# SDK使用概述

## 目标和范围

本说明书仅仅适用于Android应用开发者，其它的应用请参考相关文档。

## 文档和SDK包的获取

首先是公司或个人来cally网站上注册一个用户，利用该用户信息可以管理和查看应用的使用情况。然后再申请一个AppID，开通一个网站或一个应用时就要申请一个AppID，一个用户可以申请多个AppID。

SDK包和文档请从cally网站上进行下载。

## ANDROIDMANIFEST.XML的修改

需要增加网络访问和检查网络连接状态的权限，如果你的应用中已经添加这些权限，请忽略。

添加AppID和ChannelID的信息。其中AppID必须添加，其值就是从cally网站申请的AppID。ChannelID是方便您统计和查看渠道来源，方便您按渠道统计应用的使用情况而设计的。如果你的应用没有渠道的统计需求，请忽略。

参考实例如下：

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.testcally.testuploadtrack"

android:versionCode="1"

android:versionName="1.0" >

......

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

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

<application

......

**<meta-dataandroid:name="CALLY\_YOUR\_APP\_ID" android:value="CALLY8888888" />**

**<meta-data android:name="CALLY\_YOUR\_CHANNEL\_ID" android:value="CALLY666666" />**

......

</application>

</manifest>

## SDK包的配置

### 使用Eclipse ADT开发

下载Android SDK包解压至本地目录下，将其中的webcally.jar拷贝到工程的libs文件夹下即可。

### 使用Android Studio开发

下载Android SDK包解压至本地目录下，将其中的webcally.jar拷贝到工程的libs文件夹下。在 app 目录下的build.gradle文件中的 dependencies 依赖项，添加jar包的依赖。参考实例如下：

dependencies {

…

compile files('libs/webcally.jar')

…

}

# SDK API介绍

目前支持的API函数如下表：

|  |  |  |
| --- | --- | --- |
| **编号** | **函数名称** | **描述信息** |
| 1 | initContext() | 初始化信息 |
| 2 | enableDebugLogging() | 是否打开调试日志开关，默认处于关闭状态 |
| 3 | onClick() | 事件点击时记录用户的状态 |
| 4 | onEvent() | 光标的移动，页面上移或下滑，Tab页面切换等事件信息的记录。 |
| 5 | onException() | 当应用发生异常时，记录用户的异常堆栈信息 |
| 6 | onActivityStart() | 界面启动时记录界面的状态 |
| 7 | onActivityStop() | 界面停止时记录界面的状态 |
| 8 | onFragmentStart() | 界面组件启动时记录界面的状态 |
| 9 | onFragmentStop() | 界面组件停止时记录界面的状态 |
| 10 | onPageStart () | 利用Webview组件开发网页时记录网页的状态 |
| 11 | onPageStop() | 利用Webview组件关闭网页时记录网页的状态 |

## API函数介绍

### 初始化信息

**函数定义：**

voidinitContext(final Context context)

**输入参数：**

context：应用的上下文。

**输出参数：**

无。

**备注信息：**

使用该SDK时，必须先调用该函数进行初始化信息。需要在应用的onCreate中调用。

### 是否打开调试日志开关

**函数定义：**

voidenableDebugLogging(final booleanenableLogging)

**输入参数：**

enableLogging：是否打开调试日志的开关。

**输出参数：**

无。

**备注信息：**

是否打开调试日志的开关，如果enableLogging为true时，打开SDK的调试日志。默认关闭。

### 记录用户的点击状态

**函数定义：**

voidonClick(final Context context, final View view, final String extValue)

**输入参数：**

context：应用的上下文，当前的Activity。

view：当前点击的控件。

extValue：当前点击控件的值或扩展的值，如果不需要统计扩展的值也用使用null。

**输出参数：**

无。

**备注信息：**

记录用户的点击事件，在需要统计控件的点击事件中增加该函数。

### 记录用户的事件信息

**函数定义：**

voidonEvent(final Context context, final View view, final String eventName, final String eventValue, final String extValue)

**输入参数：**

context：应用的上下文，当前的Activity。

view：当前选中的控件。

eventName：当前事件的标签或名称，必须在当前界面中唯一确定。

eventValue：当前事件的值。

extValue：当前选中控件的值或扩展的值，如果不需要统计扩展的值也用使用null。

**输出参数：**

无。

**备注信息：**

记录用户光标的移动，页面上移或下滑，Tab页面切换等事件信息的记录。

### 记录用户的异常信息

**函数定义：**

voidonException(final Context context, final Exception ex)

**输入参数：**

context：应用的上下文，当前的Activity。

ex：当发生异常时，异常信息的内容。

**输出参数：**

无。

**备注信息：**

记录用户的异常信息。在可能发生的地方增加try catch语句，在catch语句中增加记录用户的异常信息。

### 记录用户启动界面的信息

**函数定义：**

voidonActivityStart(final Context context)

**输入参数：**

context：应用的上下文，当前的Activity。

**输出参数：**

无。

**备注信息：**

记录用户启动的界面信息。在Activity的OnStart函数中增加该函数。

### 记录用户关闭或隐藏界面的信息

**函数定义：**

voidonActivityStop(final Context context)

**输入参数：**

context：应用的上下文，当前的Activity。

**输出参数：**

无。

**备注信息：**

记录用户关闭或隐藏的界面信息。在Activity的OnStop函数中增加该函数。

### 记录用户启动fragment的信息

**函数定义：**

voidonFragmentStart(final Fragmentfragment)

**输入参数：**

fragment：应用的fragment，当前的fragment。

**输出参数：**

无。

**备注信息：**

记录用户启动的fragment信息。在Fragment的OnStart或OnCreate函数中增加该函数。

### 记录用户关闭或隐藏fragment的信息

**函数定义：**

voidonFragmentStop(final Fragmentfragment)

**输入参数：**

fragment：应用的fragment，当前的Fragment。

**输出参数：**

无。

**备注信息：**

记录用户关闭或隐藏的fragment信息。在Fragment的OnStop或OnDestroy函数中增加该函数。

### 记录用户利用WebView打开网页的信息

**函数定义：**

voidonPageStart(final WebViewwebview, final String pageTitle)

**输入参数：**

webview：应用的webview，当前的Webview。

pageTitle：当前网页的标题

**输出参数：**

无。

**备注信息：**

记录用户利用Webview打开网页的信息。在Webview的onPageStarted函数中增加该函数。

### 记录用户利用WebView关闭网页的信息

**函数定义：**

voidonPageStop(final WebViewwebview, final String pageTitle)

**输入参数：**

webview：应用的webview，当前的Webview。

pageTitle：当前网页的标题

**输出参数：**

无。

**备注信息：**

记录用户关闭当前网页时记录信息。在Webview的onPageFinished函数中增加该函数。

**参考事例：**

classMyAndroidWebViewClient extends WebViewClient {

@Override

public void onPageStarted(WebView view, String url, Bitmap favicon) {

// To Do …….

onPageStart(view, view.getTille());

}

@Override

public void onPageFinished(WebView view, String url) {

// To Do…….

onPageStop(view, view.getTille());

}

}

webview.setWebViewClient(new MyAndroidWebViewClient ());

# 常见问题解答

## 如何验证集成SDK是否正确

集成后请进行必要的真机测试，登录网站统计后台进行查阅数据，数据查询和你测试的数据一致后，才能说明集成正确。测试中请注意。

如果有问题，请依次检查AndroidManifest.xml里必要的权限是否已正确添加，确保App ID添加正确，没有输入多余的空格等难发现的字符。

如果你设置enableDebugLogging为true, 你可以看到事件的log信息。

## 如何避免混淆对统计SDK的影响

有些开发者对打了统计SDK的程序进行混淆时，有可能会覆盖某些java类，导致统计SDK无法正常工作，解决方法如下：

请开发者在混淆配置文件proguard.cfg或proguard-project.txt的最后加上-keep public class com.cally.CallyTrack.\*\* { public protected \*;}，使混淆的时候不会影响统计SDK的命名空间。

## 打开和关闭界面统计的时间不准确

请检查onActivityStart()和onActivityStop()是否成对出现，如果不成对出现统计时可能会出现统计不准确的问题。同时，onFragmentStart()和onFragmentStop()也一定成对出现。