# 集成文档

# 1. **Android 游戏分析 SDK使用指南**

## 1.1新增场景类型设置接口

场景类型设置接口。

MobclickAgent.setScenarioType(Context context, EScenarioType etype)

etype是官方场景，有如下场景:

EScenarioType. E\_UM\_NORMAL 普通统计场景类型

EScenarioType. E\_UM\_GAME 游戏场景类型

EScenarioType. E\_UM\_ANALYTICS\_OEM 统计盒子场景类型

EScenarioType. E\_UM\_GAME\_OEM 游戏盒子场景类型

## 1.2 UTDID相关

为防止冲突UTDID包单独提供,如果需要请导入UTDID包

# 2.介绍

友盟游戏分析服务可以跟踪移动应用的用户属性及操作行为，并且通过友盟的网站后台展示数据处理结果，帮助您调整产品、运营、推广策略的决策。我们目前支持iOS，Android平台。您可以在五分钟内轻松完成集成过程。

注意：

使用游戏统计分析需使用全新的友盟APPKEY。如果有游戏或应用正在使用友盟统计分析，请勿在中途替换成游戏统计分析！

\*特别提醒：我们建议开发者在注册账号时使用企业邮箱，避免使用个人邮箱注册，防止由于个人离职带来的问题，建议使用的账号形式 ：umeng@企业域名、apps@企业域名、dev@企业域名。

下载的压缩包中将包括以下内容:

| **文件名称** | **介绍** |
| --- | --- |
| example/ | 该文件夹中包含了示例工程 |
| libs/ | 该文件夹中包含了SDK的库文件 |
| readme.txt | 关于SDK的简短说明 |
|  |  |

# 3.基本功能集成指南

**提示**   
请先在友盟的管理后台中创建App，获得AppKey

* 导入SDK
  + [下载](http://dev.umeng.com/game-android-sdk-quick-download) umeng\_sdk\_\*\_analytics.zip并解压缩
  + 导入插件  
    下载最新版SDK的zip包，将其中的libs 文件夹合并到本地工程libs子目录下。
  + 配置  
    Eclipse用户右键工程根目录，选择Properties -> Java Build Path -> Libraries，然后点击Add External JARs... 选择指向jar的路径，点击OK，即导入成功。（ADT17及以上版本不需要手动导入）
* 配置 AndroidManifest.xml
* <manifest……>
* <application ……>
* ……
* <activity ……/>
* <meta-data android:value="YOUR\_APP\_KEY" android:name="UMENG\_APPKEY"></meta-data>
* <meta-data android:value="Channel ID" android:name="UMENG\_CHANNEL"/>
* </application>
* <uses-sdk android:minSdkVersion="4"></uses-sdk>
* <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE"></uses-permission>
* <uses-permission android:name="android.permission.INTERNET"></uses-permission>
* <uses-permission android:name="android.permission.READ\_PHONE\_STATE"></uses-permission>
* <uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE"></uses-permission>
* </manifest>
* 权限说明

|  |  |
| --- | --- |
| **META-DATA** | **用途** |
| UMENG\_APPKEY | 用来定位该应用程序的唯一性。 |
| UMENG\_CHANNEL | 用来标注应用推广渠道，区分新用户的来源来查看统计，您可以使用20位以内的英文和数字的组合为渠道定名，替换value中的"Channel ID"。详见渠道统计。 注意不要使用纯数字！ |

|  |  |
| --- | --- |
| **权限** | **用途** |
| INTERNET(必须) | 允许应用程序联网，以便向我们的服务器端发送数据。 |
| READ\_PHONE\_STATE(必须) | 获取用户手机的IMEI，用来唯一的标识用户。(如果您的应用会运行在无法读取IMEI的平板上，我们会将mac地址作为用户的唯一标识，请添加权限： android.permission.ACCESS\_WIFI\_STATE ) |
| ACCESS\_NETWORK\_STATE(必须) | 检测网络状态，友盟SDK 1.6版本以上需要此权限。 |

* 添加代码  
  在游戏所在的Activity的onCreate方法调用 UMGameAgent.init(Context) 方法。在onResume方法中调用UMGameAgent.onResume(Context), onPause方法中调用UMGameAgent.onPause(Context)
* **public** **void** **onCreate**(Bundle savedInstanceState) {
* **super**.onCreate(savedInstanceState);
* UMGameAgent.setDebugMode(**true**);*//设置输出运行时日志*
* UMGameAgent.init( **this** );
* }
* **public** **void** **onResume**() {
* **super**.onResume();
* UMGameAgent.onResume(**this**);
* }
* **public** **void** **onPause**() {
* **super**.onPause();
* UMGameAgent.onPause(**this**);
* }

**说明**

1. Eclipse ADT 17 以上版本用户，请在工程目录下建一个文件夹libs，把jar包直接拷贝到这个文件夹下，再在>Eclipse里面刷新一下工程就好了。不要通过上述步骤手动添加jar包引用。 详情请参考[Dealing with dependencies in Android projects](http://tools.android.com/recent/dealingwithdependenciesinandroidprojects).
2. 方法将会自动地从AndroidManifest.xml文件里读取UMENG\_APPKEY。
3. 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。

到此，基本统计功能集成已经完成。

## **3.1发送策略**

发送策略定义了用户由统计分析SDK产生的数据发送回友盟服务器的频率。

您需要在程序的入口 Activity 中添加

MobclickAgent.updateOnlineConfig( mContext );

Android平台的数据发送策略有两种方式：  
\* 启动时发送：APP启动时发送当次启动数据和上次的使用时长等缓存数据，当次使用过程中产生的自定义事件数据缓存在客户端，下次启动时发送  
\* 按间隔发送：按特定间隔发送数据，间隔时长介于90秒与1天之间。您可以在后台自定义发送间隔。

在没有取到在线配置的发送策略的情况下，会使用默认的发送策略：启动时发送。  
你可以在友盟后台“设置->发送策略”页面自定义数据发送的频率。

# 4.游戏接口集成

## **4.1关卡统计集成**

我们提供了一组方法来统计关卡相关的事件。

UMGameAgent.startLevel(String level); *//level 为关卡ID(eg：level-01)*

UMGameAgent.failLevel(String level);

UMGameAgent.finishLevel(String level);

在游戏开启新的关卡的时候调用 UMGameAgent.startLevel(level) 方法，在关卡失败的时候调用UMGameAgent.failLevel(level) 方法，在成功过关的时候调用 UMGameAgent.finishLevel(level) 方法。调用 failLevel 或 finishLevel 的时候会计算从 startLevel 开始的时长，作为这一关卡的耗时。level字段最好为非空可排序的字符串。

SDK 会默认减去程序切入后台的时间，可以设置 UMGameAgent.setTraceSleepTime(false); 来关闭默认设置，这样统计到的关卡时长是从 startLevel 到 failLvel(finishLevel) 之间的真实时长。

测试方法及报表对应关系请见[产品使用文档](http://dev.umeng.com/game_analytics/gt/game-test" \l "5_4" \t "_blank" \o "使用说明)。

**注意**

1. 目前每个游戏最多支持500个关卡。

## **4.2真实消费统计**

这部分API用来统计用户(或者玩家) 在游戏内付费的统计，包括购买虚拟币，道具等。

API 说明： public static void pay(double money, double coin, int source)   
money 本次消费金额(非负数)   
coin 本次消费的等值虚拟币(非负数)   
source 支付渠道, 1 ~ 99 之间的整数， 1-20 已经被预先定义, 21~99 之间需要在网站设置。

如下表：

| **数值** | **含义** |
| --- | --- |
| 1 | Google Play |
| 2 | 支付宝 |
| 3 | 网银 |
| 4 | 财付通 |
| 5 | 移动通信 |
| 6 | 联通通信 |
| 7 | 电信通信 |
| 8 | paypal |

在游戏中充值或者购买虚拟币的时候调用此方法，比如通过支付宝用 10元钱 购买了 1000 个金币，可以这样调用：

UMGameAgent.pay(10,1000,2);

有些时候在游戏中会直接购买某个道具，比如10元购买 2个魔法药水,每个药水50个金币，可以调用下面的方法在付费的同时购买道具。

UMGameAgent.pay(10,"magic\_bottle",2,50,2);

public static void pay(double money,String item , int number, double price,int source)   
money 本次消费的金额(非负数)   
item 购买物品的ID(不能为空)   
number 购买物品数量(非负数)   
price 每个物品等值虚拟币的价格(非负数)   
source 支付渠道 (见上表)

测试方法及报表对应关系请见[产品使用文档](http://dev.umeng.com/game_analytics/gt/game-test" \l "5_1" \t "_blank" \o "使用说明)。

### 4.2.1 带订单的真实消费

这部分API用来统计用户(或者玩家) 在游戏内付费的统计,可以统计不同的付费金额和币种,还支持统计订单ID等。

API 说明： exchange(**double** currencyAmount, String currencyType, **double** virtualAmount, **int** channel, String orderId)  
currencyAmount本次消费金额(非负数)   
currencyType 本地消费金额的币种(可为空)  
virtualAmount本次消费的等值虚拟币(非负数)

channel 支付渠道, 1 ~ 99 之间的整数， 1-20 已经被预先定义, 21~99 之间需要在网站设置。

orderId 付费的订单ID(可为空)

## **4.3虚拟消费统计**

游戏中所有虚拟消费，比如用金币购买某个道具都使用 buy 方法, 声明如下：

public static void buy(String item, int number,double price)   
item 购买物品的ID   
number 购买物品数量   
price 购买物品的单价(虚拟币)

比如在游戏中使用金币购买了1个头盔，一个头盔价值 1000 金币，可以这样统计：

UMGameAgent.buy("helmet", 1, 1000);

测试方法及报表对应关系请见[产品使用文档](http://dev.umeng.com/game_analytics/gt/game-test" \l "5_2" \t "_blank" \o "使用说明)。

## **4.4物品消耗统计**

游戏中的物品损耗，比如使用了2瓶魔法药水,每个需要50个虚拟币，可以这样统计：

UMGameAgent.**use**("magic\_bottle", 2 , 50);

public static void use(String item, int number,double price)   
item 消耗的物品ID   
number 消耗物品数量   
price 物品单价（虚拟币）

测试方法及报表对应关系请见[产品使用文档](http://dev.umeng.com/game_analytics/gt/game-test" \l "5_3" \t "_blank" \o "使用说明)。

**注意**

1. 目前每个游戏最多支持1000个道具的统计。

## **4.5额外奖励**

针对游戏中额外获得的虚拟币进行统计，比如系统赠送，节日奖励，打怪掉落。

比如连续5天登陆游戏奖励1000金币。

UMGameAgent.bonus(1000, 1);

public static void bonus(double coin, int trigger)   
coin 赠送的虚拟币数额   
trigger 触发奖励的事件, 取值在 1~10 之间，“1”已经被预先定义为“系统奖励”， 2~10 需要在网站设置含义。

如果是掉落道具,比如掉落一把价值100金币的宝剑可以这样调用：

UMGameAgent.bonus("sword", 1, 100, 1);

public static void bonus(String item, int num, double price,int trigger)   
item 奖励物品ID   
num 奖励物品数量   
price 物品的虚拟币单价   
trigger 触发奖励的事件, 取值在 1~10 之间，“1”已经被预先定义为“系统奖励”， 2~10 需要在网站设置含义。

如果在五一假期中举行登陆有奖活动，开发者想统计活动期间所赠送的金币数量，可以这样调用：

UMGameAgent.bonus(1000, 2);

在后台“设置”——“虚拟币来源管理”中将“2”这个事件设为节日奖励。即可看到相应的数据了。

注意：所有的浮点类型，只精确到百分位；所有的数值类型不能为负数，否则不予处理。tigger的取值在1~10之间，“1”已经被预先定义为“系统奖励”，超过这个范围将不予处理。

## **4.6玩家信息统计**

增加玩家账号统计维度，同时可以追踪玩家的等级情况。以便结合关卡和付费数据整理出更有价值的用户。开发者应该在程序开始的地方尽早调用。

### 4.6.1账号统计

**注意：使用老版SDK中账号统计的开发者，请尽早跟新SDK并使用新接口**

**public** **static** **void** **onProfileSignIn**(String ID) ;

**public** **static** **void** **onProfileSignIn**(String Provider, String ID) ;

ID：玩家账号ID，长度小于64字节

Provider：账号来源。如果玩家通过第三方账号登陆，可以调用此接口进行统计。不能以下划线"\_"开头，使用大写字母和数字标识，长度小于32字节; 如果是上市公司，建议使用股票代码。

**public** **static** **void** **onProfileSignOff**();

账号登出时需调用此接口，调用之后不再发送账号相关内容。

当玩家使用游戏自有账号登录时，可以这样统计：

UMGameAgent.onProfileSignIn("playerID");

当玩家使用第三方账号（如新浪微博）登录时，可以这样统计：

UMGameAgent.onProfileSignIn("WB"，"playerID");

### 4.6.2等级统计接口

当玩家建立角色或者升级时，需调用此接口

**public** **static** **void** **setPlayerLevel**(**int** level);

level：大于1的整数，最多统计1000个等级

当玩家从1级升至2级时，可以这样统计：

UMGameAgent.setPlayerLevel(2);

# 5.高级功能集成指南

## **5.1使用自定义事件**

自定义事件可以实现在应用程序中埋点来统计用户的点击行为。自定义事件目前包括“计数事件”和“计算事件”，二者的区别以及详细说明请**[点击这里](http://dev.umeng.com/analytics/functions/numekv" \o "http://dev.umeng.com/analytics/functions/numekv)**

**☺自定义事件使用中的问题请参见友盟开发者社区：[自定义事件常见问题](http://bbs.umeng.com/thread-5417-1-1.html" \t "_blank" \o "http://bbs.umeng.com/thread-5417-1-1.html)**

### 5.1.1自定义事件字段说明

event id：自定义事件id   
key：自定义事件下的参数   
value：自定义事件参数下的参数值

### 5.1.2使用自定义事件的依赖条件

1. 使用自定义事件功能请先登陆[**友盟官网**](http://www.umeng.com/)， 在 “统计分析->设置->事件” （子账户由于权限限制可能无法看到“设置”选项，请联系主帐号开通权限。）页面中添加相应的事件id（事件id可用英文或数字，不建议使用中文），然后服务器才会对相应的事件请求进行处理。
2. 自定义事件的代码需要放在Activity里的onResume方法后面，不支持放在onCreat()方法中。

### 5.1.3计数事件

使用计数事件需要在后台添加事件时选择“计数事件”。

#### 5.1.3.1统计发生次数

在您希望跟踪的代码部分，调用如下方法：

MobclickAgent.onEvent(Context context, String eventId);

context指当前的Activity，eventId为当前统计的事件ID。

示例：统计微博应用中"转发"事件发生的次数，那么在转发的函数里调用

MobclickAgent.onEvent(mContext,"Forward");

#### 5.1.3.2统计点击行为各属性被触发的次数

考虑事件在不同属性上的取值，可以调用如下方法：

MobclickAgent.onEvent(Context context, String eventId, HashMap<String,String> map);

map 为当前事件的属性和取值（Key-Value键值对）。

示例：统计电商应用中“购买”事件发生的次数，以及购买的商品类型及数量，那么在购买的函数里调用：

HashMap<String,String> map = **new** HashMap<String,String>();

map.put("type","book");

map.put("quantity","3");

MobclickAgent.onEvent(mContext, "purchase", map);

### 5.1.4计算事件

使用计算事件需要在后台添加事件时选择“计算事件”。

#### 5.1.4.1统计数值型变量的值的分布

统计一个数值类型的连续变量（该变量必须为整数），用户每次触发的数值的分布情况，如事件持续时间、每次付款金额等，可以调用如下方法：

MobclickAgent.onEventValue(Context context, String id, Map<String,String> m, int du)

id 为事件ID  
map 为当前事件的属性和取值   
du 为当前事件的数值为当前事件的数值，取值范围是-2,147,483,648 到 +2,147,483,647 之间的有符号整数，即int 32类型，如果数据超出了该范围，会造成数据丢包，影响数据统计的准确性。

示例：统计一次音乐播放，包括音乐类型，作者和播放时长，可以在音乐播放结束后这么调用：

int duration = 12000; *//开发者需要自己计算音乐播放时长*

　　Map<String, String> map\_value = **new** HashMap<String, String>();

　　map\_value.put("type", "popular");

　　map\_value.put("artist", "JJLin");

MobclickAgent.onEventValue(**this**, "music", map\_value, duration);

上面方法在Android统计分析V5.2.2之后(>=)才提供，之前的版本可以通过下面的方式封装计算事件：

public static **void** onEvent(Context context, String id, HashMap<String,String> m, long value){

m.put("\_\_ct\_\_", String.valueOf(value));

MobclickAgent.onEvent(context, id, m);

}

**注：另外，onEventBegin(...),onEventEnd(...)以及onEventDuration(...)后续不再支持，不建议使用。如需统计使用时长，开发者需要自己上传时长数据（如示例中的duration所示），时长数据会在计算事件的数值型部分展示；如果使用了onEventBegin(...),onEventEnd(...)以及onEventDuration（...）方法，时长数据会在计算事件的字符串型中显示。**

如图：使用了onEventBegin(...),onEventEnd(...)以及onEventDuration（...）方法，时长数据会在计算事件的字符串型中显示。此方法已不建议使用。



如图：开发者使用计算事件自己上传时长数据（如示例中的duration所示），时长数据会在计算事件的数值型部分展示。



### 5.1.5统计点击次数及各属性触发次数

计算事件除能够统计数值型的参数外还具有计数事件的所有功能，**[详细说明](http://dev.umeng.com/analytics/functions/numekv" \o "http://dev.umeng.com/analytics/functions/numekv)**。

### 5.1.6注意事项

1. event id不能使用特殊字符，不建议使用中文，且长度不能超过128个字节；map中的key和value 都不能使用特殊字符，key 不能超过128个字节，value 不能超过256个字节。
2. id， ts， du是保留字段，不能作为event id及key的名称。
3. 每个应用至多添加500个自定义事件，每个event 的 key不能超过10个，每个key的取值不能超过1000个。如需要统计支付金额、内容浏览数量等数值型的连续变量，请使用**计算事件**（不允许通过key-value结构来统计类似搜索关键词，网页链接等随机生成的字符串信息）。

## **5.2使用分发渠道分析**

有时需要统计应用程序的分发渠道，例如有多少用户来从应用汇 下载了您的应用,又有多少用户通过GooglePlay下载到您的应用程序。您只需要在AndroidManifest.xml里添加meta-data，并将 value属性修改为对应的发布渠道名。

**注意：**

每台设备仅记录首次安装激活的渠道，在其他渠道再次安装不会重复计量。 所以在测试不同的渠道的时候，请使用不同的设备来分别测试 不要改变'UMENG\_CHANNEL'，修改'Channel ID'为您的渠道名称，注意不能是纯数字(eg.value="GooglePlay")

配置AndroidManifest.XML添加下面代码

<application ……>

<activity ……/>

<meta-data android:value="Channel ID" android:name="UMENG\_CHANNEL"/>

</application>

当然，这需要您在不同渠道发布应用程序时，重新编译打包。

## **5.3在线参数**

在Android统计分析SDK5.6.1及以上版本中，setOnlineConfigureListener()这个接口已经废弃，如果使用在线参数功能，请下载独立的在线参数SDK 。

在线参数SDK下载链接：[http://dev.umeng.com/online-parameters/android-doc/sdk-download](http://dev.umeng.com/online-parameters/android-doc/sdk-download" \t "_blank" \o "http://dev.umeng.com/online-parameters/android-doc/sdk-download)

**以下【在线参数】内容仅针对5.6.1.0以前版本的SDK**

在线参数使您可以在网站上动态配置预设好的参数的参数值。例如： 动态修改应用的欢迎语，修改应用中开关选项的"on"或"off"，以及类似游戏中虚拟物品的价格等。

您需要在程序的入口 Activity 中添加

MobclickAgent.updateOnlineConfig( mContext );

在线参数需要在网站编辑，在程序中使用这句代码从服务器获取参数，并缓存本地。 当在项目里需要获取某个具体参数时调用

String value = MobclickAgent.getConfigParams( mContext, "xxxx" );

xxxx为友盟后台事先设置好的参数id。

同时，我们提供了在线参数回调接口，注意此接口只在在线参数有变化的时候才会回调，实现此接口：

MobclickAgent.updateOnlineConfig(**this**);

MobclickAgent.setOnlineConfigureListener(**new** UmengOnlineConfigureListener(){

@Override

**public** **void** **onDataReceived**(JSONObject data) {

}

});

data 为在线参数回调函数，JSONObject 对应网页上编辑的参数值。

## **5.4页面统计**

页面统计集成正确，才能够获取正确的页面访问路径、访问深度（PV）的数据。  
页面访问路径页面的数据为抽样统计（抽样比例为所有log的3%），数据量达到一定程度才会有数据，若您希望查看测试数据，请使用[集成测试服务](http://dev.umeng.com/analytics/reference/functions/testmode)。

### 5.4.1只由Activity构成的应用

如果您已完成上述步骤，那么SDK已默认统计了每个Activity的跳转路径。页面统计不需要再添加其他代码。

### 5.4.2包含Activity、Fragment或View的应用

在程序入口处，调用 MobclickAgent.openActivityDurationTrack(false) 禁止默认的页面统计方式，这样将不会再自动统计Activity。  
仍需在每个Activity中调用onResume()和onPause()方法。

然后在您要统计的Activity、Fragment或View中调用如下方法（pageName不能使用中文）：

MobclickAgent.onPageStart(String pageName);

MobclickAgent.onPageEnd(String pageName);

应用程序页面的实现可能是基于Activity也可能是Fragment，推荐在相应的 onResume 和 onPause 方法中调用，比如 Activity :

**public** **void** **onResume**() {

**super**.onResume();

MobclickAgent.onPageStart("SplashScreen");

MobclickAgent.onResume(**this**);

}

**public** **void** **onPause**() {

**super**.onPause();

MobclickAgent.onPageEnd("SplashScreen"); *// 保证 onPageEnd 在onPause 之前调用,因为 onPause 中会保存信息*

MobclickAgent.onPause(**this**);

}

比如 Fragment：

**public** **void** **onResume**() {

**super**.onResume();

MobclickAgent.onPageStart("MainScreen");

}

**public** **void** **onPause**() {

**super**.onPause();

MobclickAgent.onPageEnd("MainScreen");

}

这组方法的调用，需要保证线性不交叉，每个 start 都有一个 end 配对，如下：

onPageStart ->onPageEnd-> onPageStart -> onPageEnd -> onPageStart ->onPageEnd

这样才能保证每个页面统计的正确，关于页面统计API的最佳实践说明见[这里](http://blog.umeng.com/index.php/2013/08/%E5%8F%8B%E7%9B%9F%E7%BB%9F%E8%AE%A1%E5%88%86%E6%9E%90%EF%BC%9Afragment%E9%A1%B5%E9%9D%A2%E7%BB%9F%E8%AE%A1/) 。

# 6.社交统计

针对社交行为的垂直统计，可以非常详尽地统计应用中发生的各种社交行为。 只需要调用一行代码，便可享用到丰富的社交行为和社交用户分析报表。

在发生社交行为，比如分享了到新浪微博之后，调用这样一行代码：

UMPlatformData platform = **new** UMPlatformData(UMedia.SINA\_WEIBO, "user\_id");

platform.setGender(GENDER.MALE); *//optional*

platform.setWeiboId("weiboId"); *//optional*

MobclickAgent.onSocialEvent(**this**, platform);

会把分享信息发送到友盟服务器, 我们会通过这些信息创建社交行为报表。

相关参数说明：  
UMPlatformData：  
UMeida meida 平台枚举类型（必填）  
String usid 用户的id（必填）  
String weiboId 微博id   
String name 用户姓名   
GENDER gender 用户性别

# 7.其他API

* \*MobclickAgent.setOpenGLContext(GL10 gl) 设置openGL 信息，辅助统计GPU 信息
* MobclickAgent.onKillProcess(Context context) 如果开发者调用 Process.kill 或者System.exit 之类的方法杀死进程，请务必在此之前调用此方法，用来保存统计数据。

# 8.测试与调试

UMGameAgent 继承自 MobclickAgent 包含 MobclickAgent 提供的所有方法。

## **8.1开启调试模式**

UMGameAgent.setDebugMode( true );

## **8.2 调试步骤**

1. 确认所需的权限都已经添加：INTERNET, READ\_PHONE\_STATE
2. 确认APPKEY已经正确的写入Androidmanifest.xml
3. 确认所有的Activity中都调用了onResume和onPause方法
4. 确认测试手机(或者模拟器)已成功连入网络
5. 启动应用程序，几分钟之后您应该已经可以看到相应的报表.
6. 如果几分钟后报表中仍然没有收到数据，请与我们的技术支持联系 **QQ:800083942** 或者邮件到\*\*support@umeng.com\*\*我们会尽快回复您的报表.
7. [常见错误汇集](http://dev.umeng.com/faq/faq_com.html)

# 9.其他

1. UMGameAgent.setDebugMode(boolean enable) 设置是否debug模式，在debug模式下，会有log输出，方便调试 (默认true)
2. UMGameAgent.setAutoLocation(boolean enable) 设置是否允许收集地理位置信息 （默认true）
3. UMGameAgent.setSessionContinueMillis(long interval) 设置session启动的沉默参数，详见30秒规则 （默认30000）
4. UMGameAgent.openActivityDurationTrack(boolean enable) 设置是否打开页面路径访问功能（默认打开）
5. UMGameAgent.setOpenGLContext(GL10 gl) 设置openGL 信息，辅助统计GPU 信息
6. UMGameAgent.onKillProcess(Context context) 如果开发者调用 Process.kill 或者 System.exit 之类的方法杀死进程，请务必再次之前调用此方法，用来保存统计数据。
7. UMGameAgent.reportError(Context context,String error) 此方法会把传入的错误信息发回服务器

# 10.技术支持

[点击访问：友盟开发者社区](http://bbs.umeng.com/forum-gamesanalytics-1.html)

QQ:800083942 (在线时间：工作日10:00~18:00)  
Email：support@umeng.com请发邮件至[support@umeng.com](mailto:support@umeng.com)，我们会尽快回复您。

为了能够尽快响应您的反馈，请提供您的appkey及log中的详细出错日志，您所提供的内容越详细越有助于我们帮您解决问题。