



AppsFlyer Support > SDK集成相关 > 过去SDK版本

# ios - 4.6.3

关注

Print / PDF



Jamie Weider

最近更新时间: 2018年11月20日11:13AM

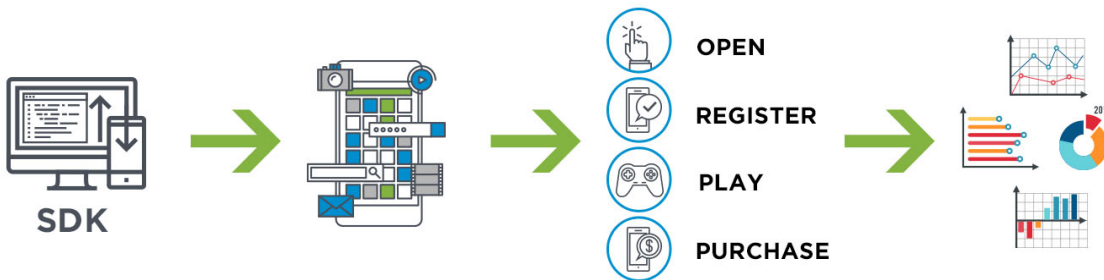
## 页面内容:

注意事项:



### 该版本新增功能 - 当前版本 4.6.3

AppsFlyer的SDK提供应用安装和应用内事件的监控。我们的SDK经过多次迭代，目前已非常稳定（已监控超过80亿次安装），安全，轻量，并且可以很轻松的完成嵌入



你可以监控安装，应用更新和应用打开（需要完成基本嵌入），也可以进一步监控应用内事件（包括购买，游戏过关等），从而可以评估渠道的ROI以及用户粘性等深层次指标

对于Static Lib, iOS SDK可适用于所有操作系统版本在6及以上的iOS设备 (iPhone, iPod, iPad) ; 对于Framework, iOS SDK可适用于所有操作系统版本在8及以上的iOS设备 (iPhone, iPod, iPad)

本指南分为前期准备, 安装追踪 (基础设置), 应用内事件追踪, 以及高级设置四个部分。其中前两个部分为必选项, 完成后即可实现对安装的跟踪和留存的计算。应用内事件追踪为 (强烈建议的) 可选项, 最后是进行付费确认, 自定义用户ID等高级功能。

## 注意事项:

- ✔ AppsFlyer的SDK支持IPv6 DNS64/NAT64 Networks, 详情请[查看此文档](#)。
- ✔ AppsFlyer SDK会使用NSUserDefaults class, 如果清除NSUserDefaults值可能会影响归因追踪。

## 更新须知

- ✔ 添加“SystemConfiguration” framework
- ✔ 现在setUserEmail中支持SHA256加密
- ✔ Bug修复和功能加强

## 前期准备 (必选)

1. 如果该应用使用的是Unity进行最后打包, 请使用AppsFlyer的Unity Plugin进行对接。

[AppsFlyer SDK 对接 - Unity](#)

### 2. SDK下载

- ✔ 如果希望以 **static library** 的方式把SDK添加到项目中, 请[点此下载](#)

☑ 如果希望以 **framework** 的方式把SDK添加到项目中, 请[点此下载](#)

### 3. 将SDK添加到项目中

#### 3.1 以 **Framework** 的方式添加

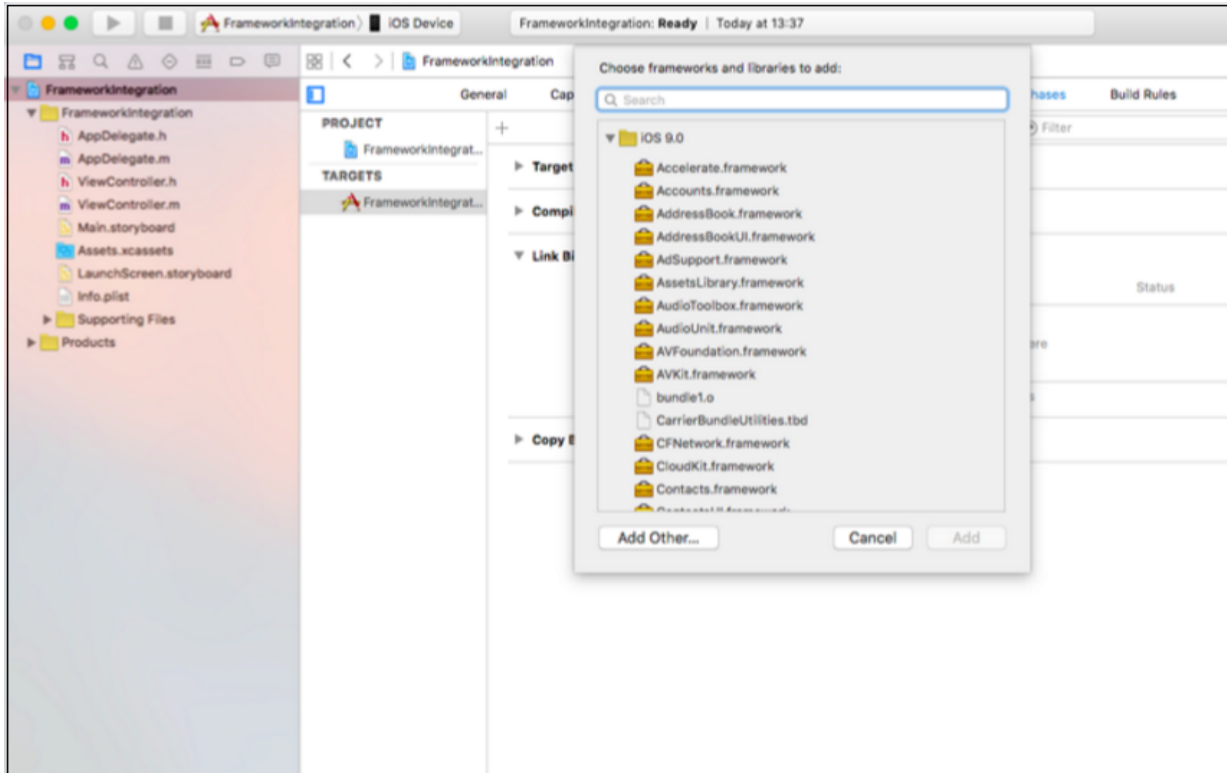
如果你使用 **cocoapods**, 添加AppsFlyer SDK可以简单到只需一行代码:

☑ 请把如下代码加入你的 pod 文件

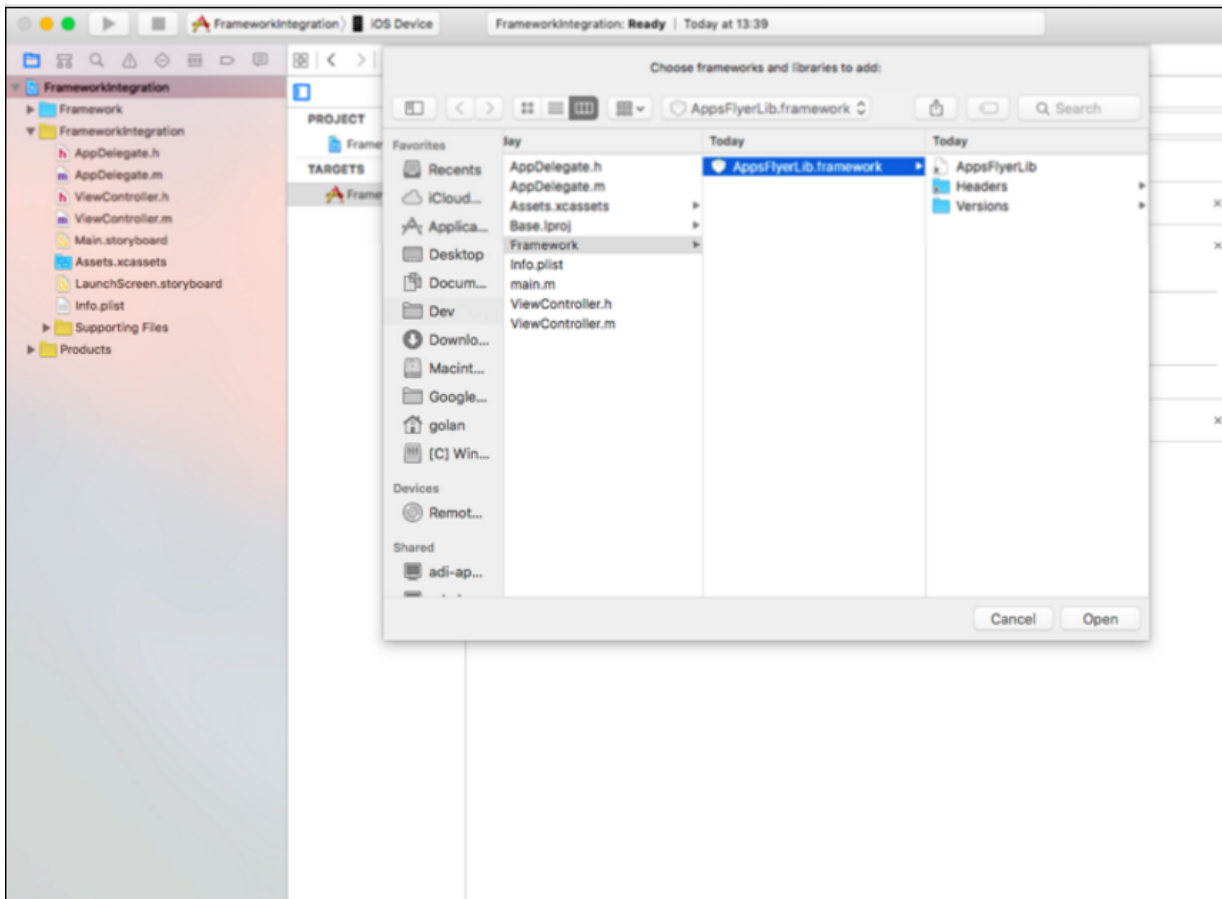
```
pod 'AppsFlyerFramework'
```

如果你不使用cocoapods, 请按如下步骤进行添加:

在Xcode中, 打开 **Build Phases >> Link Binary with Libraries >>**, 点击+ 以添加一个新的 library



☑ 点击 **Add Other**, 再选择 **AppsFlyerLib.framework**



- ✔ 在代码中加入以下头文件
- ✔ `#import <AppsFlyerTracker/AppsFlyerTracker.h>`

### 3.2 以Static Library的方式进行添加

**Static Library 现已弃用Cocoapods。请直接点击上文中提到的链接进行下载。**

请按如下步骤进行添加：

- ✔ 将头文件和lib文件添加到项目中。所需的两个文件（libAppsFlyerLib.a & AppsFlyerTracker.h）可以从[这里下载](#)。
- ✔ 把以下头文件声明添加到工程中
 

```
#import "AppsFlyerTracker.h"
```
- ✔ 将**AdSupport.framework** 添加到工程中，并把它设置成**Optional**。这个在把App提交到商店进行审核的时候会用到。详情请参考[提交审核指南](#)

### 3.3 收集IDFA以及Apple Search Ads

*注意:*

- ✔ AppsFlyer *只有在AdSupport.framework被添加到工程中之后才能收集IDFA*
- ✔ *不添加该framework将导致Facebook、Twitter和其他媒体平台不能正常追踪*
- ✔ 把 **iAd.framework** 添加到工程中。
- ✔ 因其为追踪Apple Search Ads的强制步骤，强烈建议添加该framework

## 安装追踪（必选）

*注意: 这是使用AppsFlyer进行广告效果监控的最基础设置*

### 1. 初始化SDK

请将下面的代码添加到“**didFinishLaunchingWithOptions**”函数中:

```
[AppsFlyerTracker sharedTracker].appsFlyerDevKey = @"REPLACE THIS WITH YOUR Dev_Key";  
[AppsFlyerTracker sharedTracker].appleAppID = @"REPLACE THIS WITH YOUR App_ID";
```

其中, [Dev\_Key]可以在AppsFlyer的后台里找到。请用你的帐号登录到AppsFlyer后台, 转到 **Settings >> Integrate the SDK into...**, 即可找到Dev\_Key。appleAppID处需填写应用的iTunes ID, 但不包含“id”。

比如你的Dev\_Key是**a4kGpVyxm8iGgzvzT8NPaV**, AppID是**id123456789**, 那么上面的代码应该写成:

```
[AppsFlyerTracker sharedTracker].appsFlyerDevKey = @"a4kGpVyxm8iGgzvzT8NPaV";  
[AppsFlyerTracker sharedTracker].appleAppID = @"123456789";
```

### 2. 追踪应用打开

请将下面的代码添加到 **AppDelegate.m** 文件的 “**applicationDidBecomeActive**” 函数中:

```
#import "AppsFlyerTracker.h"  
-(void)applicationDidBecomeActive:(UIApplication *)application  
{
```

```
// Track Installs, updates & sessions(app opens) (You must include this API to en  
[[AppsFlyerTracker sharedTracker] trackAppLaunch];  
}
```

### 3.访客找回追踪

如果您的应用支持deeplinking (Universal links 或者标准deeplinking链接), 或者您打算进行访客找回活动, 请务必完成高级操作中第7部分的操作。

---

**至此, AppsFlyer SDK的最基础嵌入已经完成。你已经可以通过AppsFlyer追踪广告投放的安装和留存的效果。但是对于广告投放优化来说, 仅仅追踪这两点是不够的。我们强烈建议开发者继续往下阅读, 进一步追踪应用内事件。**

---

## 应用内事件追踪 (可选)

应用内事件是安装之后用户与应用交互的行为 (如注册, 登录, 付费等), 对它的追踪可以更好的判断用户质量 (包括反作弊), 从而可以对广告投放进行优化。

AppsFlyer富应用内事件的基本格式是事件名称和事件的值。对其追踪的语法为:

```
(void) trackEvent:(NSString *)eventName withValues:(NSDictionary*)values
```

**eventName**是一个任意的字符串, 代表事件名称。

*\* 事件名称不能超过45个字符, 否则该事件将不会出现在后台, 只会出现在原始数据报告, 以及 Push和Pull API返回的数据中*

**values**是一个包含该事件所有可能值的字典类型 (NSDictionary\*)

\* 事件的值可以为0个, 1个, 或多个

### **富应用内事件收益:** 请使用“af\_revenue” (常数)

AFEventParamRevenue

其可为任何数值, 正负均可。

**注意:** 如果有些收益您不希望记录在控制面板中的Revenue中, 可使用“af\_price”

AFEventParamPrice

不会作为revenue进行记录, 因此不会影响收益的记录以及LTV的监测。

请[点击这里](#)了解AppsFlyer富应用内事件 (iOS) 的详情。下面给出几个富应用内事件的例子

### **示例1: 登陆事件**

```
[[AppsFlyerTracker sharedTracker] trackEvent: AFEventLogin];
```

**注意:** AppsFlyer 也接收此类没有事件值的事件。

### **示例2: 游戏过关事件**

```
[[AppsFlyerTracker sharedTracker] trackEvent: AFEventLevelAchieved withValues:@{
    AFEventParamLevel: @9,
    AFEventParamScore : @100 }];
```

这段代码将创建一个“af\_level\_achieved”事件, 对应事件的值为:

```
{af_level: 9 , af_score: 100}
```

### **示例3: 付费事件**

```
[[AppsFlyerTracker sharedTracker] trackEvent:AFEventPurchase withValues: @{
    AFEventParamContentId:@"1234567",
    AFEventParamContentType : @"category_a",
    AFEventParamRevenue: @200,
    AFEventParamCurrency:@"USD"}];
```

这段代码将创建一个“af\_purchase”事件, 对应的值为:

```
{af_content_id: "1234567", af_content_type: "category_a", af_revenue: 200, af_currency:
"USD"}
```

**注意**，您在任何事件中包含的AFEventParamRevenue (af\_revenue)类型的变量，都会被我们单独统计，并作为Revenue显示在后台和报表中。其他变量类型的值都不会影响Revenue的统计。

## 高级设置（可选）

以下的API属于AppsFlyer的高级设置，请根据需要选用

### 1. 设置货币单位

AppsFlyer默认的货币单位是美元（USD）。以下代码可以设置全局的货币单位，对应的ISO代码可以从[这里找到](#)。

```
[[AppsFlyerTracker sharedTracker].currencyCode = @"GBP"];
```

**注意**，您可以在af\_purchase事件中，通过“af\_currency” (AFEventParamCurrency)，单独为该事件的revenue指定货币单位

### 2. 获取AppsFlyer ID

AppsFlyer ID是AppsFlyer对于某个app的新用户赋予的唯一ID，它被用于在系统内部标识某个app的独立用户，出现在所有原始数据报表和API中。你可以通过以下的代码获取该ID：

```
(NSString *) [AppsFlyerTracker sharedTracker] getAppFlyerUID
```

### 3. 设置广告主的用户ID

如果你有自己的用户系统，把你们的用户ID发送给AppsFlyer，可以轻松实现跨应用，跨平台的用户追踪，以及把AppsFlyer的广告投放效果整合到广告主的BI系统中。设置广告主用户ID的代码如下：

```
[[AppsFlyerTracker sharedTracker].customerUserID = @"YOUR_CUSTOM_DEVICE_ID"];
```

**注意：**

 用户ID必须在trackAppLaunch前被调用



- ✔ 只有在这个API被设置之后，后续的应用内事件才会带上广告主用户ID。所以建议客户在应用启动时就开始设置/获取广告主用户ID，并且在获得该用户ID之后马上通过上述的代码告诉我们
- ✔ 如果想通过AppsFlyer的数据回传来使用第三方数据分析平台如Mixpanel和Swrve的服务，则广告用户ID的设置是必需的

更多关于广告主用户ID的详情，请参考[这个文档](#)。

#### 4. 获取归因数据

你可以在客户端第一时间获取AppsFlyer的归因数据，从而可以实现延迟深度链接的功能。具体细节请参考[这个文档](#)。

注意：该功能相关函数会在每次应用打开的时候调用。

#### 5. 开启对应用扩展 (App Extension) 的支持

首先，你需要添加访问互联网的权限

- ✔ 打开App Extension的 **info.plist** 文件
- ✔ 在NSExtension / NSExtensionAttributes中，把 RequestsOpenAccess 标记设置成YES

然后，在App Extension 的 ViewController 的 ViewDidLoad函数中初始化 AppsFlyerLib：

```
[AppsFlyerTracker sharedTracker].appsFlyerDevKey = @"YOUR_DEV_KEY_HERE";  
[AppsFlyerTracker sharedTracker].appleAppID = @"APP_ID_HERE";  
[AppsFlyerTracker sharedTracker].delegate = self;  
[[AppsFlyerTracker sharedTracker] trackAppLaunch];
```

接下来是获取归因数据：

- ✔ 打开App Extension 的 ViewController.h
- ✔ 导入AppsFlyer头文件

```
#import "AppsFlyerTracker.h"
```

- ✔ 在接口声明的地方添加<AppsFlyerTrackerDelegate>

最后，在Extension 的 ViewController.m文件中，添加如下代码：

```

-(void)onConversionDataReceived:(NSDictionary*) installData {
    id status = [installData objectForKey:@"af_status"];
    if([status isEqualToString:@"Non-organic"]) {
        id sourceID = [installData objectForKey:@"media_source"];
        id campaign = [installData objectForKey:@"campaign"];
        NSLog(@"This is a none organic install. Media source: %@ Campaign: %@", source
    } else if([status isEqualToString:@"Organic"]) {
        NSLog(@"This is an organic install.");
    }
}
}
-(void)onConversionDataRequestFailure:(NSError *) error {
    NSLog(@"%@", error);
}
}
-(void) onAppOpenAttribution:(NSDictionary*) attributionData {
    NSLog(@"attribution data: %@", attributionData);
}
}
-(void) onAppOpenAttributionFailure:(NSError *)error {
    NSLog(@"%@", error);
}
}

```

## 6. 设置用户邮箱

研发可手机用户邮箱地址信息，然后按照希望的加密方式传送给AppsFlyer。我们支持明文以及 Sha1、Sha256和MD5加密方式。例如：

```

[[AppsFlyerTracker sharedTracker] setUserEmails:@[@"email1@domain.com",
@"email2@domain.com"] withCryptType:EmailCryptTypeSHA1];

```

注意：个人验证信息（PII）如邮箱地址是不会被AppsFlyer进行保存，该信息将不会在任何报告中显示收集的该信息仅用于向媒体平台回传。

## 7. 用户访客找回归因（上报深度链接）

Deeplinking为访客找回活动追踪中很重要的一部分，如果要开展访客找回活动，AppsFlyer强烈建议使用该方法。

iOS 9及以上版本，deeplinking通过Universal Links实现，因此必须在您的应用中整合Universal Links。

点击[此处](#)查看AppsFlyer Universal Links整合指南。

追踪该类开启，请添加以下代码至app delegate:

```
// Reports app open from a Universal Link for iOS 9
- (BOOL) application:(UIApplication *)application continueUserActivity:(NSUserActivity *)
{
[[AppsFlyerTracker sharedTracker] continueUserActivity:userActivity restorationHandler:re
return YES;
}

// Reports app open from deeplink for iOS 8 or below (DEPRECATED)
- (BOOL) application:(UIApplication *)application openURL:(NSURL *)url
    sourceApplication:(NSString*)sourceApplication annotation:(id)annotation
{

[[AppsFlyerTracker sharedTracker] handleOpenURL:url sourceApplication:sourceApplication w
return YES;
}

// Reports app open from deeplink for iOS 10
- (BOOL) application:(UIApplication *)application openURL:(NSURL *)url
    options:(NSDictionary *) options
{

[[AppsFlyerTracker sharedTracker] handleOpenURL:url options:options];
return YES;
}
```

更多细节请参考[OneLink设置指南](#)。

## 8. 付费验证

*注意：该功能需要OS7及以上的支持*

AppsFlyer支持和苹果商店服务器的付费验证。你只需在SKStoreKit的 **completeTransaction** 回调函数中调用AppsFlyer SDK的 **validateAndTrackInAppPurchase** 方法。它会自动生成一个“af\_purchase”事件，所以你无需再创建一个单独的“af\_purchase”事件

```

- (void) validateAndTrackInAppPurchase:(NSString *) productIdentifier
price:(NSString *) price
currency:(NSString *) currency
transactionId:(NSString *) transactionId
additionalParameters:(NSDictionary *) params
success:(void (^)(NSDictionary *response)) successBlock
failure:(void (^)(NSError *error, id response)) failedBlock;

```

这个方法有两个回调，一个是成功，一个是失败（各种原因导致的失败，包括验证失败）。如果成功，则会从苹果商店服务器返回包含收据信息的字典变量

注意：测试该方法时，我们建议把 **useReceiptValidationSandbox** 标记设置成YES，这样测试数据就不会进入Apple正式服务器。

```
[AppsFlyerTracker sharedTracker].useReceiptValidationSandbox = YES;
```

示例代码如下：

```

[[AppsFlyerTracker sharedTracker] validateAndTrackInAppPurchase:product.productIdentifier
currency:@"USD"
transactionId:trans.transactionIdentifier
additionalParameters:@{@"test": @"val" , @"test1" : @"val 1"}
success:^(NSDictionary *result) {
    NSLog(@"Purchashe succeeded And verified!!! response: %@", res
} failure:^(NSError *error, id response) {
    NSLog(@"response = %@", response);
}];

```

## 9. 终端用户Opt-Out

AppsFlyer允许你停止对某些用户的追踪。这个功能符合公司最新的隐私政策，以及Facebook的数据隐私政策。默认设置为NO，意味着默认所有的用户都将被追踪。如果想取消对某个用户的追踪，

请在SDK初始化的时候添加如下代码：

```
[AppsFlyerTracker sharedTracker].deviceTrackingDisabled = YES;
```

## 10. 停止IDFA的收集 (不推荐)

只有在 AdSupport.framework 被添加之后，AppsFlyer SDK才会开始收集IDFA。所以一般不需要特别指定打开或关闭IDFA的收集。但如果你想明确的停止对IDFA的收集，请在SDK初始化的时候调用如下的代码：

```
[AppsFlyerTracker sharedTracker].disableAppleAdSupportTracking = YES;
```

当**disableAppleAdSupport** 追踪设置为YES时，IDFA将不会被发送到AppsFlyer的服务器上。

## 11. 卸载监测

您需要首先使用下面的方法注册使用卸载的功能。

需要调用此方法：

```
- (void)application:(UIApplication *)application didRegisterForRemoteNotificationsWithD
```

以下是启用该功能的代码片段：

```
// Register for Push Notifications
```

```
UIUserNotificationType userNotificationTypes = (UIUserNotificationTypeAlert |
                                                UIUserNotifi
                                                UIUserNotifi
UIUserNotificationSettings *settings = [UIUserNotificationSettings settingsForTypes:

[application registerUserNotificationSettings:settings];
[application registerForRemoteNotifications];

[application setMinimumBackgroundFetchInterval:UIApplicationBackgroundFetchIntervalM
application.applicationIconBadgeNumber = 0;
```

```
- (void)application:(UIApplication *)application didRegisterForRemoteNotificationsWithD
    [[AppsFlyerTracker sharedTracker] registerUninstall:deviceToken];
}
```

另外接入详情，也请参考[此文档](#)。

## 示例

```
[[AppsFlyerTracker sharedTracker] registerUninstall:deviceToken];
```

当在Apple Development环境测试卸载时，请保持该flog的默认设置为NO:

```
[AppsFlyerTracker sharedTracker].useUninstallSandbox = NO;
```

当前我们只支持在TestFlight和Production进行卸载测试。

为正确完整地完成该步骤，请务必阅读[此处](#)。

## 12. 通过OneLink定制应用内深度跳转

### 通过OneLink定制应用内深度跳转

OneLink原来已支持深度链接，以直达在OneLink内af\_dp参数定义或者Universal Link的应用内特定页面

而此处提到的新功能，您需要接入 **onAppOpenAttribution** 回调。您可利用此回调获取OneLink中的参数信息，以自定义用户应用内深度跳转。

```
(void) onAppOpenAttribution:(NSDictionary*) attributionData; ///iOS
```

详情，请参考[此文档](#)。

## 13. 消息推送监测

AppsFlyer可以将消息推送作为广告活动一部分进行监测。

请添加以下代码至您的应用AppDelegate中开启消息推送追踪功能。

```
-(void) application:(UIApplication *)application didReceiveRemoteNotification:(NSDictionary *)userInfo {  
    [[AppsFlyerTracker sharedTracker] handlePushNotification:userInfo];  
}
```

推送消息需含有 "af" 参数及AppsFlyer追踪参数。

示例

```
{  
    "aps": {  
        "alert": "Push text",  
        "sound": "default",  
        "category": "REMINDER_CATEGORY"  
    },  
    "_p": 123456,  
    "payloadKey": "payloadValue"  
    "af": {  
        "pid": "swrve_int",  
        "is_retargeting": true,  
        "c": "test_campaign"  
    }  
}
```

## SDK嵌入测试

1. 在应用提交到苹果商店之前的测试, 请[参考此文档](#)
2. 在应用提交到苹果商店之后的测试, 请[参考此文档](#)

## 评论

0 条评论

成为第一个写评论的人。

提交