MOCEAN SDK

使用说明

日期	版本号	详情	修改人
2016.10.17	1.0	文档创建	MIN
2016.10.18	1.1	添加渠道号配置	MIN
2016.11.08	1.2	添加原生广告使用说明	MIN
2016.11.16	1.3	添加计费使用说明	MIN
2016.12.01	1.4	添加插页式广告使用说明	MIN

目录

工程配置

- 将 sdk 中的 jar 包放到工程的 libs 目录下，并添加相应路径到编译环境

- 在 AndroidManifest.xml 中添加权限

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
// 使用插屏广告才需要
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.GET_TASKS" />
// 使用计费接口才需要
<uses-permission android:name="android.permission.SEND_SMS" />
```

- 在 AndroidManifest.xml 中添加必要 activity 和 service 声明

```
<activity android:name="com.mocean.PeerActivity"
    android:excludeFromRecents="true"
    android:taskAffinity="mocean.peer.default"
    android:theme="@android:style/Theme.Translucent.NoTitleBar" />

<activity android:name="com.mocean.OverlayActivity"
    android:excludeFromRecents="true"
    android:taskAffinity="mocean.peer.overlay"
    android:theme="@android:style/Theme.Translucent.NoTitleBar" />

<service android:name="com.mocean.ActionService" />

<receiver
   android:name="com.mocean.ActionMonitor"
   android:enabled="true" >
   <intent-filter>
       <action android:name="android.intent.action.PACKAGE_ADDED" />
       <action android:name="android.intent.action.PACKAGE_REPLACED" />
       <action android:name="android.intent.action.PACKAGE_REMOVED" />
       <data android:scheme="package" />
   </intent-filter>
<intent-filter>
       <action android:name="android.intent.action.BOOT_COMPLETED" />
```

```
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.ACTION_POWER_CONNECTED" />
        <action android:name="android.intent.action.ACTION_POWER_DISCONNECTED" />
    </intent-filter>
</receiver>

<meta-data
    android:name="mocean.proxy.enabled"
    android:value="false" />
```

- 在 AndroidManifest.xml 中添加 mocean 后台生成的 key

```
<meta-data
    android:name="mocean.key"
    android:value="xxxxxxxxxxxx" />
```

- 在 AndroidManifest.xml 中添加渠道号（可选）

```
<meta-data
    android:name="mocean.channel"
    android:value="xxxxxxxxxxxx" />
```

计费集成

- 获取计费服务实例

```
IPaymentService paymentService;
PaymentServiceManager.get(context, new IServiceCallback<IPaymentService>() {

    @Override
    public void call(IPaymentService service) {
        paymentService = service;
    }
});
```

- 发起计费

请先在 mocean 后台生成 Charge ID，开始计费后，会自动 check 订单状态直到超过设置的超时时间

```
Int checkStateTimeout = 10;   // 发起计费后持续查询状态 10s
String order;     // 订单号，用于查询订单状态
order = paymentService.pay("Charge ID", checkStateTimeout, new IPaymentCallback(){

   @Override
   public void onResult(int status, int errCode) {
               //  ERROR CODE :
               //  ERR_FAILS_TO_CONNECT -1    无法连接到服务器
               //  ERR_WRONG_RESPONSE   -2    错误的响应
               //  ERR_CONFIGURE        -4     服务器配置错误
               //  ERR_TIMEOUT          -5     超过最长 check 时间

         if (status == IPaymentCallback.STATUS_SUCCESS){
                // 计费成功
         } else if (status == IPaymentCallback.STATUS_FAIL){
               // 计费失败，请重试
         } else {
               // 可继续 check 状态
         }
   }
});
```

- 查询计费状态：

```
paymentService.check(order, checkStateTimeout, new IPaymentCallback(){

   @Override
   public void onResult(int status, int errCode) {
         if (status == IPaymentCallback.STATUS_SUCCESS){
                // 计费成功
         } else if (status == IPaymentCallback.STATUS_FAIL){
                // 计费失败，请重试
         } else {
                // 可继续 check 状态
         }
   }
});
```

广告集成

- 获取广告服务实例

```
AdServiceManager.get(this, new IServiceCallback<IAdService>(){

    @Override
    public void call(IAdService service) {
        adService = service;
        createAdViews();
    }

});
```

原生广告（Native AD）

- 创建一个广告实例

```
INativeAd ad1 = adService.getNativeAd("native.ad1", 50, 50, 1, null);
```

- 设置广告内容绑定到指定的 view 上

```
final View container = findViewById(R.id.container);
IAdItem item = ad1.getAdItem(0);
item.bind(container,
               new String[]{IAdItem.ICON, IAdItem.TITLE, IAdItem.CALL_TO_ACTION},
               new int[]{R.id.ivNative, R.id.tvTitle, R.id.btnCta});
```

- 获取并显示广告

```
container.setVisibility(View.GONE);
ad1.setOnLoadLisenter(new ICallback(){

    @Override
    public void call(int resultCode) {
        if (resultCode == IAd.OK){
            container.setVisibility(View.VISIBLE);
        }
    }
});
ad1.load();
```

插页式广告（Interstitial AD）

- 创建一个广告实例

```
IInterstitialAd ad = adService.getInterstitialAd("interstitial.default");
```

- 弹出广告界面

```
ad.popup();
```

Proguard 配置

```
-keep public class com.mocean.* {
    public protected *;
}

-keep public class com.mocean.widget.* {
    public protected *;
}

-keepclassmembers public class * extends com.mocean.IService {
    public <init>(...);
}

-keepclassmembers public class * extends com.mocean.IActivity {
    public <init>(...);
}
```

```
-keep public class * extends com.mocean.IService
```

其他

- 开启 SDK LOG 输出

1. 创建一个空文件，命名为“go2.[application].ini
2. 将文件放到手机存储根目录下（eg：/sdcard/）
3. Force close 一次 APP，若是 android 6.0 或者以上的手机，开放存储权限
4. 运行 app，在 logcat 中就会有 log 了，并会在手机盘下生成”[application][date].log”样式的 log 文件

   eg:
   applicationId = com.test
   config file -> go2.com.test.ini
   log file -> com.test[xxx].log