS_ENUM(NSUInteger, SFMediaPlayerStatus) {
    SFMediaPlayerStatusInitial = 0,           // 初始状态
    SFMediaPlayerStatusLoading = 1,           // 加载中
    SFMediaPlayerStatusStarted = 2,           // 开始播放
    SFMediaPlayerStatusPaused = 3,            // 用户行为导致暂停
    SFMediaPlayerStatusError = 4,             // 播放出错
    SFMediaPlayerStatusStoped = 5,            // 播放停止
};

```

在广告渲染到屏幕上之后，需要调用 `registerAdViewForBindImage` 方法注册广告的点击事件，如下所示：

```

[self.feedManager registerAdViewForBindImage:adImageView adData:adData
clickableViews:@[clickViews]];

```

1.3.3 横幅广告

- **类型说明：**横幅广告是为满足媒体多元化需求而开发的一种广告。SDK会返回已渲染完成的广告视图,开发只需展示即可,避免了接入方的大量工作量。
- **使用说明：**横幅广告 使用 SFBannerManager 对象调用 loadAdData 请求广告，使用 showBannerAdWithView 添加广告对象来进行广告展示，通过设置 SFBannerDelegate 代理，获取广告获取成功、广告获取失败、点击、从落地页返回、关闭等回调。具体可参考Demo中 BannerViewController 部分示例代码

1. 导入

```
#import <MSaas/MSaas.h>
```

2. 遵循代理

```
<SFBannerDelegate>
```

3. 请求广告

```
self.banner = [SFBannerManager new];
self.banner.size = self.bannerView.bounds.size;
self.banner.delegate = self;
self.banner.mediaId = @"获取的广告位";
self.banner.showAdController = self;
[self.banner loadAdData];
```

- 广告的代理回调

```
/**
 * 广告数据：加载成功
 */
- (void)bannerAdDidLoad;
/**
 * 广告数据：加载失败
 * @param error : 错误信息
 */
- (void)bannerAdDidFailed:(NSError *)error;
/**
 * 广告视图：展示
 */
- (void)bannerAdDidVisible;
/**
 * 广告视图：点击
 */
- (void)bannerAdDidClick;
/**
 * 落地页或者appstoe返回事件
 */
```



```

- (void)bannerAdDidCloseOtherController;
/**
 * 广告视图：关闭
 */
- (void)bannerAdDidClose;

```

- 在广告成功的回调中展示广告：

```
[self.banner showBannerAdWithView:self.bannerView];
```

1.3.4 插屏广告

- **类型说明：**插屏广告是为满足媒体多元化需求而开发的一种广告。SDK会返回已渲染完成的广告视图,开发只需展示即可,避免了接入方的大量工作量。
- **使用说明：**插屏广告 使用 SFInterstitialManager 对象调用 loadAdData 请求广告，使用 showInterstitialAd 添加广告对象来进行广告展示，通过设置 SFInterstitialDelegate 代理，获取广告获取成功、广告获取失败、点击、从落地页返回、关闭等回调。具体可参考Demo中 InterstitialViewController 部分示例代码

1. 导入

```
#import <MSaas/MSaas.h>
```

2. 遵循代理

```
<SFInterstitialDelegate>
```

3. 请求广告

```

self.interstitialAd = [SFInterstitialManager new];
self.interstitialAd.mediaId = @"获取的广告位";
self.interstitialAd.showAdController = self;
self.interstitialAd.delegate = self;
[self.interstitialAd loadAdData];

```

- 广告的代理回调

```

/**
 * 广告数据：加载成功
 */
- (void)interstitialAdDidLoad;
/**
 * 广告数据：加载失败
 * @param error : 错误信息
 */
- (void)interstitialAdDidFailed:(NSError *)error;
/**

```

```

* 广告视图: 展示
*/
- (void)interstitialAdDidVisible;
/**
* 广告视图: 点击
*/
- (void)interstitialAdDidClick;
/**
* 落地页或者appstoe返回事件
*/
- (void)interstitialAdDidCloseOtherController;
/**
* 广告视图: 关闭
*/
- (void)interstitialAdDidClose;

```

- 在广告成功的回调中展示广告：

```
[self.interstitialAd showInterstitialAd];
```

1.3.5 模板广告

- **类型说明：** 模板广告即视图广告,SDK会返回已渲染完成的广告视图数组,开发只需展示即可,避免了接入方的大量工作量。
- **使用说明：** 在SDK里只需要使用 SFTemplateManager 就可以获取模板广告, SFTemplateManager 类提供了模板广告的各种信息, 具体可参考Demo中 NativeExpressAdController 的部分示例代码。

1. 导入

```
#import <MSaas/MSaas.h>
```

2. 遵循代理

```
<SFTemplateDelegate>
```

3. 请求广告

```

self.manager = [[SFTemplateManager alloc] init];
self.manager.mediaId = @"获取的广告位";
self.manager.adCount = 1;
//广告view大小尺寸,高度为0时,将自适应高度(推荐高度传0进行自适应高度)
self.manager.size = CGSizeMake(SF_ScreenW, 0);
self.manager.showAdController = self;
self.manager.delegate = self;
[self.manager loadAdData];

```

- 广告的代理回调

```
/**
 * 广告数据：加载成功
 */
- (void)templateAdDidLoadViews:(NSArray<__kindof UIView *> *)views;
/**
 * 广告数据：加载失败
 * @param error : 错误信息
 */
- (void)templateAdDidFailed:(NSError *)error;
/**
 * 广告视图：点击
 */
- (void)templateAdDidClickedWithADView:(UIView *)templateAdView;
/**
 * 广告视图：渲染成功
 */
- (void)templateAdDidRenderSuccessWithADView:(UIView *)templateAdView;
/**
 * 落地页或者appstoe返回事件
 */
- (void)templateAdDidCloseOtherControllerWithADView:(UIView *)templateAdView;
/**
 * 广告视图：关闭
 */
- (void)templateAdDidCloseWithADView:(UIView *)templateAdView;
```

- 在 `- (void)templateAdDidLoadViews:(NSArray<__kindof UIView *> *)views` 获取广告视图并展示广告：
- 在 `- (void)templateAdDidRenderSuccessWithADView:(UIView *)templateAdView` 回调中获取广告视图的高度刷新界面；

1.3.6 激励视频

- **类型说明：**激励视频广告是一种全新的广告形式，用户可选择观看视频广告以换取有价值物，例如虚拟货币、应用内物品和独家内容等等；这类广告的长度为 5-60 秒，且广告的开始画面会显示激励画面，引导用户进行后续动作。通过设置 SFRewardVideoDelegate 代理，获取广告获取成功、广告获取失败、点击、播放达到激励条件、关闭等回调。具体可参考Demo中 MotivationVideoViewController 部分示例代码。

1. 导入

```
#import <MSaas/MSaas.h>
```

2. 遵循代理

```
<SFRewardVideoDelegate>
```

3. 创建属性对象

```
@property (nonatomic, strong) SFRewardVideoManager *motivationVideo;
```

4. 使用示例

```
self.motivationVideo = [SFRewardVideoManager new];  
self.motivationVideo.mediaId = @"获取的广告位";  
self.motivationVideo.delegate = self;  
[self.motivationVideo loadAdDataWithExtra:nil];
```

5. 在广告获取成功的回调里面展示广告

```
[self.motivationVideo showRewardVideoAdWithController:self];
```

● 广告的代理回调

```
/**  
 * 激励视频广告-视频-加载成功  
 */  
- (void)rewardedVideoDidLoad;  
  
/**  
 * 激励视频广告素材加载失败  
 * @param error 错误对象  
 */  
- (void)rewardedVideoDidFailWithError:(NSError *)error;  
  
/**  
 * 激励视频广告成功展示  
 */  
- (void)rewardedVideoDidVisible;  
  
/**  
 * 激励视频广告点击  
 */  
- (void)rewardedVideoDidClick;  
  
/**  
 * 激励视频广告播放达到激励条件  
 * @param extra 额外参数，即初始化传入的extra  
 */  
- (void)rewardedVideoDidRewardEffectiveWithExtra:(NSDictionary*)extra;  
  
/**
```

```
* 激励视频广告已经关闭
*/
- (void)rewardedVideoDidClose;
```

1.4 高级功能

SDK支持自定义转换器，当SDK自带的转换器因版本不匹配或功能不匹配时，可自定义转换器，包括具体的每一类广告（可覆盖SDK已实现的联盟，也可自定义添加新的广告联盟），下面以广点通开屏广告为例：

1. 自定义一个类，使之继承自SDK的SFBaseManager类，具体可参考demo的CustomGdtSplashManager类！
2. 实现加载广告和展示广告的方法

```
// 必须实现请求广告的方法
- (void)loadADWithModel:(SFConfigModelAd_Sources *)model{
//model为服务器返回的配置的广告信息
/*
//广告位ID,1=穿山甲, 2=优量汇, 3=快手, 4=京准通 等等。。。。。
@property (nonatomic, assign) NSInteger adv_id;
//广告主key值
@property (nonatomic, copy) NSString *app_key;
//广告主应用ID
@property (nonatomic, copy) NSString *app_id;
//广告源广告位id
@property (nonatomic, copy) NSString *tagid;
*/

// 下面为示例的请求广点通开屏的广告代码
[GDTSDKConfig registerAppId:model.app_id];
self.gdtSplash = [[GDTSplashAd alloc] initWithPlacementId:model.tagid];
self.gdtSplash.delegate = self;
self.gdtSplash.fetchDelay = self.waitDataDuration;
[self.gdtSplash loadAd];
}
```

```
// 必须实现展示广告的方法
- (void)showSplashAdInWindow:(UIWindow *)window withBottomView:(UIView
*)bottomView{
// 下面为示例的展示广点通开屏的广告代码
[self.gdtSplash showAdInWindow:window withBottomView:bottomView skipView:nil];
}
```

3. 在广点通的回调里面，通过block回调传递函数信息，这样在SFSplashManager遵守代理的情况下，可收到广告的代理回调

```
// 以下为block回调的示例代码(其中type代表的函数名为: //1: 广告加载成功 2: 广告加载失败 3: 广告
点击 4: 从落地页返回 5: 广告关闭 6: 素材成功展示 7: 激励视频获得奖励回调 8: 素材渲染成功,这些
函数不一定全部实现, 具体以对应的广告类的代理方法为主)
self.baseModel.type = blockType;
if (self.successBlock) {
    self.successBlock(self.baseModel);
}
```

4. 在对应的广告类请求广告时（如果是覆盖SDK实现的联盟适配器，则需要将自己对应的自定义转换器类进行注册，如果是新的广告平台，在后台创建后，可以直接正常调用），如下所示：

```
SFSplashManager *manager = [SFSplashManager new];
// 注意：必须在对应的广告类创建之后，并且在广告请求之前进行注册，
// 第一个参数为广告主对应平台的ID(目前对应的是: //广告位ID,1=穿山甲, 2=优量汇, 3=快手, 4=京准通
等等... ), 当添加新的广告主主时, 次步可省略, 在后台配置后, 可直接调用
// 第二个参数为自定义广告转换器的类名
[manager registerADVID:@"2" ClassName:@"CustomGdtSplashManager"];
manager.delegate = self;
manager.mediaId = splash_id;
[manager loadAdData];
self.manager = manager;
```

此时，广告流程将走自定义广告请求类！Done！

对于其他任意广告主，可以在平台自定义创建添加！

附录

错误码

下面是各种ErrorCode的值

SDK 错误码

| | | |
|-----------|--------|--------------|
| ErrorCode | = 406, | // 直客素材渲染失败 |
| ErrorCode | = 405, | // 请检查广告配置 |
| ErrorCode | = 404, | // 网络请求失败 |
| ErrorCode | = 403, | // 请检查广告位ID |
| ErrorCode | = 402, | // 请检查APPID |
| ErrorCode | = 401, | // 解析的数据没有广告 |
| ErrorCode | = 400, | // 请求广告配置为空 |

[穿山甲广告错误码](#)

[优量汇错误码](#)

快手错误码

| code | 说明 |
|--------|-------------------------------|
| 40001 | 没有网络 |
| 40002 | 数据解析失败 |
| 40003 | 广告数据为空 |
| 40004 | 缓存视频资源失 |
| 100001 | 参数有误 |
| 100002 | 服务器错误 |
| 100003 | 不允许的操作 |
| 100004 | 服务不可用 |
| 310001 | appId未注册 |
| 310002 | appId无效 |
| 310003 | appId已封禁 |
| 310004 | packageName与注册的packageName不一致 |
| 310005 | 操作系统与注册的不一致 |
| 320002 | appId对应账号无效 |
| 320003 | appId对应账号已封禁 |
| 330001 | posId未注册 |
| 330002 | posId无效 |
| 330003 | posId已封禁 |
| 330004 | posid与注册的appId信息不一致 |

[京准通错误码](#)

百度错误码

| 错误码 | 解释 |
|---------|---------------------------|
| 0 | 广告返回 |
| 0103010 | 应用ID信息缺失 |
| 0103011 | 应用ID信息错误，百度联盟(百青藤)未收录 |
| 0103012 | 应用ID信息无效，百度联盟(百青藤)上未生效 |
| 0103060 | 应用包名信息错误，请保证注册包名和实际请求包名一致 |
| 0107001 | 广告位ID未收录 |
| 0107002 | 广告位ID未启用 |
| 0107003 | 广告位ID与APPSID不匹配 |
| 1020001 | 网络连接失败 |
| 1040001 | 请求时使用了错误的参数，比如使用错误的广告位ID |
| 1040003 | 请求超时 |
| 3030002 | 缓存物料失败 |
| 3040001 | 广告展现标准不达标 |

FAQ

1. 为什么demo可以运行，接入项目会出错？

答：接入SDK需要很多的配置工作，请按照文档说明配置齐全，保证没有遗漏！

2. 广告位在哪获取？

答：联系广告运营获取。

3. 广告对接成功，但是没有收益是怎么回事？

答：广告收益一般在第二天会在后台系统显示，节假日顺延，如果还是没看到，请确保广告位是否使用正确，如果误用测试广告位，这个是没有广告收益的！

3. iOS集成的包大小是多少？

答：MSaaS主包根据我们demo打包后的计算为0.7MB左右，其他广告主依赖包请参考联盟接入文档，具体大小会根据导入的功能有所差别，实际情况以集成后的包大小为主。