

---

百度

# 百度移动游戏统计

---

开发者手册 Android-GAME-1.7 版本

2016/12/16

百度移动统计SDK产品使用说明

## 修改记录

版本号	变更内容
1.5	proguard 配置更新, manifest 权限更新
1.4	proguard 配置说明, manifest 权限更新
1.3	(1) 游戏统计数据实时发送
1.2	(1) 游戏部分的优化包括关卡和等级等
1.1	(1) 增加 exception 截获接口 (2) 减少逻辑偶和
1.0	在 SDK-FOR-SDK1.2 的基础上改造而成, 变更版本号为 1.0 版本 (1) 支持自定义事件时长统计功能 (2) 支持 Fragment 页面统计接口 (只支持顺序调用, 不支持交叉调用) (3) 支持自定义页面统计接口 (只支持顺序调用, 不支持交叉调用) (4) 不支持 SDK 错误收集 (5) 支持 SDK 统计的 log 调试开关 (发布时请去除调用) (6) 支持代码渠道设置接口 (有效防止设置的渠道号丢失) (7) 支持版本号设置 (8) 支持游戏统计 (游戏部分见游戏的文档)

## 目录

前言 关于调试（ <b>注意第五条和第六条</b> ） .....	4
第一章 简介 .....	4
第二章 阅读对象 .....	4
第三章 版本支持 .....	5
第四章 游戏说明 .....	5
第一节 摘要 .....	5
第一节 游戏的初始化 .....	5
第二节 游戏的启动和关闭 .....	6
第三节 统计玩家账户信息 .....	6
第四节 统计玩家充值 .....	6
第五节 统计游戏内消费 .....	7
第六节 统计游戏任务、关卡或副本 .....	7
第七节 日志发送策略 .....	8
第五章 集成使用 .....	8
第一节 申请 APPKEY .....	8
第二节 添加 SDK 到项目 .....	9
第三节 文档使用 .....	9
<b>第四节 配置 AndroidManifest.xml（1.5 修改必读）</b> .....	10
第六章 代码集成 .....	10
第一节 初始化 .....	10
(1) 设置渠道 .....	11
(2) 设置日志发送延迟时间 .....	11
(3) 打开 Exception log 收集 .....	11
第二节 Activity 页面统计 .....	11
第三节 其他页面统计 .....	12
第五节 自定义事件 .....	13
(1) 自定义事件次数统计 .....	13
(2) 自定义事件时长统计 .....	13
第六节 应用版本统计 .....	14
第七节 关于渠道设置 .....	14
第八节 获取 CUID 接口 .....	14
第九节 打开调试 Log 开关 .....	14
第十节 设置发布的 SDK 的版本号 .....	15
第七章 统计实例 .....	15
第八章 统计 SDK 包的目录 .....	15
<b>第九章 混淆相关（1.5 修改必读）</b> .....	15
第十章 联系我们 .....	15

---

## 前言 关于调试（注意第五条和第六条）

- (1) 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。
- (2) 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。
- (3) 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。
- (4) 调试时可以设置 Log 调试开关，具体参照第五章第九节打开 Log 开关。
- (5) 设置版本号需要和设置渠道以及 debug 一样在调用其他统计接口前调用。

## 第一章 简介

百度移动统计游戏 SDK(ANDROID)是百度官方推出的移动游戏统计 SDK 在 ANDROID 平台上的版本(以下简称 SDK)。SDK 的发行版本中包括 JAR 包、文档、示例以及您正在阅读的用户手册。 以下使用<SDK\_PATH>代表 SDK 包解压后的根目录。

- JAR 包：<SDK\_PATH>/ANDROID\_API\_GAME\_1\_\*.JAR 以及 NEED\_LIB.JAR，开发过程中导入 ANDROID 项目；
- 文档：<SDK\_PATH>/ANDROID\_DOC，提供开发详细的接口 API 帮助信息；
- 示例：<SDK\_PATH>/ANDROID\_GAME\_DEMO，统计示例程序代码，帮助您迅速熟悉 SDK 的使用方式；
- 用户手册：本文档。

如果阅读完还没有领会，请导入我们的 DEMO 实例到您的工程中，参考使用。

如有其他问题可以参考网站的 FAQ，或者及时与我们联系（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

可运行于 ANDROID 1.5（API LEVEL 3）及以上版本。

## 第四章 游戏说明

### 第一节 摘要

百度移动游戏统计旨在帮助游戏开发者解决游戏数据的收集以及数据分析问题，并以报表的形式在百度移动游戏开发者平台提供详细的数据分析结果。

针对移动游戏的特点，为了更好地帮助开发者分析游戏数据，在传统的设备维度的数据收集、分支之外，百度移动游戏还提供了用户维度的数据分析，除了玩家的基础信息，还有玩家在对应游戏里面的级别、角色、任务、付费、消费等详细的数据：

1: 设备

安装了游戏的移动终端设备没通过设备的相关信息来标识。

2: 用户（游戏账号）

游戏账号，通过游戏账号名，用户名或者存档号等唯一信息来标识。

3: 付费：

玩家充值现金换取虚拟币的过程。

4: 消费：

指玩家在游戏内消费虚拟币的过程。

针对玩家的一次游戏，是指玩家从打开游戏界面到离开游戏界面的过程。目前的统计分析定义玩家离开游戏后，30S 内重新回到游戏，将被认为是上次游戏的延续，不会重新计算一次游戏启动。

### 第一节 游戏的初始化

游戏的初始化代码包括以下几个函数调用，需要在其他函数调用前调用。

`BDGameSDK.initGame(this, Conf.APPKEY);` //首先调用，否则无游戏数据

`BDGameSDK.setOn(this, BDGameSDK.EXCEPTION_LOG, Conf.APPKEY);` //打开exception的统计

`BDGameSDK.setAccount(this, "accountID2", Conf.APPKEY);` // 设置账户，需要在游戏的所有行为（购买、级别、任务等）调用之前调用，否则无游戏数据。

`StatSDKService.setAppChannel(this, "channelapp", true, Conf.APPKEY);` //设置渠道数据

`StatSDKService.setDebugOn(true, Conf.APPKEY);` //调试时调用，可以看到调试log，log的tag是statsdk

---

StatSDKService.setAppVersionName("1.5", Conf.APPKEY);//设置游戏的版本, 如果不设置默认位1.0

## 第二节 游戏的启动和关闭

游戏的启动和关闭需要在所有的ACTIVITY或者FRAGMENT或者自定义页面里面添加页面统计代码。在ACTIVITY中需要调用

```
StatSDKService.onResume(Context context, "appkey" ),
```

```
StatSDKService.onPause(Context context, "appkey" ),
```

在其他页面中需要调用

```
StatSDKService.onPageStart(Context context, "pageName" "appkey" )和
```

```
StatSDKService.onPageEnd(Context context, "pageName" "appkey" )
```

## 第三节 统计玩家账户信息

定义一个玩家, 通过接口调用实时更新玩家最新的属性信息。需要注意 AccountID 需要保证唯一性, 且不会发生变化。

BDGameSDK.setAccount(Context context, "accountId", "appKey"); // 设置玩家帐户唯一 ID, 如果没有帐户 ID, 可使用 BDGameSDK.getDeviceID()填充

BDGameSDK.setAccountName(Context context, "accountName", "appKey"); // 设置玩家帐户的名称

BDGameSDK.setAccountType(Context context, int type, "appKey"); // 设置玩家帐户类型, 系统定义类型可见 BDGameAccountType, 101 至 200 为预留自定义

BDGameSDK.setLevel(Context context, int level, "appKey"); 设置玩家等级, 默认等级为 1, 最大支持 1000 级

BDGameSDK.setGender(Context context, int gender, "appKey"); // 设置玩家性别, 性别定义参见 BDGender

BDGameSDK.setAge(Context context, int age, "appKey"); // 设置玩家年龄, 范围为 0 ~ 100

BDGameSDK.setServer(Context context, "server", "appKey"); // 设置玩家登录服务器, 建议填写 ASCII 可见字符

## 第四节 统计玩家充值

充值过程分成 2 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经完成。

BDGameSDK.onRecharegeRequest(String orderId, String iapId, double amount, int

---

`virtualAmount, int paymentType, "appKey");` // 在玩家发起充值请求时调用(例如某玩家选择了某充值包, 进入充值流程)

`BDGameSDK.onRechargeSuccess(String orderId, "appKey");` // 玩家充值成功后告知是哪个订单

参数:

`orderId`: 订单 Id, 建议传入 ASCII 可见字符

`iapId`: 充值包 Id, 建议传入 ASCII 可见字符

`amount`: 充值金额, `double` 类型, 保留 2 位小数, 值在 0 ~ 100 万之间, 若充值金额为 0, 而 `virtualAmount` 不为 0, 表示该虚拟币为赠送

`virtualAmount`: 虚拟币充值金额

`paymentType`: 支付途径, 0: 未知; 1: App Store; 2: 百度钱包; 3: 支付宝; 4: 微信支付; 5: 网银; 6: 财付通; 7: 移动; 8: 联通; 9 电信; 10: PayPal。0 ~ 100 为系统预留, 101 ~ 200 用户自定义, 超出 200, 视为充值无效

## 第五节 统计游戏内消费

游戏内消费分为 2 种:

1. 游戏中虚拟币的使用情况

2. 游戏中道具的使用情况

`BDGameSDK.onPurchase(String ppoint, String ppointNumber, int unitPrice, "appKey");`

`BDGameSDK.onUse(String ppoint, int number, "appKey");`

参数:

`ppoint`: 购买道具 Id, 建议传入 ASCII 可见字符

`ppointNumbe`: 购买道具数量

`unitPrice`: 道具单价

`number`: 道具使用数量

## 第六节 统计游戏任务、关卡或副本

跟踪玩家任务/关卡/副本的完成情况, 同一 `taskId` 如果在未结束前, 重新开始, 则该任务重新开始计时, 上一次被丢弃。同时多个 `taskId` 同时进行(都已开始, 但未完成或失败), 是同时进行计时, 而不是暂停其他只计时一个。

`BDGameSDK.onTaskStart(String taskId, "appKey");` // 任务、关卡或副本开始

`BDGameSDK.onTaskFinished(String taskId, "appKey");` // 任务、关卡或副本完成

`BDGameSDK.onTaskFailed(String taskId, String reason, "appKey");` // 任务、关卡或副本失败

参数:

`taskId`: 任务、关卡或副本 Id, 最多 32 个字符, 别名可在报表编辑。建议填写 ASCII 可见字符

`reason`: 玩家任务失败原因

## 第七节 日志发送策略

游戏日志发送策略为实时发送，以下情况下会触发发送：

1. 用户设置等级
2. 用户充值成功
3. 用户购买道具成功
4. 用户使用道具
5. 用户开始任务
6. 用户结束任务

实时日志发送失败，会缓存本地，下次发送触发时再次发送。

如果设置 wifi 下发送，3G 网络状态下的实时发送，亦会缓存本地，下次发送。

## 第五章 集成使用

### 第一节 申请 APPKEY

AppKey (BAIDUMOBAD\_STAT\_ID) 在百度移动统计平台申请，用于标识您的应用程序。这个 KEY 是您的应用在我们的统计平台的唯一标识，请登录百度移动统计平台（登陆地址：[HTTP://MTJ.BAIDU.COM/WEB/WELCOME/LOGIN](http://MTJ.BAIDU.COM/WEB/WELCOME/LOGIN)），然后点击“新增应用”按钮，完成菜单的填写后，点击“下一步”就可以看到生成的 APPKEY。同时请下载 SDK 工具包（包含 ANDROID\_API.JAR、ANDROID\_DOC、ANDROID\_DEMO 等），请解压到本地使用。





---

## 第二节 添加 SDK 到项目

(注：以下提及的各个文件均可在<SDK\_PATH>/android\_demo中找到。)

➤ 将 SDK 中的 JAR 包导入您创建的 Android 项目，以下假设您已经创建了项目 A。

- 右键点击你的工程，然后选择“PROPERTIES”；
- 在工程属性对话框左边选择“JAVA BUILD PATH”；
- 在工程属性对话框主窗口选择“LIBRARIES”；
- 点击“ADD JARs”，添加<SDK\_PATH>/ANDROID\_API\_GAME\_1\_\*.JAR 和 NEED\_LIB.JAR。

## 第三节 文档使用

- 若是在 ECLIPSE 或其他 IDE 下开发，可设置 JAR 包的 JAVADOC 路径为<SDK\_PATH>/DOC，便于在 IDE 中获得即时的开发帮助信息；
- 也可以直接在浏览器中打开<SDK\_PATH>/DOC/INDEX.HTML 查看该文档。

## 第四节 配置 AndroidManifest.xml（1.5 修改必读）

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.baidu.mobstat.demo"
android:versionCode="1" android:versionName="1.0">
    <uses-sdk android:minSdkVersion="4" />
    <!-- 必须声明的权限 -->
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.WRITE_SETTINGS" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />（1.4 新增）
    <uses-permission android:name="android.permission.GET_TASKS" />（1.5 新增）
    <uses-permission android:name="android.permission.BLUETOOTH" />（1.4 新增）
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />（1.4 新增）
    <uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" />（1.4 新增）

    <!-- 可选的权限 -->
    <uses-permission
    android:name="android.permission.ACCESS_FINE_LOCATION" />
```

图1 AndroidManifest.xml

### ■ 添加必要的权限

- ◆ 必须权限：您必须声明的权限（若不声明将统计不到数据）
- ◆ 可选权限：您可根据实际需求设置

## 第六章 代码集成

### 第一节 初始化

初始化的操作需要在接口调用之前调用。渠道设置可以在代码中，可以让开发者通过参数 `true` 来保存渠道，从而杜绝网站渠道统计缺失的情况。每个接口的最后一个参数为您在 `mtj` 上申请的 `APPKEY` 字符串。需要在调用基本接口前保证设置渠道的接口以及调试接口优先调用，初始化一次即可，具体参见 `demo`。

---

示例如下：

在调用其他接口前比如页面或者自定义事件，下面的初始化函数。最后面的字符串"898587128f"为 appkey，需要在 mtj 网站上申请，参见第四章第一节。建议在 onCreate 函数中。

```
StatSDKService.setAppChannel(context, "channelB", true, "898587128f");
StatSDKService.setDebugOn(true, "898587128f");
StatSDKService.setAppVersionName("1.0", "898587128f");
```

## (1) 设置渠道

```
/*
```

(1) 如果第三个参数设置为true（防止渠道代码设置会丢失的情况），将会保存该渠道，每次设置都会更新保存的渠道

(2) appChannel是SDK的发布渠道，不需要在mtj网站上注册，直接填写就可以

(3) 使用时直接使用下面的函数调用就可以，替换成您自己的真实渠道

```
*/
```

```
StatSDKService.setAppChannel(this, "RepleceWithYourChannel", true, "appkey");
```

## (2) 设置日志发送延迟时间

```
/*
```

\* 设置启动时日志发送延时的秒数<br/> 单位为秒，大小为0s到30s之间<br/>

\* 注：请在StatService.setSendLogStrategy之前调用，否则设置不起作用

\* 如果设置的是发送策略是启动时发送，那么这个参数就会在发送前检查您设置的这个参数，表示延迟多少S发送。<br/>

\* 这个参数的设置暂时只支持代码加入， 在您的首个启动的Activity中的onCreate函数中使用就可以。<br/>

```
*/
```

```
StatSDKService.setLogSenderDelayed(seconds, appkey)
```

## (3) 打开 Exception log 收集

```
BDGameSDK.setOn(this, BDGameSDK.EXCEPTION_LOG ,appkey);
```

开关类型的API，功能是否被启用，通过参数flag控制。默认是不收集错误的，如果需要收集，请按照上面的调用方法调用。

## 第二节 Activity 页面统计

开发者需要在每个 Activity 的 onResume()和 onPause()中调用 SDK 提供的 StatSDKService.onResume(Context context, “appkey” )

```
StatSDKService.onPause (Context context, “appkey” );
```

---

context 参数是当前 Activity 的引用,这里请不要将全局的 application context 传入。  
**注意:**

如果开发者在父类 Activity 中添加了这两个函数调用,那么在子类 Activity 中就务必不要添加这两个函数调用。

注: 为了使使用时长等数据更加准确, 建议开发者在每个页面都添加统计接口, 至少要在主界面添加。

**说明:**

- 确保在所有的 activity 中都调用 StatService.onResume(this) 和 StatService.onPause(this)方法, 这两个调用将不会阻塞应用程序的主线程, 也不会影响应用程序的性能。
- 注意如果您的 Activity 之间有继承或者控制关系, 请不要同时在父和子 Activity 中重复添加 onPause 和 onResume 方法, 否则会造成统计数据混乱(例如.使用 TabHost、TabActivity、ActivityGroup 时)。
- 一个应用程序在多个 activity 之间连续切换时, 将会被视为同一个 session(启动)。
- 当用户两次使用之间间隔超过 30 秒时, 将被认为是两个的独立的 session(启动), 例如用户回到 home, 或进入其他程序, 经过一段时间后再返回之前的应用。

**特别注意:**

Activity 的页面的 onResume 以及 onPause 函数不能多个页面交叉调用如出现调用顺序: onResume-----onResume 或者 onPause-----onPause, 统计将会出问题。必须是顺序的 onResume---onPause---- onResume ---- onPause。

如果需要交叉调用, 那么可以使用 onPageStart 和 onPageEnd 自定义页面统计函数来配合使用。onPageStart 和 onResume 是可以交叉调用的。

具体参见 demo 使用。

百度移动统计SDK产品使用说明

### 第三节 其他页面统计

其他页面统计如Fragment或者自定义page, 开发者需要在需要统计的View页面的onPageStart和onPageEnd()中调用SDK提供的

```
StatSDKService.onPageStart(Context context, String pageName, "appkey");
```

```
StatSDKService.onPageEnd(Context context, String pageName, "appkey");
```

context 参数是 Activity 的 Context 或者 application 的 Context 的引用, 这里请注意同一个页面 pageName 必须一样, 否则无法统计到。

**注意:**

如果开发者在父类自定义 View 中添加了这两个函数调用, 那么在子类 View 中就务必不要添加这两个函数调用。

注: 为了使使用时长等数据更加准确, 建议开发者在每个页面都添加统计接口, 至少要在能够填充整个界面的自定义 View 中添加。

## 第五节 自定义事件

### (1) 自定义事件次数统计

```
void onEvent(Context context, String event_id, String label, "appkey" )
void onEvent(Context context, String event_id, String label, int acc,
"appkey" )
```

参数:

context	设备上下文
event_id	业务端注册的事件 id
label	event id 下的各种事件添加的标签
acc	事件的发生次数, 不指定时值为 1.

**嵌入位置:** 任何地方。注意: 时长统计函数中的 event\_id 应该与 onEvent 函数中的 event\_id 不同。

**功能:** 统计开发者的自定义事件, 自增计数。单纯的统计自定义事件的次数。这里的 event\_id 需要开发者在 mtj 网站上添加自定义事件的 id 相同。例如: 我们要统计更新的各个渠道的次数, 那么我们可以定义 event\_id 为 update, 更新 1、更新 2 等各个渠道的更新表示 label。

那么在 mtj.baidu.com 网站上需要相同的 eventId: registered id  
自定义事件管理 ?



### (2) 自定义事件时长统计

#### (1) 使用事件开始和结束统计事件时长

```
void onEventStart(Context context, String event_id, String label, "appkey")
void onEventEnd(Context context, String event_id, String label, "appkey")
```

**嵌入位置:** 任何地方, 配对使用。注意: 时长统计函数中的 event\_id 应该与 onEvent 函数中的 event\_id 不同。

**功能:** 统计开发者的自定义事件时长统计

参数:

---

context	设备上下文
event_id	业务端注册的事件 id（需在 mtj 网站上注册）
label	事件添加的标签
appkey	mtj 上申请的 sdk-app 的唯一标识

也可以直接使用函数 `onEventDuration`，传入自己计算的事件时长：

(2) 直接传入某事件的时长：

```
void onEventDuration(Context context, String event_id, String label, int milliseconds, "appkey")
```

注：参数 `milliseconds` 的单位为 毫秒。该值是您自己计算出来的事件时长。

## 第六节 应用版本统计

请在 `AndroidManifest.xml` 文件中设置 `versionName`，SDK 会读取该版本号来作为你的 app 的版本。

## 第七节 关于渠道设置

详见第五章第一节（设置渠道一段文字）以及第四章第四节例子。

在代码中设置渠道，请在调用其他接口前优先调用一次即可。

```
StatSDKService.setAppChannel(this, "RepleceWithYourChannel", true, "appkey");
```

## 第八节 获取 CUID 接口

`String getCuid(Context context)`（SDK3.1 新增）

1. 一般在 `Activity.onCreate` 函数中调用。
2. 功能：用于取得设备的 CUID 号。
3. 参数：`context` --- `Activity` 的上下文。
4. 调用函数：

```
StatSDKService.getCuid(this);
```

## 第九节 打开调试 Log 开关

调试时，可以在第一节初始化的地方增加调试 Log 的打开调用，这样你在接入 sdk 的时候可以通过 Log 的打印来看接入的情况。发布时去除该行代码即可。  
打开调试的代码：

```
StatSDKService.setDebugOn(true, "appkey");
```

---

## 第十节 设置发布的 SDK 的版本号

第一个参数是版本号，第二个参数是 appkey。具体注意事项参见第五章第一节初始化。

```
StatSDKService.setAppVersionName("1.0", "898587128f");
```

## 第七章 统计实例

Demo 工程已经嵌入了 SDK 提供的所有功能接口，开发者只需将 APPKey 更换成自己申请的 ID，然后运行，在网络畅通的情况下，统计数据就会被发送到业务端，开发者可以查看相应报表。Demo 工程中的接口都有详细注释，使用时请注意用法。

## 第八章 统计 SDK 包的目录

zip 包中解压会有以下目录结构：

- JAR 包：BAIDU\_MOBSTAT\_ANDROID\_FOR\_GAME\_SDK\_\*.JAR，开发过程中导入 Android 项目，如果 app 同时使用了移动统计 app 统计 SDK，并且在导入后出现 Duplicate 错误，根据需求选择保留一个即可；
- 文档：<SDK\_PATH>/ANDROID\_DOC，提供开发帮助信息；
- 示例：<SDK\_PATH>/ANDROID\_GAME\_DEMO，统计示例程序代码，帮助您迅速熟悉 SDK 的使用方式；
- 用户手册：本文档。

## 第九章 混淆相关（1.5 修改必读）

SDK 已经做了相关必要混淆，开发者集成此 SDK 时无需再进行混淆，不当的混淆可能造成 SDK 功能不可用甚至崩溃。因此，请使用 keep 选项指定不对 SDK 进行混淆，具体在混淆配置文件 proguard-project.txt 中添加如下 keep 配置即可：

```
-keep class com.baidu.bottom.** { *; }  
-keep class com.baidu.mtjstatsdk.** { *; }
```

## 第十章 联系我们

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