**Merculet集成文档（Android）**

[一. 集成准备 2](#_Toc496301441)

[1.1添加资源文件 2](#_Toc496301442)

[1.2添加权限(AndroidManifest.xml内) 2](#_Toc496301443)

[1.3添加merculet AppId信息 3](#_Toc496301444)

[1.3.1后台获取merculet Appkey等信息 3](#_Toc496301445)

[1.3.2代码内添加(AndroidManifest.xml内) 3](#_Toc496301446)

[二．基本功能集成（必加项） 4](#_Toc496301447)

[2.1初始化SDK 4](#_Toc496301448)

[2.2 Session功能 4](#_Toc496301449)

[2.3 自定义事件功能 5](#_Toc496301449)

[三．测试与调试 9](#_Toc496301463)

[4.1测试前的准备 9](#_Toc496301464)

[4.2测试流程 9](#_Toc496301465)

[4.3 FAQ 9](#_Toc496301466)

# 一. 集成准备

## 1.1导入SDK

导入SDK有以下三种方法，选择其中一种即可 。

### 1.1.1  eclipse开发环境SDK集成

将需要的jar包拷贝到本地工程libs子目录下；在Eclipse中右键工程根目录，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及以上不需要手动导入）

### 1.1.2 使用Android Studio导入SDK

Android Studio是谷歌推出了新的Android开发环境，【友盟+】支持AndroidStudio的Gradle配置，如您使用Android Studio开发，请在您的App对应build.gradle文件中加入对统计SDK和Common通用组件库的依赖：

dependencies

{

compile file(dir:'libs',**include**:[' MerculetSDK.jar'])

}

### 1.1.3  添加SDK在maven中心库的线上依赖

通过在Android Studio工程build.gradle配置脚本中添加maven线上依赖，导入最新版本组件化基础库和统计SDK。

在Gradle依赖中添加：

dependencies {

compile 'io.merculet:MerculetSDK:1.0.1'

}

如果无法正常集成请添加如下配置：

allprojects {

repositories {

mavenCentral()

}

}

## 

## 1.2添加权限(AndroidManifest.xml内)

<!-- 连接互联网Internet权限 -->

<uses-permission android:name="android.permission.INTERNET" />

<!-- 允许应用程序联网，以便向我们的服务器端发送数据。 -->

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />

<!--判断程序是否在前台运行,必须 -->

<uses-permission android:name="android.permission.GET\_TASKS" />

<!-- 检测手机基本状态 -->

<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />

<!-- 缓存资源优先存入SDcard -->

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.MOUNT\_UNMOUNT\_FILESYSTEMS" />

## 1.3添加Merculet AppKey信息

### 1.3.1后台获取merculet Appkey等信息

登录后台管理（[http://merculet.cn/](http://merculet.cn/%20) ）。进入“产品管理”菜单，填写相应内容创建应用，并获取merculet App Key. （对应移动端AndroidManifest.xml内的M\_APP\_KEY）

### 1.3.2代码内添加(AndroidManifest.xml内)

<!-- 此处跟activity同级，需要放在Application内，AppKey替换为后台的”App Key”值-->

<meta-data

android:name="M\_APP\_KEY"

android:value="你的AppKey" />

<meta-data

android:name="ACCOUNT\_KEY"

android:value="你的account key" />

<meta-data

android:name="ACCOUNT\_SECRET"

android:value="你的account secret " />

# 二．基本功能集成（必加项）

## 2.1初始化SDK

注意: 如需要使用AndroidManifest.xml中配置好的appkey,account\_key,account\_secret 值，UMConfigure.init调用中appkey,account\_key,account\_secret 参数请置为null）：

MConfiguration.init(Application context,String app\_key, String account\_key, String account\_secret)

在程序Application的onCreate函数中调用初始化接口

MConfiguration.init(this)

.setLogEnable(false)

.setPageTrackWithFragment(true);

需要将用户的一些信息，比如用户名或者手机号（注意隐私）储存到服务器，以便节省某些活动的再次输入。可利用以下接口传递

在用户登录时，调用

TrackAgent.currentEvent().setUserProfile(String userId);

在用户注销登录时，调用

TrackAgent.currentEvent().cancelUserProfile()

## 2.2 Session功能

注：Session为基础功能，必须集成，否则后台无法获取正确报告数据。

方法1**：（Android-14之后才起作用。）**

**MConfiguration.init(this)内已调用Session.setAutoSession(this)；**

如果需要兼容 Android-14 （4.0）之前的版本，请用以方法：

方法2 ：

在BaseActivity(父类activity)或者每个activity的相应函数里加入以下代码：

@Override

protected void onPause() {

Session.onPause(this);

super.onPause();

}

@Override

protected void onResume() {

Session.onResume(this);

super.onResume();

}

## 2.3自定义事件功能

自定义事件的注册（配置）包括事件id的注册和事件，id下参数信息的注册。

### 2.3.2统计场景举例

1. 时间点事件

统计发生次数

// eventId事件标识

TrackAgent.currentEvent().event("您自定义的event id");

统计带属性的行为触发次数

// eventId事件标识

// properties事件参数的键值对

TrackAgent.currentEvent().event("您自定义的event id", Map<String,String>);

代码示例：：

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

map.put("invitationCode","2098");

map.put("amount","300");

map.put("sign in","300"); //登录

TrackAgent.currentEvent().event ("您自定义的event id", map);

（2）时间段事件

代码示例：

// eventId事件标识

// properties 事件参数的键值对

//调用位置：事件开始时调用

TrackAgent.currentEvent().event Start(String eventId );

//调用位置：事件结束后调用

TrackAgent.currentEvent().event (String eventId, Map<String,String> properties);

注意：

① 如果调用Process.kill或者System.exit之类的方法杀死进程，请务必在此之前调用Session.onKillProcess()方法，用来保证Session正确。

②如果需要混淆代码，为保证sdk 的正常使用，需要在proguard.cfg\_加上如下配置：

-keep class io.merculet.\*\* {\*;}

-dontwarn io.merculet.\*\*

-keep class io.merculet.\*\* {\*;}

-keep public enum

# 三．测试与调试

## 4.1测试前的准备

* 确认所需的权限都已经添加：INTERNET, READPHONESTATE等
* 确认MW\_APPID与WECHAT\_APPID已经正确的写入Androidmanifest.xml
* 确认MWActivity在Androidmanifest.xml里正确声明
* 确认Session 正确集成
* 确认测试手机(或者模拟器)已成功连入网络

## 4.2测试流程

使用普通测试流程，请先在程序入口初始化MConfiguration添加以下代码打开调试模式：

config.setDebugMode( true );

打开调试模式后，您可以在logcat中查看您的数据是否成功发送到merculet服务器，以及集成过程中的出错原因等。

|  |  |  |
| --- | --- | --- |
| Log的tag | 用途 | 结果 |
| SDKDebug | 其他log | 系统调试的一些其他log |

注意：请使用普通测试流程，您的测试数据会与用户的真实使用数据同时处理，从而导致一定的数据污染。