GAME ANALYTICS

TalkingData 用数据说话

iOS 接入指南 3.1.x

最后修正：2015-1-23

目录

一、 综述

1. 适用范围

TalkingData Game Analytics 帮助游戏开发者解决玩家数据收集至数据标准化分析的全部繁琐问题，以行业标准指标形式将数据展现于报表中。

SDK 适用于 ARMv7、ARMv7s、ARM64（iOS 5.1.1 及以上操作系统）的设备。
注：使用 Unity 3D 开发的游戏请下载专用 SDK，并使用 Unity 专用开发指南进行集成。

2. 统计标准

为了与游戏自有体系更好的结合，统计中我们采用游戏自身的帐户来做为一个玩家单元。数据中除了玩家基础信息外，主要帮助处理游戏过程中的升级、任务、付费、消费等详细行为数据。行为标准可参考以下说明：

- 玩家
  游戏自身的帐户，通常为一个唯一号、唯一名或一份唯一存档号。是平台中计算数据的最基础单元。
- 设备
  指一台安装了游戏包的终端。
- 玩家的一次游戏
  玩家从打开游戏界面至离开游戏界面的完整过程，如果玩家在离开游戏界面后 30 秒内重新回到游戏中，将被认为是上次游戏被打扰后的延续，记为一次完整游戏。

- 付费
  特指玩家充值现金换取虚拟币的过程。充值的现金将作为收入进行统计。
- 消费
  指玩家在一个消费点上消耗虚拟币的过程。

二、 接入流程

STEP 1、为游戏申请 APP ID

进入 https://www.talkingdata.com/product-game.jsp?languagetype=zh_cn 网站，使用您的注册账号登录后，请预先创建一款游戏，您将获得一串 32 位的 16 进制 APP ID，用于唯一标识您的一款游戏。

STEP 2、向工程中导入追踪 SDK

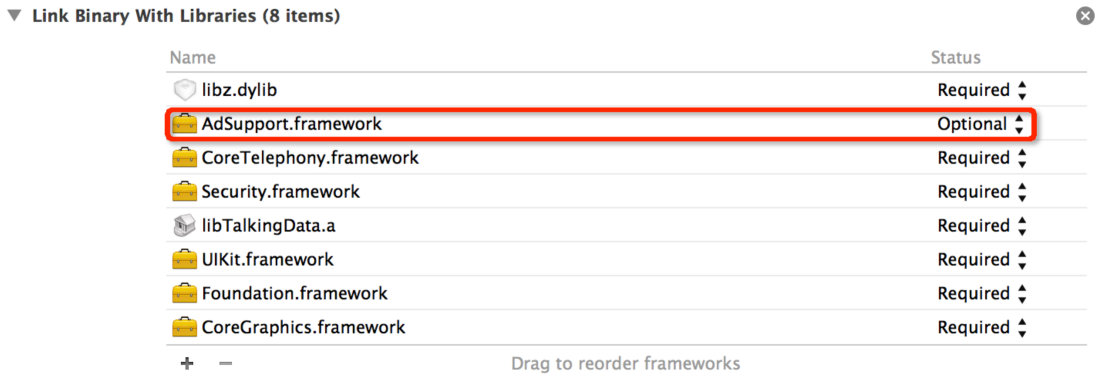
下载游戏统计 SDK 压缩包(点击下载 SDK 包)并解压至本地目录 ,将其中的.h 和.a 导入到您的应用工程中。
在您的工程里→选择 File --> Add Files to “Your Project” --> 选择 TalkingDataGA.h 和 libTalkingDataGA.a 两个文件勾选 Copy items into destination group’ s folder (if needed) 并确保所有要用到 SDK 的 targets 都处于选中状态。

STEP 3、添加依赖的框架

Game Analytics 需要使用 CoreTelephony.framework 框架获取运营商标识，使用 AdSupport.framework 获取 advertisingIdentifier ,使用 Security. framework 框架辅助存储设备标识，使用 libz.dylib 进行数据压缩。
Xcode 的添加方式如下所示：

xcode 版本	操作
Xcode6.0 以上	在您的工程里，选择 target-->Build Phases-->Link Binary With Libraries，点击 + 号，选择 CoreTelephony.framework、AdSupport.framework、Security.framework、和 libz.dylib

如要支持iOS6.0以下版本系统，需要将AdSupport.framework框架的依赖条件改为 Optional，如下图：



STEP 4、添加调用方法

参考“三、添加调用方法“的指导来完成开发。

STEP 5、进行数据测试

调用方法添加完毕后，应当对游戏打包并进行数据测试，以确保打包的正确。

三、 添加调用方法

1. 游戏启动和关闭

用途和用法

- 用于准确跟踪玩家的游戏次数，游戏时长等信息。
- 在需要调用 SDK 方法的文件中导入 TalkingDataGA.h。
  并在application:didFinishLaunchingWithOptions:方法里面调用onStart法进行初始化。
- 可以选择性的在 channelId 中填入推广渠道的名称 ,数据报表中则可单独查询到他们的数据。每台设备仅记录首次安装激活的渠道，更替渠道包安装不会重复计量。不同的渠道 ID 包请重新编译打包。

特别说明：SDK 将自动侦测设备的“锁屏、切换程序、Home 键”等操作，来确定游戏被玩家关闭，无需您做其他调用。

接口说明：（TalkingDataGA 类）

```
//游戏启动
+ (void)onStart:(NSString *)appId withChannelId:(NSString *)channelId;
```

参数说明：

参数	类型	描述
appId	NSString	填写创建游戏时获得的 App ID，用于唯一识别您的游戏。
channelId	NSString	用来标注游戏推广渠道，区分玩家的来源来查看统计。 格式：32 个字符内，支持中文、英文、数字、空格、英文点和下划线，请勿使用其他符号。

示例代码：

```
#import "TalkingDataGA.h"

- (BOOL)application:(UIApplication *)application
    didFinishLaunchingWithOptions:(NSDictionary *)launchOptions{
      [TalkingDataGA onStart:@"APP_ID" withChannelId:@"CHANNEL_ID"];
      //other code

}
```

2. 统计玩家帐户

用途和用法

- 定义一个玩家单元，更新玩家最新的属性信息。
- 在可确定玩家帐户后（注册、登录、切换账户操作完成时都需添加）尽早调用 TDGAAccount.setAccount(String accountId)方法来传入 accountId，对一个玩家的分析依赖于这个 accountId，不调用会造成无数据的情况。
  注意：
  如果您的游戏登录后会在本地保留较长时间免登录状态，每次从后台状态恢复到游戏中不需玩家再次登录，请在每次恢复到游戏时（呈现了游戏界面）也进行以上调用，来确保 accountId 的稳定性，使数据准确。
- accountId 需保证全局唯一，且终生不变。在多设备中传入同样 accountId，数据将归入同一个玩家内，设备激活增加，玩家数不增加；同一设备中传入多个不同 accountId（游戏允许玩家刷小号的情况），会计算多个玩家账户数。

- 玩家的等级、年龄性别等属性发生变化时，尽快调用对应 set 方法来更新帐户属性。每次玩家登录或变更区服后，游戏通常会同步玩家游戏资料，在同步完成时也需调用 setLevel，以确保 Level 准确，数据更精准。

- 注意：如果期望以“设备”为单元来统计分析数据，Game Analytics 也可良好支持，您在 setAccount 时传入[TalkingDataGA getDeviceId]即可，我们会帮助定义和管理设备 ID，全部数据指标也会依据设备 ID 来做计算。详细参考示例 3。

接口说明：（TDGAAccount 类）

```
//设置帐户
+ (TDGAAccount *)setAccount:(NSString *)accountId;
//设置帐户类型
- (void)setAccountType:(TDGAAccountType)accountType;
//设置帐户的显性名
- (void)setAccountName:(NSString *)accountName;
//设置级别
- (void)setLevel:(int)level;
//设置性别
- (void)setGender:(TDGAGender)gender;
//设置年龄
- (void)setAge:(int)age;
//设置区服
- (void)setGameServer:(NSString *)gameServer;
```

参数说明：

参数	类型	描述
accountId	NSString	设定帐户唯一标识，用于区分一个玩家，最多 64 个字符。其他调用依赖于此 ID。 *如果无玩家账户或期望以设备为单位计算玩家，调用时传入 [TalkingDataGA getDeviceId] 即可。
accountType	enum TDGAAccountType	传入账户的类型，预留匿名和自有帐户显性注册两种类型，并支持您自行定义如 QQ、微博等其他类型 匿名: kAccountAnonymous 显性注册：kAccountRegistered

		国内主流的第三方帐号: 新浪微博：kAccountSianWeibo QQ：kAccountQQ 腾讯微博：kAccountTencentWeibo 网龙 91：kAccountND91 另外系统预留了 10 种自定义的帐户类型，分别为 kAccountType1 到 kAccountType10
accountName	NSString	在帐户有显性名时，可用于设定帐户名，最多支持 64 个字符。
level	int	设定玩家当前的级别，未设定过等级的玩家默认的初始等级为“1”，支持的最大级别为 1000。 *等级发生变化时尽快进行调用。 *每次玩家登录、换区服，游戏通常会同步玩家资料，同步完成时需调用 setLevel，可使等级数据精准。 （伴随 setAccount 调用，如等级可确定，都建议在其后调用一下 setLevel。）
gender	enum TDGAGender	设定玩家性别： 男: kGenderMale 女: kGenderFemale 未知：kGenderUnknown
age	int	设定玩家年龄，范围为 0-120。
gameServer	NSString	传入玩家登入的区服，最多 16 个字符。

示例 1：

如一个游戏内 UID 为 10000 的匿名游戏玩家以匿名（快速登录）方式在国服 2 区进行游戏，并在游戏中由 1 级升至 2 级，之后玩家使用 QQ 号 5830000 在游戏中进行显性注册，并设定为 18 岁男玩家的整个过程。

```
TDGAAccount *account = [TDGAAccount setAccount:@"10000"];
[account setAccountType:kAccountAnonymous];
[account setLevel:1];
[account setGameServer:@"国服 2"];
```

在玩家升级时，做如下调用

```
[account setLevel:2];
```

在玩家显性注册成功时做如下调用

```
[account setAccountName:@"5830000@qq.com"];
[account setAccountType:kAccountQQ];
[account setLevel:18];
[account setGender:kGenderMale];
```

示例 2：

如一款必须在注册后进行的不分区服的游戏中，玩家使用其已注册的帐号：xiaoming@163.com（游戏服务器中为玩家分配了 101111 的内部号）登录进行游戏。

```
TDGAAccount *account = [TDGAAccount setAccount:@"101111"];
[account setAccountName:@"xiaoming@163.com "];
[account setAccountType:kAccountRegistered];
```

示例 3：

如在一款类似愤怒小鸟的休闲游戏中，玩家直接进入进行游戏。

您可自行决定唯一 ID 规则，如设备 MAC 地址、IDFA 等，也可以直接使用 TalkingData 提供的设备唯一 ID 接口[TalkingDataGA getDeviceId]，例如：

```
TDGAAccount *account = [TDGAAccount setAccount:[ TalkingDataGA getDeviceId]];
[account setAccountType:kAccountAnonymous];
```

3．跟踪玩家充值

用途和用法

- 跟踪玩家充值现金而获得虚拟币的行为，充入的现金将反映至游戏收入中。
- 充值过程分两个跟踪阶段：1、发出有效的充值请求；2 确认某次充值请求已完成充值。

  您可在玩家发起充值请求时（例如玩家选择了某个充值包，进入支付流程那一刻）调用 onChargeRequest，并传入该笔交易的唯一订单 ID 和详细信息；在确认玩家支付成功时调用 onChargeSuccess ,并告知完成的是哪个订单 ID。

注意：

1、orderID 是标识交易的关键，每一次的充值请求都需要是不同的 orderID，否则会被认为重复数据而丢弃，造成收入数据偏差的情况。

2、orderID 由您自己构造和管理，可以使用类似 userID+时间戳+随机数 的方式来自己定义 orderID，来保障其唯一性。
3、收入数据以调用了 onChargeSuccess 为准，Success 调用时的 orderID 要与 Request 中 orderID 对应，才可追溯到交易内容，有效计量。
Request 必须调用，且需要早于 Success，否则可能影响收入数据的金额计数。

接口说明：（TDGAVirtualCurrency 类）

```
//充值请求
+ (void)onChargeRequst:(NSString *)orderId iapId(NSString *)iapId
currencyAmount:(double)currencyAmount currencyType:(NSString *)currencyType
virtualCurrencyAmount:(double)virtualCurrencyAmount paymentType:(NSString
*)paymentType;
//充值成功
+ (void)onChargeSuccess:(NSString *)orderId;
```

参数说明：

参数	类型	描述
orderId	NSString	订单 ID，最多 64 个字符。 用于唯一标识一次交易。 *如果多次充值成功的 orderID 重复，将只计算首次成功的数据，其他数据会认为重复数据丢弃。 *如果 Success 调用时传入的 orderID 在之前 Request 没有对应 orderID，则只记录充值次数，但不会有收入金额体现。
iapId	NSString	充值包 ID，最多 32 个字符。 唯一标识一类充值包。 例如：VIP3 礼包、500 元 10000 宝石包
currencyAmount	double	现金金额或现金等价物的额度
currencyType	NSString	请使用国际标准组织 ISO 4217 中规范的 3 位字母代码标记货币类型。点击查看参考 例：人民币 CNY；美元 USD；欧元 EUR （如果您使用其他自定义等价物作为现金 亦可使用 ISO 4217 中没有的 3 位字母组合传入货币类型，我们会在报表页面中提供汇率设定功能）
virtualCurrencyAmount	double	虚拟币金额

paymentType	NSString	支付的途径，最多 16 个字符。 例如：“支付宝” “苹果官方” “XX 支付 SDK

示例 1：

玩家使用支付宝方式成功购买了“大号宝箱”（实际为 100 元人民币购入 1000 元宝的礼包），该笔操作的订单编号为 account123-0923173248-11。可以如下调用：

1）在向支付宝支付 SDK 发出请求时，同时调用：

```
[TDGAVirtualCurrency onChargeRequst:@"account123-0923173248-11" iapId: @"大号宝箱" currencyAmount:100 currencyType@"CNY" virtualCurrencyAmount:1000 paymentType: @"Alipay"];
```

2）订单 account123-0923173248-11 充值成功后调用：

```
[TDGAVirtualCurrency onChargeSuccess:@" account123-0923173248-11"];
```

示例 2：

在一款与 91 联运的游戏中，游戏使用了 91 的支付聚合 SDK，玩家购买一个“钻石礼包 1”（10 个 91 豆购买 60 钻石）,该笔操作的订单号为“7837331”。由于此类聚合 SDK 往往要求使用其自有的“代币”（91 使用 91 豆，兑换人民币比例 1：1）做充值依据，建议将“代币”折算为人民币后再调用统计：

1）在向 91 支付 SDK 发出请求时，进行调用

```
[TDGAVirtualCurrency onChargeRequst:@"7837331" iapId: @"钻石礼包 1" currencyAmount:10 currencyType@”CNY” virtualCurrencyAmount:60 paymentType: @"91 SDK "];
```

2）订单 order001 充值成功：

```
[TDGAVirtualCurrency onChargeSuccess: @"7837331"];
```

4. 跟踪获赠的虚拟币（可选）

用途和用法

- 游戏中除了可通过充值来获得虚拟币外，可能会在任务奖励、登录奖励、成就奖励等环节免费发放给玩家虚拟币，来培养他们使用虚拟币的习惯。开发者可通过此方法跟踪全部免费赠予虚拟币的数据。
- 在成功向玩家赠予虚拟币时调用 onReward 方法来传入相关数据。

- 只获得过赠予虚拟币的玩家不会被记为付费玩家。赠予的虚拟币会计入到所有的虚拟币产出中，也计入到留存虚拟币中。

接口说明：（TDGAVirtualCurrency 类）

```
//赠予虚拟币
+ (void)onReward:(double)virtualCurrencyAmount reason:(NSString *)reason;
```

参数说明：

参数	类型	描述
virtualCurrencyAmount	double	虚拟币金额。
reason	NSString	赠送虚拟币原因/类型。 格式：32 个字符内的中文、空格、英文、数字。 不要带有任何开发中的转义字符，如斜杠。 注意：最多支持 100 种不同原因。

示例 1：

玩家在完成了新手引导后，成功获得了免费赠送的 5 个钻石：

```
[TDGAVirtualCurrency onReward:5 reason:@"新手奖励"];
```

示例 2：

玩家在游戏竞技场中排名较高，而获得了 100 消费券奖励：

```
[TDGAVirtualCurrency onReward:100 reason:@"竞技场 Top2"];
```

5. 跟踪游戏消费点

用途和用法

- 跟踪游戏中全部使用到虚拟币的消费点，如购买虚拟道具、VIP 服务、复活等
- 跟踪某物品或服务的耗尽
- 在任意消费点发生时尽快调用 onPurchase，在某个道具/服务被用掉（消失）时尽快调用 onUse
- 消费点特指有价值的虚拟币的消费过程，如果游戏中存在普通游戏金币可购买的虚拟物品，不建议在此处统计。

接口说明：（TDGAItem 类）

```
//记录付费点
+ (void)onPurchase:(NSString *)item itemNumber:(int) number
priceInVirtualCurrency:(double) price;
//消耗物品或服务等
+ (void) onUse:(NSString *)item itemNumber:(int)number;
```

参数说明：

参数	类型	描述
item	NSString	某个消费点的编号，最多 32 个字符。
number	int	消费数量
price	double	虚拟币单价

示例 1：

玩家以 25 元宝/个的单价购买了两个类别号为“helmet1”的头盔 ,可以调用：

```
[TDGAItem onPurchase: @"helmet1" itemNumber:2 priceInVirtualCurrency:25];
```

其中一个头盔在战斗中由于损坏过度而消失。

```
[TDGAItem onUse: @"helmet1" itemNumber:1];
```

示例 2：

玩家在某关卡中死亡，使用 5 个钻石进行复活。可调用：

```
[TDGAItem onPurchase: @"revival" itemNumber:1 priceInVirtualCurrency:5];
```

6. 任务、关卡或副本

用途和用法

- 跟踪玩家任务/关卡/副本的情况。
- 同一个 missionId 如果在未结束前，重复进行了 onBegin 调用，则重新开始计时，上一次的调用被丢弃。
- 如果多个不同的 MissionID 同时在进行（都调用了开始，但并未完成或失败），他们都会同时进行计时，而不是只有一个计时其他暂停计时。

接口说明：（TDGAMission 类）

```
//接到任务
+ (void)onBegin:(NSString *)missionId;
//完成任务
+ (void)onCompleted:(NSString *)missionId;
//任务失败
+ (void)onFailed:(NSString *)missionId failedCause:(NSString *)cause;
```

参数说明：

参数	类型	是否必填	描述
missionId	NSString	必填	任务、关卡或副本的编号，最多 32 个字符。此处可填写 ID，别名可在报表编辑。
cause	NSString	必填	失败原因 ,最多 16 个字符。共支持 100 种原因。

示例 1：

玩家进入名称为“蓝色龙之领地”的关卡。可调用：

```
[TDGAMission onBegin:@"蓝色龙之领地"];
```

玩家成功打过了关卡：

```
[TDGAMission onCompleted:@"蓝色龙之领地"];
```

示例 2：

玩家接到了“主线任务 5”后，又接受了某个直线任务“赚钱 1”，之后他在赚钱任务 1 进行中因为觉得任务过难，放弃任务而失败。

```
[TDGAMission onBegin:@"主线任务 5"];
[TDGAMission onBegin:@"赚钱 1"];
[TDGAMission onFailed:@"赚钱 1" failedCause:@"quit"];
```

7. 自定义事件

用途和用法

- 用于统计任何您期望去跟踪的数据，如：点击某功能按钮、填写某个输入框、触发了某个广告等。
- 可以自行定义 eventId，在游戏中需要跟踪的位置进行调用，注意 eventId 中仅限使用中英文字符、数字和下划线，不要加空格或其他的转义字符。

- 除了可以统计某自定义 eventId 的触发次数外，还可以通过 key-value 参数来对当时触发事件时的属性进行描述。如定义 eventId 为玩家死亡事件，可添加死亡时关卡、死亡时等级、死亡时携带金币等属性，通过 key-value 进行发送。
- 每款游戏可定义最多 200 个不同 eventId，每个 eventId 下，可以支持 20 种不同 key 的 500 种不同 value 取值（NSString 类型），并且注意每个单次事件调用时，最多只能附带 10 种不同 key。

接口说明

- 在游戏程序的 event 事件中加入下面格式的代码，也就成功的添加了一个简单的事件到您的游戏程序中了：

  [TalkingDataGA onEvent:@ "event_id" eventData:your_dictionary];

  注:

  1）NSDictionary 的 Value 目前仅支持字符串（NSString）和数字（NSNumber）类型，key 类型必须是 NSString，一次事件最多只支持 10 个参数。如果 value 为 NSString，Game Analytics 会统计每种 value 出现的次数；如果为 NSNumber 类型，那么 Game Analytics 会统计 value 的总和/平均值。

  2）eventId、Dictionary 的 key 和 NSString 类型的 value，分别最多支持 32 个字符。

示例 1：

使用自定义事件跟踪玩家的死亡情况，并记录死亡时的等级、场景、关卡、原因等信息，可在玩家死亡时调用：

```
// 可定义 eventId=dead
NSDictionary *dic = [ [NSMutableDictionary alloc] init];
[dic setObject:@"50-60" forKey:@"level"];//级别区间
[dic setObject:@"沼泽地阿卡村" forKey:@"map"]; //地图场景
[dic setObject:@"屠龙副本" forKey:@"mission"]; //关卡
[dic setObject:@"PK 致死" forKey:@"reason"]; //死亡原因
[dic setObject:@"10000～20000" forKey:@"coin"]; //携带金币数量
[TalkingDataGA onEvent:@"dead" eventData:dic];
[dic release];
```

注：在某 key 的 value 取值较离散情况下，不要直接填充具体数值，而应划分区间后传入，否则 value 不同取值很可能超过平台最大数目限制，而影响最终展示数据的效果。

如：示例中金币数可能很离散，请先划分合适的区间。

示例 2：

使用自定义事件跟踪玩家在注册过程中每个步骤的失败情况：

```
NSDictionary *dic = [ [NSMutableDictionary alloc] init];
[dic setObject:@"第一步" forKey:@"step"];// 在注册环节的每一步完成时，以步骤名作为 value 传送数据
[dic setObject:@"第二步" forKey:@"step"];
[TalkingDataGA onEvent:@"register" eventData:dic];
[dic release];
```

8. 位置信息

Game Analytics 默认不统计用户的位置信息，只提供了记录接口，请根据苹果公司的审核原则合理使用用户的位置信息，如有需要，可以将已获取的位置信息提交到 Talking Data 统计服务器，服务器只保存最近一次提交的位置信息。

```
[TalkingDataGA setLatitude:纬度  longitude:经度];
```

四、 游戏营销策略

注册 Notification 类型：

```
- (BOOL)application:(UIApplication *)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    if ([application respondsToSelector:
    @selector(isRegisteredForRemoteNotifications)]) {
        // iOS 8 Notifications
        [application registerUserNotificationSettings:
          [UIUserNotificationSettings settingsForTypes:
          UIRemoteNotificationTypeBadge |
          UIRemoteNotificationTypeSound |
          UIRemoteNotificationTypeAlert categories:nil]];
        [application registerForRemoteNotifications];
```

```
    } else {
        // iOS < 8 Notifications
        [application registerForRemoteNotificationTypes:
          UIRemoteNotificationTypeBadge |
          UIRemoteNotificationTypeSound |
          UIRemoteNotificationTypeAlert];
    }
    // other code
}
```

在 application: didRegisterForRemoteNotificationsWithDeviceToken:方法中调用 setDeviceToken 传入 DeviceToken。

```
- (void)application:(UIApplication *)application
didRegisterForRemoteNotificationsWithDeviceToken:(NSData
*)deviceToken {
    [TalkingDataGA setDeviceToken:deviceToken];
}
```

分别在 application: didRegisterForRemoteNotificationsWithDeviceToken:和 application:didReceiveRemoteNotification:方法中调用 handleTDGPushMessage 传入消息。

```
- (BOOL)application:(UIApplication *)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    if (![TalkingDataGA handleTDGAPushMessage:launchOptions]) {
        // 非来自TalkingData的消息，可以在此处处理该消息。
    }
    // other code
}

- (void)application:(UIApplication *)application
didReceiveRemoteNotification:(NSDictionary *)userInfo {
    if (![TalkingDataGA handleTDGAPushMessage:userInfo]) {
        // 非来自TalkingData的消息，可以在此处处理该消息。
    }
}
```

五、 集成检查列表（CHECKLIST）

- 如支持 iOS6.0 以下版本操作系统确保以下框架已经选定为 optional：

  AdSupport.framework；——必须

- 确保已经在 application:didFinishLaunchingWithOptions:方法里面调用

  onStart 方法进行初始化，并填入了正确 AppID；——必须

- 确保调用了 TDGAAccount.setAccount。——必须

  在玩家注册、登录、切换区服、游戏从后台被唤醒时都进行调用，玩家资料同步完成时请调用 setLevel，以确保玩家账户和等级信息正确，数据精确。

- 如果您有添加对玩家充值的追踪，确保不仅只发送了 onChargeSuccess，在成功前必须发送对应订单的 onChargeRequest。确保每笔订单的 orderID 是唯一的。

六、 FAQ

1. 我需要跟踪收入，但是调用接口时我不知道 orderID，怎么办？

答：orderID 您可以在每次订单发起时自己构造一个，保证其唯一性即可，您需要自己管理此 orderID，必要时交付给游戏服务器保存。

2. 为何安装了好几次游戏，设备激活量没有变化？

答：同一台设备反复卸载和安装游戏不会记录多次设备激活；如果是在多个设备上打开的游戏，请稍后几分钟刷新一下页面，查看数据是否变更。

3. Game Analytics 如何判定一台 iOS 设备呢？

答：Game Analytics 首先对操作系统版本进行校验，低于 iOS7 的系统中会通过获取设备 MAC 地址（en0）来确定唯一设备；而 iOS7 和以后的设备已经不可在获取这些信息，我们通过获取 IDFA 和 IDFV 来判定设备，获取不到时会使用 UUID 策略生成 ID。

4. 为何测试渠道包时，渠道数据报表中没有任何数据？

答：请确认是否是在同一台设备上在卸载和安装渠道包进行测试，每台设备只被记录到初装渠道，更换渠道包不会产生新的激活量。测试每个渠道包时请使用一台没测试过的设备进行。

5. 为何测试了多个设备上玩游戏，玩家数没有增加？

答：您是否在多个设备上使用了相同的账户进行了登录呢，那么激活会记录多个，但是玩家数不会增加，使用新的账户进行游戏玩家数才会变化。

6. 为何游戏玩家账户数比设备激活量还高呢？

答：如果玩家在一台设备上更换多个账户登录，就会产生此种情况。如果游戏存在刷小号情况，此现象会更显著。

7. 为何我的玩家账户数高于设备量了，注册转化率却不到 100%呢？

答：注册转化率是衡量所有已激活设备里有多少比例已经成功注册了游戏账户，如果有些设备上用了多个账户做登录，但仍然存在还没有账户注册的激活设备就会存在此种情况。

8. 为何测试收入数据时，有付费次数和人数，但是收入金额是 0 呢？

答：请检查追踪收入当中是否只调用了 success，而没有调用 request 呢。

9. 我的游戏中有多种虚拟币，如何集成呢？

答：Game Analytics 目前只支持记录一种虚拟币，如果您存在多币种，建议按照自身的折算关系将另一种币种折算为首选虚拟币来进行记录。

10. 游戏的每个账户下支持多个角色，如何集成？

答：您可以使用角色 ID 在作为 accountId 来进行集成，但是请注意，如此集成时，总的玩家账户数量会比实际值高。

技术支持方式

如果您在集成和阅读数据时遇到任何问题，请及时与我们取得联系：

- 技术支持邮箱：support@tendcloud.com
- 企业 QQ：4008701230
- 热线：4008701230