

百度

百度移动统计 SDK

开发者手册 IOS-3.4 版本

MTJ

2015/3/12

修改记录

版本号	变更内容
3.4	(1)去除了不支持 ARM64 的包，只留下支持 ARM64 的一个包。 (2)去除对 TOUCHJSON 的引用，使用 iOS 自带的 JSON 解析函数。 (3)支持版本改为 iOS 5.0+。 (4)新增对内嵌 WEBVIEW 页面的统计。 (5)DEMO 程序中新增 WEBVIEW 统计相关文件。 (6)修正部分 BUG。
3.3	(1)修正 DEMO 因静态库引用路径造成的编译错误问题 (2)修正部分注释文字 (3)错误日志中添加版本信息 (4)新增支持 ARM64- 64 位编译的库
3.22	(1)去除广告标识符的引用 (2)开发者可以根据需求来设置广告标识符
3.2	(1)优化错误信息采集部分的代码，采集错误更细化 (2)增加调试功能接口 (3)修复单页面启动时页面丢失的问题 (4)增加广告的依赖库
3.1	(1)增加统计事件发生时间 (2) 增加自动生成序列 CUID 自定义设备标示符 (3) 增加统一的 SDK 库，模拟器和真实设备共用一个库 (4) 优化升级本地缓存自定义事件部分日志的合并 (5) 增加 SHORTVERSION 接口，设置应用版本号接口 (6) 优化启动时发送策略，使发送更及时
3.0	(1) 增加统计事件发生时长 (2) 设置应用进入后台再回到前台为同一次 SESSION 的间隔时间的方法
2.1	(1) 更新 TOUCHJSON 库到最新版本 (2) 支持 IOS6 SDK，XCode4.5 开发环境 (3) 解决与友盟库同时添加的符号冲突
2.0	(1) 增加日志发送策略 (2) 增加应用版本统计 (3) 废弃了启动 GPS 的接口 (防止开发者在非 GPS 应用中启用该选项而导致被苹果拒绝上架)

目录

前言	4
1. 关于调试(注意事项)	4
2. 关于改动(注意事项)	4
第一章 简介	5
第二章 阅读对象	5
第三章 版本支持	5
第四章 集成使用	5
第一节 添加 SDK 到项目	5
第二节 参数申请	6
第五章 代码集成	6
第一节 启动功能	6
第二节 页面统计	8
第三节 渠道统计	8
第四节 应用版本统计	9
第五节 设置应用版本号（3.1 新增）	9
第六节 日志发送策略	9
第七节 调试模式（3.2 新增）	10
第八节 广告标识符设置（3.22 新增）	10
第九节 内嵌 WebView 页面统计（3.4 新增）	11
第六章 包的目录结构	11
第七章 联系我们	12

前言

1. 关于调试(注意事项)

- (1) **保证客户端时间的准确性。**因为我们 `mtj.baidu.com` 网站上面的统计数据是根据客户端时间来统计的，所以如果您需要测试统计数据，那么请保证您的设备的时间是正确的，如果不正确，有可能会造成统计数据丢失或者延迟展现。
- (2) **保证自定义事件的 ID 已经注册。**如果您需要使用自定义事件，那么需要在 `mtj.baidu.com` 网站上自定义事件中定义您需要的自定义事件的 ID，然后按照自定义事件的规范写入代码。
- (3) **保证您已经替换了 AppKey。**Appkey 需要在 `mtj.baidu.com` 网站上注册应用，那么就会生成一个 Appkey，该 key 是该应用的唯一标识。如果没有该 key，那么，统计数据将不会展现。
- (4) 如果不知道怎么接入代码，请参考 `mtj.baidu.com` 网站上下载的 SDK 包中的 demo 实例程序。
- (5) 可以选择设置相应的发送策略。具体参见第五章第六节说明。
- (6) **如果没有数据怎么办？**请确认以上五点都已经确认，网站统计数据会在 15 分钟左右展示出来，如果没有数据有可能是您的时区设置或者时间出现暂时异常，可以选择换台设备来测试，或者联系我们，具体参见第七章。
- (7) **关于渠道统计结果为 appstore，**说明您设置渠道在某种情况下丢失。请保证用户可能进入的页面已经设置了渠道，否则如果出现小型的崩溃或者内存不足时有可能导致起初设置的参数丢失。
- (8) **SDK 调试相关。**为了方便调试 3.2 版本新增了 `enableDebug` 接口来方便开发者通过 Log 来调试 SDK。
- (9) **怎样快速的发送统计日志。**可以设置 `session` 失效时间为 1S（默认为 30S，一般设置在 1 到 600S 之间）。这样每次启动间隔 1S 以上都会发送日志（前提是设置启动时发送而不是定时发送）。
- (10) **怎样查看发送日志。**打开调试开关，每次发送时在 `xcode` 工具中都会有 `sdk` 的 Log 发出。

2. 关于改动(注意事项)

- (1) 3.3 版本新增了支持 `arm64-64` 位编译的库，可选择使用。详情可阅读第六章内容。
- (2) 3.4 版本去除了不支持 `arm64` 框架的包，只留下一个支持 `arm64` 框架的包。

第一章简介

百度移动统计 SDK(iOS)是百度为 iOS 平台提供的应用统计工具(以下简称 SDK)。该文档提供了对如何使用 SDK 的一个详细说明。建议阅读时下载我们的 API 用例,按照用例 设置自己的工程。SDK 下载地址:

<http://mtj.baidu.com/web/welcome/sdk>, 包的详细介绍见第六章。

如有其他问题可以参考网站的 FAQ, 或者与我们联系: (apptongji@baidu.com)

第二章阅读对象

本文档面向所有使用该 SDK 的开发人员、测试人员。

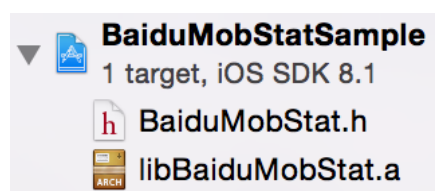
第三章版本支持

IOS5.0+

第四章集成使用

第一节 添加 SDK 到项目










- 添加 sdk 静态库和.h头文件



- 添加系统依赖库

需要添加的系统依赖库如下:

▼ Linked Frameworks and Libraries

Name	Status
 libBaiduMobStat.a	Required ⬆️⬆️
 Security.framework	Required ⬆️⬆️
 CoreLocation.framework	Required ⬆️⬆️
 SystemConfiguration.framework	Required ⬆️⬆️
 libz.1.2.5.dylib	Required ⬆️⬆️
 libstdc++.dylib	Required ⬆️⬆️
 CoreTelephony.framework	Required ⬆️⬆️
 CoreGraphics.framework	Required ⬆️⬆️
 UIKit.framework	Required ⬆️⬆️
 Foundation.framework	Required ⬆️⬆️

第二节 参数申请

在百度移动统计平台(<http://mtj.baidu.com>) 申请应用 ID (APP KEY) 用于标识您的应用程序。

在百度移动统计平台(<http://mtj.baidu.com>)的应用配置功能中创建 EventId 来定义您的自定义事件。

第五章代码集成

第一节 启动功能

注意： 2.0 中移除了 enableLocation 接口。

➤启动代码。在应用启动函数(didFinishLaunchingWithOptions)中调用如下代码即可完成启动功能；启动的过程中同时您可以配置是否打开崩溃日志收集 (statTracker.enableExceptionLog)和是否启用gps(statTracker.enableLocation)信息。

```
BaiduMobStat* statTracker = [BaiduMobStat defaultStat];
statTracker.enableExceptionLog = YES; |
statTracker.channelId = @"ReplaceMeWithYourChannel";
statTracker.logStrategy = BaiduMobStatLogStrategyCustom;
statTracker.logSendInterval = 1;
statTracker.logSendWifiOnly = YES;
statTracker.sessionResumeInterval = 10;
statTracker.shortAppVersion = IosAppVersion;
statTracker.enableDebugOn = YES;
/*如果有需要，可自行传入adid
NSString *adId = @"";
if([[[[UIDevice currentDevice] systemVersion] floatValue] >= 6.0f]){
adId = [[[ASIdentifierManager sharedManager] advertisingIdentifier] UUIDString];
}
statTracker.adid = adId;
*/
[statTracker startWithAppId:@"ReplaceMeWithAppKey"]; //设置您在mtj网站上添加的app的appkey
```

➤ 一次启动间隔时间设定。3.0 中新增设置应用进入后台再回到前台为同一次 session 的间隔时间的方法：

sessionResumeInterva: 设置范围为[0~600s],超过600s 则设为600s, 默认为30s

➤ 发送策略设置。statTracker.logStrategy 可以有三种发送策略设置，具体见日志发送策略一节，上图可视。

➤ 事件统计。下面的是事件统计部分代码：（注意: eventId 需要预先在服务器端配，eventLabel 不可以为空）

```
- (void)tabBarController:(UITabBarController *)tabBarController didSelectViewController:(UIViewController *)viewController
{
    int index = tabBarController.selectedIndex;
    BaiduMobStat* statTracker = [BaiduMobStat defaultStat];
    [statTracker logEvent:@"TabClick" eventLabel:[NSString stringWithFormat: @"Tab-%d", index]];
}
```

➤ 自定义事件时长。3.0 中新增的自定义事件的方法（详见.h 头文件）。用来统计自定义事件的持续时长：

```
/**
 * v3.0 新增
 * 记录一次事件的时长，eventId请在网站上创建。未创建的eventId记录将无效。eventId与eventLabel必须是有内容的字符串，不可为nil或者空字符串。
 */
- (void) logEventWithDurationTime:(NSString*) eventId eventLabel:(NSString*)eventLabel durationTime: (unsigned long)duration;
/**
 * v3.0 新增
 * 记录一次事件的开始，eventId请在网站上创建。未创建的eventId记录将无效。eventId与eventLabel必须是有内容的字符串，不可为nil或者空字符串。
 */
- (void) eventStart:(NSString*) eventId eventLabel:(NSString*)eventLabel;
/**
 * v3.0 新增
 * 记录一次事件的结束，eventId请在网站上创建。未创建的eventId记录将无效。eventId与eventLabel必须是有内容的字符串，不可为nil或者空字符串。
 */
- (void) eventEnd:(NSString*) eventId eventLabel:(NSString*)eventLabel;
```

事件时长统计新增两种方法：

1. 在事件开始的时候调用 eventStart 方法，在事件结束时调用 eventEnd 方法。Sdk 自动计算事件发生时长。
2. 调用 logEventWithDurationTime 方法，并将事件时长作为第三个参数 duration 传入。

第二节 页面统计

```
#pragma mark - View lifecycle

-(void) viewWillAppear:(BOOL)animated
{
    NSString* cName = [NSString stringWithFormat:@"%s", self.tabBarItem.title, nil];
    [[BaiduMobStat defaultStat] pageviewStartWithName:cName];
}

-(void) viewWillDisappear:(BOOL)animated
{
    NSString* cName = [NSString stringWithFormat:@"%s", self.tabBarItem.title, nil];
    [[BaiduMobStat defaultStat] pageviewEndWithName:cName];
}
```

我们建议您在 UIViewController 的 viewWillAppear 函数中调用

pageviewStartWithName:xxx。

在 viewWillDisappear 函数中调用

pageviewEndWithName:xxx。

注意：

Pageview 的名字为开发者自定义。pageName (XXX) 不能为空。

您也可以在程序的其他地方构造 pageview 的 start 和 end,只要在逻辑上构成了一段时间的页面访问即可。

第三节 渠道统计

如下图，您可以在startWithAppId 之前调用statTracker.channelId = @"你的渠道名”，设置不同的渠道名，然后编译成不同的ipa 文件发布。

如果channelId 属性未设置，系统默认会采用AppStore 为您的应用渠道。

```
BaiduMobStat* statTracker = [BaiduMobStat defaultStat];
statTracker.enableExceptionLog = YES; |
statTracker.channelId = @"ReplaceMeWithYourChannel";
statTracker.logStrategy = BaiduMobStatLogStrategyCustom;
statTracker.logSendInterval = 1;
statTracker.logSendWifiOnly = YES;
statTracker.sessionResumeInterval = 10;
statTracker.shortAppVersion = IosAppVersion;
statTracker.enableDebugOn = YES;
/*如果有需要，可自行传入adid
NSString *adId = @"";
if([[[UIDevice currentDevice] systemVersion] floatValue] >= 6.0f){
    adId = [[[ASIdentifierManager sharedManager] advertisingIdentifier] UUIDString];
}
statTracker.adid = adId;
*/
[statTracker startWithAppId:@"ReplaceMeWithAppKey"]; //设置您在mtj网站上添加的app的appkey
```


第四节 应用版本统计

如下图，您可以在应用配置文件xxx.plist 中设置Bundle version， SDK 会默认读取该值作为您应用的版本号。如果你想设置应用的版本号请阅读下节。

Bundle creator OS Type code	String	????
Bundle version	String	1.1
Application requires iPhone environm	Boolean	YES

第五节 设置应用版本号（3.1 新增）

开发者可以调用接口来设置 app 的版本号，该版本号由开发者获取后传入该函数。当然也可以不设置，那么系统将会读取@CFBundleVersion

原因：Xcode4 有两个版本号，一个是 Version,另一个是 Build,对应于 Info.plist 的字段名分别为 CFBundleShortVersionString 和 CFBundleVersion。

为了兼容 Xcode3 的工程，默认取的是 Build 号，如果需要取 Xcode4 的 Version，可以使用下面的方法详见跟随 sdk 的 demo 实例程序。

使用方法：

```
NSString *version = [[[NSBundle mainBundle] infoDictionary]
                    objectForKey:@"CFBundleShortVersionString"];
statTracker.shortAppVersion = version;
```

该方法的调用在设置 appkey 之前，例如：

```
BaiduMobStat* statTracker = [BaiduMobStat defaultStat];
statTracker.enableExceptionLog = YES; |
statTracker.channelId = @"ReplaceMeWithYourChannel";
statTracker.logStrategy = BaiduMobStatLogStrategyCustom;
statTracker.logSendInterval = 1;
statTracker.logSendWifiOnly = YES;
statTracker.sessionResumeInterval = 10;
statTracker.shortAppVersion = IosAppVersion;
statTracker.enableDebugUn = YES;
/*如果有需要，可自行传入adid
NSString *adId = @"";
if([[[[UIDevice currentDevice] systemVersion] floatValue] >= 6.0f]){
    adId = [[[ASIdentifierManager sharedManager] advertisingIdentifier] UUIDString];
}
statTracker.adid = adId;
*/
[statTracker startWithAppId:@"ReplaceMeWithAppKey"]; //设置您在mtj网站上添加的app的appkey
```

第六节 日志发送策略

- 启动时发送： 每次启动并联网时会将之前保存在本地的日志发送本次启动产生的日志将在下一次启动并联网时发送。

BaiduMobStatLogStrategy = BaiduMobStatLogStrategyAppLaunch

- 每日发送： 设置每日单次发送后，启动时发送数据时会做一次检查，距离上次成功发送数据的时间间隔是否超过24 小时，如果超过24 小

时,则将之前保存在本地的日志全部发送。如果还不到 24 小时,则不发送。

BaiduMobStatLogStrategy = BaiduMobStatLogStrategyAppDay;

- 自定义发送间隔:

开发者可以自定义发送间隔(单位为小时,1-24 都可以设置)如果开发者设为N 小时,启动时发送数据时会做一次检查,距离上次成功发送数据的时间间隔是否超过N 小时,如果超过N 小时,则将之前保存在本地的日志全部发送。如果还不到 N 小时,则不发送。

BaiduMobStatLogStrategy = BaiduMobStatLogStrategyCustom;

logSentInterval = N;

- 仅在WIFI 发送: 对上述所有发送间隔设置均有效
是: 必须在wifi 联网方式下才能发送数据
否: 不论联网方式都会发送数据。

logSendWifiOnly = YES/NO;

第七节 调试模式 (3.2 新增)

为了方便开发者进行调试,自 3.2 版本起增加了 SDK 的调试接口,通过以下代码可以打开调试。运行时会有 SDK 的 Log 打印。

```
BaiduMobStat* statTracker = [BaiduMobStat defaultStat];  
statTracker.enableDebugOn = YES;
```

关闭调试,可以去除该代码或者设置为 NO。

嵌入位置如图:

```
BaiduMobStat* statTracker = [BaiduMobStat defaultStat];  
statTracker.enableExceptionLog = YES; |  
statTracker.channelId = @"ReplaceMeWithYourChannel";  
statTracker.logStrategy = BaiduMobStatLogStrategyCustom;  
statTracker.logSendInterval = 1;  
statTracker.logSendWifiOnly = YES;  
statTracker.sessionResumeInterval = 10;  
statTracker.shortAppVersion = IosAppVersion;  
statTracker.enableDebugOn = YES;  
/*如果有需要,可自行传入adid  
NSString *adId = @"";  
if([[[[UIDevice currentDevice] systemVersion] floatValue] >= 6.0f]){  
    adId = [[[ASIdentifierManager sharedManager] advertisingIdentifier] UUIDString];  
}  
statTracker.adid = adId;  
*/  
[statTracker startWithAppId:@"ReplaceMeWithAppKey");//设置您在mtj网站上添加的app的appkey
```

第八节 广告标识符设置 (3.22 新增)

由于苹果禁用了没有使用广告业务的 app 使用广告标识符,所以 3.22 版本的 SDK 去除了该广告标识符的使用,开发者如果使用到了广告,可以自己设置该字段,可以提高统计精度。使用方法如下,可以参照 demo。获取 adid 需要引用系统依赖库 AdSupport.framework。

```

/*如果有需要，可自行传入adid
NSString *adId = @"";
if([[UIDevice currentDevice] systemVersion] floatValue] >= 6.0f){
    adId = [[[ASIdentifierManager sharedManager] advertisingIdentifier] UUIDString];
}
statTracker.adid = adId;
*/

```

第九节 内嵌 WebView 页面统计（3.4 新增）

如果您的页面中使用了 WebView 嵌入了 Html、JS 代码，并且希望统计 html 页面的页面访问与自定义事件，可以通过一下两个步骤实现：

1.实现 WebView 的代理方法，并再代理方法中调用 SDK 的

webViewStartLoadWithRequest:接口，传入 request 参数。

如下所示，详情见 Demo：

```

//实现WebView的代理方法，并在此函数中调用SDK的webViewStartLoadWithRequest:传入request参数，进行统计
- (BOOL)webView:(UIWebView *)webView shouldStartLoadWithRequest:(NSURLRequest *)request
    navigationType:(UIWebViewNavigationType)navigationType
{
    [[BaiduMobStat defaultStat] webViewStartLoadWithRequest:request];
    return YES;
}

```

2.在 WebView 的 Html、JS 代码中，添加对应的接口，并在合适的时候调用，从而使 SDK 获取到该行为，进一步调用 native 代码，实现统计。具体的 Html、JS 对应代码，详见 Demo 程序中的 mobstat.html、mobstat.js 两个文件。

第六章包的目录结构

从 <http://mtj.baidu.com/web/welcome/sdk> 下载下来的 zip 包中解压会有以下目录结构：



- ios_manual.pdf 文件即为本文件，是详细的说明文档。
- ios_api 目录： 此目录是要开发者包含到工程目录的文件，
 - a) 其中Release-iphoneos中的两个文件是给真机使用的，需要导入到工程中；
 - b) Release-iphonesimulator目录是给模拟器使用的，需要导入到工程中；
 - c) Universal-bin目录是前两个目录的.a文件的合体，如果开发者怕麻烦，可以直接导入此文件夹下的.a文件，这样就不用在模拟器和真机设备之间切换导入不同的.a文件了。
- ios_demo 目录是集成了统计里面的所有功能的一个实例，用户可以在其中参考使用统计的各种功能。
- ios_doc 目录是生成的类目录说明，用户可以直接查看类的用法

第七章联系我们

感谢您的阅读,如果有问题请 email 我们。 邮箱: apptongji@baidu.com