# 信天翁SSP Android SDK 接入文档

# 一、概述

本文档旨在帮助 Android 开发者快速高效的接入信天翁平台(信天翁SSP平台，以下简称信天翁SSP)　SDK。作为开发者，您只需要进行简单配置，就可以在您的应用中集成广告模块、电商优惠券模块、闲 玩模块。关于 SDK 的具体使用方法，请仔细阅读下面的文档。

# 二、概念解释

接入SDK前，请先通过信天翁SSP申请自己产品的媒体id和广告id，首先解释以下名词：

1、信天翁SSP：信天翁广告平台。信天翁SSP已接入国内外主流Ad network、Ad Exchange、 DSP平台20余 家，对接数万级的广告主资源，覆盖37个行业(品牌、效果、 App推广等)，可以实现每一个流量充分 竞价，全面提升媒体的变现能力和收益。

2、媒体id：产品的唯一id，在填写完产品必要的相关信息后，后台会自动生成；

3、广告位id：产品里对应的广告位id，如app里面有2个横幅和一个开屏广告位，那么就需要三个独立的广告id；

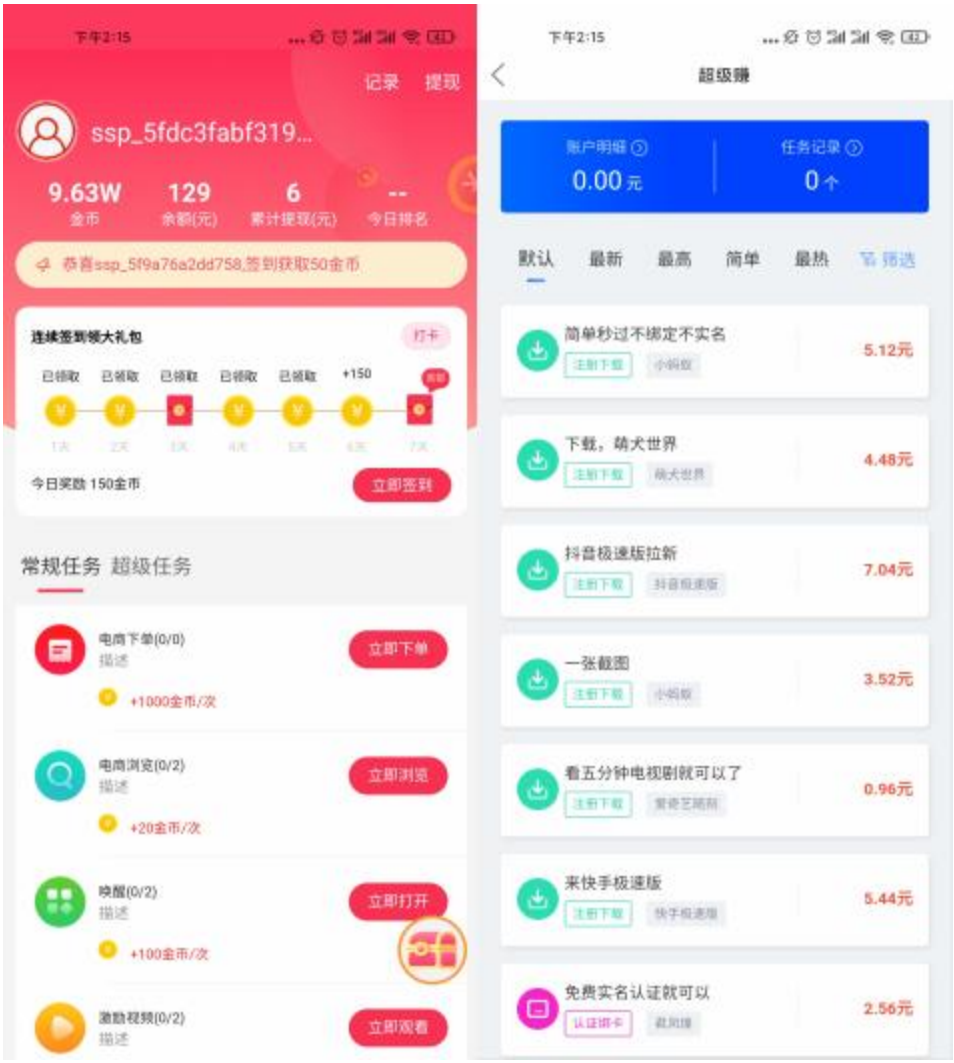
其中，平台提供了相应的测试广告位(*仅供测试使用，正式上线前请换成正式广告位*)：

| 广告类型 | 媒体**id** | 广告位**id** |
|---|---|---|
| 开屏广告 | 881 | 3561 |
| 插屏广告 | 881 | 11087 |
| 横幅广告 | 881 | 1983 |
| 模板广告 | 881 | 8461 |
| 模板视频信息流 | 881 | 8517 |
| 全屏视频 | 881 | 8811 |
| 信息流广告 | 881 | 2351 |
| 激励视频广告 | 881 | 1028 |
| 短视频内容 | 881 | - |

4、电商优惠券模块：信天翁SDK中的一个功能模块(*纯广告SDK无该模块*)，可以帮助开发者快速搭建自 己的电商模块(支持用户订单同步，用户返佣配置等功能)，丰富产品的同时，提升变现能力，增强用 户粘性。具体页面效果如下图所示：


5、闲玩模块：信天翁SDK中的一个功能模块，开发者只需要一行代码即可快速高效的接入闲玩模块完成游 戏试玩获取收益(支持用户订单同步，用户返佣配置等功能)。具体页面效果如下图所示：


6、任务模块(含超级赚)：信天翁SDK中的一个功能模块，开发者只需要一行代码即可快速高效的接入任 务模块搭建自己的任务系统，通过用户做任务获取收益(支持用户任务状态同步、用户信息打通等)。 具体页面效果如下图所示：




# 三、广告分类

信天翁SSP广告主要分为以下八种类型：

1、开屏广告：全屏展示广告，主要应用在app启动的时候显示；

2、横幅广告：横幅广告是在内容底部或顶部显示的小条形广告；

3、插屏广告：屏幕中间弹窗显示的广告；

4、模板广告：相比于横幅广告、插屏广告等，模板广告提供了更加灵活、多样化的广告样式选择；

5、模板视频信息流广告：类似抖音视频的竖版视频信息流，与视频产品完全融为一体；

6、信息流广告：用户根据请求获取的广告信息自由的展示广告，这种方式较为灵活；

7、激励视频广告：根据自己业务需求，观看完激励视频广告后对用户进行相应的奖励(支持服务端同步)；

8、短视频内容：类似抖音、快手短视频内容；

# 四、接入流程

## 1、导入aar

SDK是通过aar包的形式接入。请将aar包(ssp_android_sdk_v*.aar)导入到模块的libs文件夹下，并在该模块的build.gradle文件配置：

```
android {
    ...
   ndk { // 注：SDK默认支持的 SO 库构架如下
       abiFilters 'armeabi-v7a', 'arm64-v8a'
   }
}

//添加配置
repositories {
    flatDir {
        dirs 'libs'
    }
}

//添加依赖
dependencies {
    ...
    //使用SDK包（*** 请替换成具体的版本号，请勿使用Demo中的SDK包，注意：纯广告SDK名为ssp_ad_sdk_v*格式 ***）
    implementation (name: 'ssp_android_sdk_v*', ext: 'aar')
    //SDK依赖的三方包（如果已导入相关依赖包，无需重复导入。注意：Glide需4.0以上版本）
    //如果使用了友盟、支付宝支付，请依赖对应去UTDID版本SDK，避免UTDID冲突（使用电商模块时需注意该冲突）
    //如果三方依赖包出现重复，请以本SDK的包为准

    //如果接入了快手SDK，需要添加如下依赖（建议26以上）
    implementation "com.android.support:appcompat-v7:$support_version"
    implementation "com.android.support:recyclerview-v7:$support_version"

    //小米广告平台依赖（如果接入小米广告，需要配置如下依赖，且已该版本为准。如果不接入小米，可忽略）
    implementation 'com.google.code.gson:gson:2.8.5'
    implementation 'com.github.bumptech.glide:glide:4.9.0'
    annotationProcessor 'com.github.bumptech.glide:compiler:4.9.0'
    ...
}
```

注意事项：

如果导入SDK的模块为子模块，请在根项目的build.gradle文件做如下配置

```
allprojects {
    repositories {
        ...
        flatDir {
            dirs project(':模块名').file('libs')
        }
    }
}
```

## 2、配置权限

```
<!--存储读写权限 -->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>

<!-- 获取网络信息 -->
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>

<!-- 获取wifi信息 -->
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>

<!-- 获取手机信息 -->
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>

<!-- 连接互联网Internet权限 -->
<uses-permission android:name="android.permission.INTERNET"/>

<!-- 获取屏幕状态（不包含广告模块时可不配置） -->
<uses-permission android:name="android.permission.WAKE_LOCK"/>

<!-- 获取位置信息（可选，有助于广告收益，不包含广告模块时可不配置）-->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>

<!-- 安装APP权限，用于安装下载类广告下载的APP（不包含广告模块时可不配置） -->
<uses-permission android:name="android.permission.REQUEST_INSTALL_PACKAGES" />
```

注意事项：

为了适配Android6.0以上机型，存储读写权限、获取手机信息权限请在获取广告之前动态申请，确保获取广告之前已获取以上权限！

## 3、配置AndroidManifest文件

```
<!-- 适配Android7.0以及以上 -->
<provider
    android:name="android.support.v4.content.FileProvider"
    android:authorities="${applicationId}.SSPFileProvider"
    android:exported="false"
    android:grantUriPermissions="true">
    <meta-data
        android:name="android.support.FILE_PROVIDER_PATHS"
        android:resource="@xml/ssp_sdk_files_path" />
</provider>
```

在res/xml目录下，新建一个xml文件ssp_sdk_files_path.xml，在该文件中添加如下代码：

```
<?xml version="1.0" encoding="utf-8"?>
<paths xmlns:android="http://schemas.android.com/apk/res/android">
    <root-path name="name" path=""/>
    <external-files-path name="download" path="" />
    <external-path name="download" path="" />
    <cache-path name="download" path="" />
    <external-cache-path name="download" path="" />
    <external-path name="gdt_sdk_download_path" path="GDTDOWNLOAD" />
</paths>
```

注意事项：

1、防止Android8.0部分手机使用WebView抛SafeBrowsingResponse相关异常，请在manifest标签下添加meta-data，具体如下所示；

```
<manifest>
    <meta-data android:name="android.webkit.WebView.EnableSafeBrowsing"
               android:value="false" />
    ...
    <application> ... </application>
</manifest>
```

2、如果您想兼容Android N或者以上的设备，必须要在AndroidManifest.xml文件中配置FileProvider来访问共享路径的文件，如果你使用的第三方库也配置了同样的FileProvider,可以通过继承FileProvider类来解决合并冲突；

3、为保证targetSdkVersion>=28时正常的使用Http协议的请求网络数据，需要按要求添加如下代码：

在res文件夹下创建一个xml文件夹，然后创建一个network_security_config.xml文件，文件内容如下：

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <base-config cleartextTrafficPermitted="true" />
</network-security-config>
```

接着，在AndroidManifest.xml文件下增加以下属性：

```
<manifest ... >
    <application android:networkSecurityConfig="@xml/network_security_config"
        ... >
        ...
    </application>
</manifest>
```

4、如果您的工程使用的是AndroidX的环境，请参考官网升级AndroidX，在 gradle.properties 文件中新增如下配置：

```
# Android 插件会使用对应的 AndroidX 库而非支持库。
android.useAndroidX=true
# Android 插件会通过重写现有第三方库的二进制文件， 自动将这些库迁移为使用 AndroidX。
android.enableJetifier=true
```

并在AndroidStudio，选择Refactor ，然后选择Migrate to AndroidX；

## 4、SDK初始化

```
/**
 * 初始化（使用SDK任何模块都应先初始化）
 * 注意：在自定义Application的onCreate方法中调用该方法
 *
 * @param context 上下文
 * @param mediaId 媒体id
 * @param channelId 应用渠道id（非必填，填写后可以按渠道查看广告数据）
 * @param showLog 是否显示日志，建议正式上线时，设置为false， 日志TAG为ssp_android_sdk
 */
SSPSdk.init(context, mediaId, showLog);
SSPSdk.init(context, mediaId, channelId, showLog);

//其他配置（请务必将配置设置在SDK初始化之前）
//设置下载确认策略（设置为不确认，默认为DownloadConfirmPolicy.CONFIRM_DEFAULT:非WiFi下提示确认）
SSPSdk.setDownloadConfirmPolicy(DownloadConfirmPolicy.CONFIRM_NONE);
//设置是否初始化时就申请权限
SSPSdk.setReqPermission(false);
```

## 5、广告模块接入

开屏广告

```
//初始化广告客户端实例
AdClient adClient = new AdClient(activity);

/**
 * 请求开屏广告
 *
 * @param adContainer   展示开屏广告的容器控件
 * @param adId          广告ID
 * @param listener      广告回调适配器（见说明）
 */
adClient.requestSplashAd(adContainer, adId, new AdLoadAdapter() {
        @Override
        public void onError(String error) {
            super.onError(error);
            //获取广告失败，跳转主页
            gotoMainActivity();
        }

        @Override
        public void onAdDismiss(SSPAd ad) {
            super.onAdDismiss(ad);
            //广告关闭（开屏广告展示时间到或用户点击跳转），跳转主页
            gotoMainActivity();
        }
 });


//开屏最优处理方式：
    //1、onAdDismiss后处理页面跳转；
```

```
    //2、广告点击或切到后台后返回开屏页面时处理跳转（广告点击跳转页面后返回开屏时倒计时会结束，需要自己处理跳转逻辑）；
    //3、最好自己做一下超时处理，比如6s还未请求到广告（未回调onAdLoad）就直接跳转页面（避免网络不好情况或其他异常情况长时间停留在开屏页面）；
```

如果需要接入华为开屏广告，在开屏页面Activity布局文件的开屏广告容器中需按如下要求进行配置(不接入可忽略该配置)：

```
<!--开屏广告展示容器-->
<FrameLayout
    android:id="@+id/splash_ad_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_above="@id/custom_layout">

    <!--华为开屏广告控件（若不接入华为开屏广告，可省略）-->
    <com.huawei.hms.ads.splash.SplashView
        android:id="@+id/hw_splash_view_id"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</FrameLayout>
```

插屏广告

```
//初始化广告客户端实例
AdClient adClient = new AdClient(activity);

/**
 * 请求插屏广告
 *
 * @param adId        广告ID
 * @param listener    广告回调适配器（见说明）
 */
adClient.requestInteractionAd(adId, new AdLoadAdapter() {
        @Override
        public void onAdShow(SSPAd sspAd) {
            super.onAdShow(sspAd);
            //插屏广告显示
        }
 });
```

横幅广告

```
//初始化广告客户端实例
AdClient adClient = new AdClient(activity);

/**
 * 请求横幅广告
 *
 * @param adContainer    展示Banner广告的容器控件（请求广告成功之后，Banner广告会自动添加到该容器中，不需要开发者手动添加）
 * @param adId        广告ID
 * @param listener    广告回调适配器（见说明）
 */
```

```
adClient.requestBannerAd(adContainer, adId, new AdLoadAdapter() {
        @Override
        public void onAdLoad(SSPAd ad) {
             super.onAdLoad(ad);
        }
 });
```

模板广告

```
//初始化广告客户端实例
AdClient adClient = new AdClient(activity);

/**
 * 请求模板广告
 *
 * @param adId        广告ID
 * @param listener    广告回调适配器（见说明）
 */
adClient.requestExpressAd(adId, new AdLoadAdapter() {
        @Override
        public void onAdLoad(SSPAd ad) {
             super.onAdLoad(ad);
             //获取到模板广告，将其添加到显示的容器控件中
             contentLayout.removeAllViews();
             contentLayout.addView(ad.getView());
        }
 });
```

模板视频信息流广告

```
//初始化广告客户端实例
AdClient adClient = new AdClient(activity);

/**
 * 请求模板视频信息流广告
 *
 * @param adId        广告ID
 * @param listener    广告回调适配器（见说明）
 */
adClient.requestExpressDrawFeedAd(adId, new AdLoadAdapter() {
        @Override
        public void onAdLoad(SSPAd ad) {
             super.onAdLoad(ad);
             //获取到模板视频信息流广告，将其添加到显示的容器控件中
             contentLayout.removeAllViews();
             contentLayout.addView(ad.getView());
        }
 });
```

全屏视频

```
//初始化广告客户端实例
AdClient adClient = new AdClient(activity);

/**
 * 请求模板视频信息流广告
 *
 * @param adId        广告ID
 * @param listener    广告回调适配器（见说明）
 */
adClient.requestFullScreenVideoAd(adId, new AdLoadAdapter() {
        @Override
        public void onAdLoad(SSPAd ad) {
             super.onAdLoad(ad);
        }
 });
```

信息流广告

```
//初始化广告客户端实例
AdClient adClient = new AdClient(activity);

/**
 * 请求信息流广告
 *
 * @param adId        广告ID
 * @param listener    广告回调适配器（见说明）
 */
adClient.requestFeedAd(adId, new AdLoadAdapter() {
        @Override
        public void onAdLoad(SSPAd ad) {
             super.onAdLoad(ad);
             //获取到信息流广告信息，自己渲染展示
             Glide.with(feedAdView).load(ad.getImg()).into(feedAdView);
             //注册展示信息流广告的控件（***** 非常重要，影响收益 *****）
             adClient.registerView(feedAdLayout, ad.getAdInfo(), new
AdLoadAdapter());
        }
 });
```

当为信息流广告时，用户请求广告成功之后，需要按照自己的需求展示广告，并将展示广告的控件注册，否则会影响数据统计及计费。

```
/**
 * 信息流广告注册广告控件
 *
 * @param feedAdLayout    展示广告的布局
 * @param adInfo          广告详情信息
 * @param listener        广告回调（见说明）
 */
client.registerView(feedAdLayout, adInfo, new AdLoadAdapter() {
      @Override
      public void onAdClick() {
           super.onAdClick();
      }
 });
```

激励视频广告

```
//初始化广告客户端实例
AdClient adClient = new AdClient(activity);

/**
 * 请求激励视频广告
 * <p>如需激励视频服务端同步，请在SDK初始化后调用SSPSdk.login授权</p>
 *
 * @param adId        广告ID
 * @param listener    激励视频广告回调适配器（见说明）
 */
adClient.requestRewardAd(adId, new RewardVideoAdAdapter() {
    @Override
    public void onReward(int type) {
         super.onReward(type);
         //激励，可根据自己业务逻辑对用户进行奖励
    }
 });
```

短视频

```
//方式一：打开默认短视频页面（接入简单，只需要提供入口，快速接入）
SSPSdk.openContentPage();

//方式二：Fragment方式接入（自定义页面样式，根据自己实际业务需求添加监听，具体参照Demo）
//获取短视频内容Fragment页面
 Fragment contentFragment = SSPSdk.getContent();
//加载短视频内容页面
getSupportFragmentManager().beginTransaction().replace(R.id.content_layout,
contentFragment).commitAllowingStateLoss();
//增加内容页面监听器
SSPSdk.setContentPageListener(new IContentPageListener() {
    @Override
    public void onPageEnter(SSPContentItem sspContentItem) {
        //页面进入
    }

    @Override
    public void onPageResume(SSPContentItem sspContentItem) {
```

```
        //页面恢复可见
    }

    @Override
    public void onPagePause(SSPContentItem sspContentItem) {
        //页面不可见
    }

    @Override
    public void onPageLeave(SSPContentItem sspContentItem) {
        //页面离开
    }
});
//增加内容视频监听器
SSPSdk.setContentVideoListener(new IContentVideoListener() {
    @Override
    public void onVideoPlayStart(SSPContentItem sspContentItem) {
        //播放开始
    }

    @Override
    public void onVideoPlayPaused(SSPContentItem sspContentItem) {
        //播放暂停
    }

    @Override
    public void onVideoPlayResume(SSPContentItem sspContentItem) {
        //恢复播放
    }

    @Override
    public void onVideoPlayCompleted(SSPContentItem sspContentItem) {
        //播放完成
    }

    @Override
    public void onVideoPlayError(SSPContentItem sspContentItem, int what, int
extra) {
        //播放出错
    }
});
```

短剧

```
//方式一：打开默认短剧页面（接入简单，只需要提供入口，快速接入）
SSPSdk.openShortPlayPage();

//方式二：Fragment方式接入（自定义页面样式，根据自己实际业务需求添加监听，具体参照Demo）
//加载短剧页面
Fragment shortPlayFragment = SSPSdk.getShortPlay();
if (shortPlayFragment == null) {
    return;
}
getSupportFragmentManager().beginTransaction().replace(R.id.content_layout,
shortPlayFragment).commitAllowingStateLoss();
```

```
//增加短剧页面监听器
SSPSdk.setShortPlayPageListener(new IContentPageListener() {
            @Override
            public void onPageEnter(SSPContentItem sspContentItem) {
                //页面进入
            }

            @Override
            public void onPageResume(SSPContentItem sspContentItem) {
                //页面可见
            }

            @Override
            public void onPagePause(SSPContentItem sspContentItem) {
                //页面不可见
            }

            @Override
            public void onPageLeave(SSPContentItem sspContentItem) {
                //页面离开
            }
        });
//增加短剧视频监听器
SSPSdk.setShortPlayVideoListener(new IContentVideoListener() {
            @Override
            public void onVideoPlayStart(SSPContentItem sspContentItem) {
                //播放开始
            }

            @Override
            public void onVideoPlayPaused(SSPContentItem sspContentItem) {
                //播放暂停
            }

            @Override
            public void onVideoPlayResume(SSPContentItem sspContentItem) {
                //恢复播放
            }

            @Override
            public void onVideoPlayCompleted(SSPContentItem sspContentItem) {
                //播放完成
            }

            @Override
            public void onVideoPlayError(SSPContentItem sspContentItem, int i,
int i1) {
                //播放出错
            }
});
```

全局配置

```
/**
* 设置是否动态请求权限（默认开启，建议开启提高广告收益）
**/
SSPSdk.setReqPermission(true);
```

资源释放

```
//为了更好的释放资源，在页面销毁时调用release方法释放资源
 @Override
 protected void onDestroy() {
     super.onDestroy();
     adClient.release();
 }
```

## 6、电商优惠券模块接入

电商优惠券模块提供了两种接入方式： Activity方式接入、 Fragment方式接入，开发者可以根据自己的业务需求选择其中一种方式接入。

授权登录信天翁**SSP**平台

```
 /**
 * 渠道用户授权登录，使用电商模块(闲玩模块、激励视频服务端同步)之前，请务必先调用该方法授权登录信天翁SSP
 * 注意：如果渠道用户无用户id（比如未登录），则不要调用授权登录；
 *
 * @param tuid      渠道用户id(接入方app用户唯一标识)
 * @param nick      渠道用户昵称(非必填，建议填写)
 * @param headImg   渠道用户头像地址(非必填，建议填写)
 * @param callback 信天翁授权登录回调
 */
 SSPSdk.login(tuid,nick,headImg,new LoginCallback() {
    @Override
    public void suc(String uid) {
        //uid为信天翁SSP平台用户id
        LogUtils.d("login suc: uid=" + uid);
    }

    @Override
    public void fail(String msg) {
        //msg为授权登录错误信息
        LogUtils.e("login fail: msg=" + msg);
    }
 });
```

设置监听器

```
/**
* MallCouponAdapter:电商优惠券模块回调适配器，实现了MallCouponCallback接口，详情见说明。
*/
SSPSdk.setMallCouponCallback(new MallCouponAdapter());
```

接入电商优惠券模块

```
//Activity方式接入，适用于单独打开页面展示电商优惠券模块
startActivity(new Intent(this, SSPCouponActivity.class));

//Fragment方式接入，灵活，开发者按需集成SSPCouponFragment
<fragment
    android:id="@+id/mallCouponFragment"
    android:name="com.youxiao.ssp.coupon.fragment.SSPCouponFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

其他配置

```
/**
* 设置电商首页是否使用沉浸式（即首页是否设置状态栏高度padding）
* @param setStatusBarHeight 首页是否设置沉浸式（默认为true）
**/
SSPSdk.setSetStatusBarHeight(setStatusBarHeight);

/**
* 配置浏览商品倒计时，默认为5秒，表示5秒内每隔一秒会回调一次浏览商品回调，设置为0表示仅进入商品详情页时回一次
* @param countDown 倒计时时长，单位秒。countDown=0表示仅进入商品详情页时浏览商品回调只调用一次
**/
SSPSdk.setScanGoodsDuration(countDown);
```

电商优惠券订单同步

用户使用电商优惠券下单后或用户订单状态改变后，信天翁平台会将用户订单同步到信天翁后台配置的服务 器地址，其同步数据格式见《服务端对接文档》；

## 7、任务模块接入

信天翁任务模块旨在帮助渠道应用快速建立任务体系，通过任务的方式提升收益及用户活跃。任务模块提 供了两种接入方式： Activity方式接入、 Fragment方式接入，开发者可以根据自己的业务需求选择其中 一种方式接入。

授权登录信天翁**SSP**平台

```
 /**
 * 渠道用户授权登录，使用任务模块(电商模块、超级赚模块、闲玩模块、激励视频服务端同步)之前，请务必先调用该方法授权登录信天翁SSP
 *
 * @param tuid      渠道用户id(接入方app用户唯一标识)
 * @param nick      渠道用户昵称(非必填，建议填写，任务模块展示)
 * @param headImg   渠道用户头像地址(非必填，建议填写，任务模块展示)
 * @param callback 信天翁授权登录回调
 */
SSPSdk.login(tuid,nick,headImg,loginCallback);
```

接入任务模块

```
//Activity方式接入，适用于单独打开页面展示任务模块
startActivity(new Intent(this, SSPTaskActivity.class));

//Fragment方式接入，灵活，开发者按需集成SSPTaskFragment
<fragment
    android:id="@+id/taskFragment"
    android:name="com.youxiao.ssp.fragment.SSPTaskFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

任务数据同步

任务模块支持与渠道方产品金币体系打通，用户在任务模块完成任务后，信天翁平台会将用户完成任务及 奖励信息同步到渠道方提供的服务器回调地址，其同步数据格式见《服务端对接文档》；

## 8、超级赚模块接入

信天翁超级赚模块旨在帮助渠道应用快速建立众包任务体系，通过众包任务的方式获取高额佣金及提升用 户活跃。超级赚模块提供了两种接入方式： Activity方式接入、 Fragment方式接入，开发者可以根据自 己的业务需求选择其中一种方式接入。

授权登录信天翁**SSP**平台

```
 /**
 * 渠道用户授权登录，使用超级赚模块(电商模块、任务模块、闲玩模块、激励视频服务端同步)之前，请务必先调用该方法授权登录信天翁SSP
 *
 * @param tuid      渠道用户id(接入方app用户唯一标识)
 * @param nick      渠道用户昵称(非必填，建议填写，任务模块展示)
 * @param headImg   渠道用户头像地址(非必填，建议填写，任务模块展示)
 * @param callback 信天翁授权登录回调
 */
SSPSdk.login(tuid,nick,headImg,loginCallback);
```

接入超级赚模块

```
//Activity方式接入，适用于单独打开页面展示超级赚模块
startActivity(new Intent(this, SSPSuperTaskActivity.class));

//Fragment方式接入，灵活，开发者按需集成SSPSuperTaskFragment
<fragment
    android:id="@+id/superTaskFragment"
    android:name="com.youxiao.ssp.fragment.SSPSuperTaskFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

超级赚数据同步

超级赚模块支持与渠道方产品金币体系打通，用户在超级赚模块完成任务后，信天翁平台会将用户完成任 务及奖励信息同步到渠道方提供的服务器回调地址，其同步数据格式见《服务端对接文档》；

## 9、闲玩模块接入

闲玩模块提供了两种接入方式： Activity方式接入、 Fragment方式接入，开发者可以根据自己的业务需求选择其中一种方式接入。

授权登录信天翁**SSP**平台

```
 /**
 * 渠道用户授权登录，使用闲玩模块(电商模块、激励视频服务端同步)之前，请务必先调用该方法授权登录信天翁SSP（详情见上面电商优惠券模块接入的授权登录）
 *
 * @param tuid      渠道用户id(接入方app用户唯一标识)
 * @param nick      渠道用户昵称(非必填，建议填写)
 * @param callback 信天翁授权登录回调
 */
SSPSdk.login(tuid,nick,loginCallback);
```

接入闲玩模块

```
//Activity方式接入，适用于单独打开页面展示闲玩模块
startActivity(new Intent(this, SSPGameActivity.class));

//Fragment方式接入，灵活，开发者按需集成SSPGameFragment
<fragment
    android:id="@+id/gameFragment"
    android:name="com.youxiao.ssp.fragment.SSPGameFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

闲玩订单同步

用户在闲玩完成试玩任务后，信天翁平台会将用户订单同步到信天翁后台配置的服务器地址，其同步数据格 式见《服务端对接文档》；

## 10、说明

1、广告信息具体如下：

```
/**
 *请求广告回调中的广告信息
 */
public class SSPAd {

    //广告id
    private String adId;
    //广告标题
    private String title;
    //广告描述
    private String des;
    //图片
    private String img;
    //图片列表
    private List<String> imgList;
    //图标
    private String icon;
```

```
    //广告控件
    private View view;
    //广告详情数据
    private AdInfo adInfo;
    ....
}
```

2、全局配置

```
/**
* 设置下载确认策略（SDK初始化之前调用，尽量保持默认，否正可能会影响收益）
* @param downloadConfirmPolicy 下载确认策略
*
*  DownloadConfirmPolicy.CONFIRM_NONE: 不需要确认，默认值
*  DownloadConfirmPolicy.CONFIRM_DEFAULT: 非WiFi下需要确认
*  DownloadConfirmPolicy.CONFIRM_ALWAYS: 每次都确认
**/
SSPSdk.setDownloadConfirmPolicy(downloadConfirmPolicy);

/**
* 设置是否初始化时就申请权限（SDK初始化之前调用）
* @param requestPermission true:初始化SDK时就申请权限（默认值），false：初始化SDK时不申请权限
**/
SSPSdk.setReqPermission(requestPermission);

/**
* 设置全局自定义参数（支持最大长度255个字符），该自定义参数在激励视频广告服务端同步的时候透传。
* @param customData 自定义参数
**/
SSPSdk.setCustomData(customData);

/**
* 获取用户自定义扩展数据（用于设置扩展数据）
* @return ExtData 扩展数据，其具体信息如下
**/
SSPSdk.getExtData();

//扩展数据
public class ExtData {
    //应用名称
    private String appName;
    //应用描述
    private String appDesc;
    //回调地址
    private String backUrl;
    //按钮名称
    private String btnName;
    //是否开启热启动开屏功能（ViVo热启动开屏功能开关）
    private boolean enableHotSplash;
    //设置banner 刷新频率(ViVo的Banner广告使用)
    private int refreshIntervalSeconds = 30;
    //开屏slogan资源id（华为开屏使用：设置竖屏下的默认开屏图片，用于开屏广告素材展示前显示）
    private int sloganResId;
```

```
    //开屏Logo布局资源id（华为开屏使用：设置Logo区域，并指定Slogan的Logo区域隐藏策略，默认隐藏）
    private int logoLayoutResId;
    //是否允许获取用户信息（华为使用）
    private boolean enableUserInfo;
    //设置是否测试（华为使用）
    private boolean setTest;
    //是否允许移动网络下载（默认允许）
    private boolean accessMobileNetDownload = true;
    //自定义数据（用户可自己定义任何数据）
    private String customData;
}
```

3、请求广告回调适配器，用户可根据需求重写其方法，其实现如下：

```
/**
 * 广告回调适配器（不是所有方法都会回调）
 */
public AdLoadAdapter implements OnAdLoadListener {

    /**
     * 广告信息加载成功回调
     * @param ad 广告信息
     */
    @Override
    public void onAdLoad(SSPAd ad) {
        //广告加载成功了
    }

    /**
     * 广告点击回调
     * @param ad 广告信息
     */
    @Override
    public void onAdClick(SSPAd ad) {
        //广告点击了
    }

    /**
     * 广告显示回调
     * @param ad 广告信息
     */
    @Override
    public void onAdShow(SSPAd ad) {
        //广告显示了
    }

    /**
     * 广告隐藏或关闭回调
     * @param ad 广告信息
     */
    @Override
    public void onAdDismiss(SSPAd ad) {
        //广告隐藏或关闭了
    }
```

```java
/**
 * 广告开始下载回调
 * @param adId 广告id
 */
@Override
public void onStartDownload(String adId) {
    //开始下载了
}

/**
 * 下载完成回调
 * @param adId 广告id
 */
@Override
public void onDownloadCompleted(String adId) {
    //下载完成了
}

/**
 * 下载应用开始安装回调
 * @param adId 广告id
 */
@Override
public void onStartInstall(String adId) {
    //开始安装了
}

/**
 * 下载应用安装完成回调
 * @param adId 广告id
 */
@Override
public void onInstallCompleted(String adId) {
    //安装完成了
}

/**
 * 获取广告失败回调
 * @param code 错误码
 * @param error 错误信息
 */
@Override
public void onError(int code,String error) {
    //获取广告出错了
}

 /**
 * 状态回调
 *
 * @param type      类型
 * @param platform 平台，见SdkData中的平台类型
 * @param status    状态
 * @param msg       错误消息（失败时有效）
 */
```

```
    void onStatus(int type, int platform, int status, String msg);

    /**
     * 下一个广告信息
     *
     * @param nextAdInfo 下一个广告信息
     */
    void onNext(NextAdInfo nextAdInfo);
}
```

4、激励视频广告回调适配器，用户可根据需求重写其方法，其实现如下：

```
/**
 * 激励视频广告回调适配器（不是所有方法都会回调）
 */
public class RewardVideoAdAdapter implements RewardVideoAdCallback {

    /**
     * 加载激励视频广告成功回调
     * @param sspAd 广告信息
     */
    @Override
    public void loadRewardAdSuc(SSPAd sspAd) {
        //激励视频加载成功了
    }

    /**
     * 加载激励视频广告失败回调
     * @param errMsg 错误信息
     */
    @Override
    public void loadRewardAdFail(String errMsg) {
        //激励视频加载失败了
    }

    /**
     * 加载激励视频成功回调
     */
    @Override
    public void loadRewardVideoSuc() {
        //激励视频加载成功了
    }

    /**
     * 加载激励视频失败回调
     * @param what 错误码
     * @param extra 错误信息码
     */
    @Override
    public void loadRewardVideoFail(int what, int extra) {
        //激励视频加载失败了
    }

    /**
     * 开始播放激励视频回调
```

```
     */
    @Override
    public void startPlayRewardVideo() {
        //开始播放激励视频了
    }

    /**
     * 激励视频播放到一半回调
     */
    @Override
    public void playRewardVideoHalf() {
        //激励视频播放到一半了
    }

    /**
     * 激励视频播放结束回调
     */
    @Override
    public void playRewardVideoCompleted(int type) {
        //激励视频播放结束
    }

    /**
     * 激励（支持服务端同步）
     */
    @Override
    public void onReward(int type) {
        //激励（可根据自己业务逻辑对用户进行奖励）
    }

    /**
     * 点击激励视频回调
     */
    @Override
    public void rewardVideoClick() {
        //激励视频点击了
    }

    /**
     * 激励视频按钮点击回调
     */
    @Override
    public void rewardVideoButtonClick() {
        //激励视频操作按钮点击了
    }

    /**
     * 激励视频关闭回调
     */
    @Override
    public void rewardVideoClosed() {
        //激励视频关闭了
    }

     /**
```

```
     * 状态回调
     *
     * @param type      类型
     * @param platform  平台，见SdkData中的平台类型
     * @param status    状态
     * @param msg       错误消息（失败时有效）
     */
    void onStatus(int type, int platform, int status, String msg);

    /**
     * 下一个广告信息
     *
     * @param nextAdInfo 下一个广告信息
     */
    void onNext(NextAdInfo nextAdInfo);
}
```

5、电商优惠券模块监听器其实现如下：

```
/**
 * Description:电商优惠券回调
 */
public interface MallCouponCallback {
    /**
     * 淘宝授权成功
     *
     * @param session 淘宝session
     */
    void tbAuthSuc(Session session);

    /**
     * 淘宝退出登录
     *
     * @param userId   淘宝用户id
     * @param userNick 淘宝用户昵称
     */
    void tbLogout(String userId, String userNick);

    /**
     * 浏览商品
     *
     * @param platform  平台
     * @param goodsId   商品id
     * @param goodsName 商品名称
     * @param countDown 倒计时（0表示倒计时结束，可配置）
     */
    void scanGoods(int platform, String goodsId, String goodsName, int
countDown);

    /**
     * 点击领券
     *
     * @param platform  平台
     * @param goodsId   商品id
     * @param goodsName 商品名称
```

```
     */
    void receiveCoupon(int platform, String goodsId, String goodsName);

     /**
     * 搜索商品
     *
     * @param keywords 搜索关键字
     */
    void searchGoods(String keywords);

     /**
     * 分享
     *
     * @param shareData 分享数据（见说明5-电商优惠券分享数据）
     */
    void share(ShareData shareData);

     /**
     * 透传信息
     *
     * @param cmd    指令
     * @param params 参数
     * 透传指令cmd说明:
     * auth:表示未授权登录SSP平台成功，接收到该透传命令后，需要判断用户是否登录，如果已登录（有uid），则
     * 直接调用授权登录SSP，如果没有登录（没有uid），就toast提示先登录，再跳转登录页面，登录成功之后再调用
     * 授权登录SSP。
     */
    void transInfo(String cmd, String params);
}
```

6、电商优惠券分享数据如下：

```
/**
 * 分享数据信息
 */
public class ShareData {
    //分享标题（商品名称）
    private String title;
    //分享内容（淘宝商品时为淘口令）
    private String content;
    //分享链接（领券落地页地址）
    private String url;
    //分享图片地址（商品图片地址）
    private String imgUrl;
    //分享数据（具体的商品信息）
    private String data;
}

//其中data的具体信息如下
{
    "id":"商品id",
    "platform":"平台(1：淘宝；2：京东；3：拼多多)",
    "name":"商品名称",
```

```
    "img":"商品图片地址",
    "price":"商品原价",
    "couponUrl":"领券页面地址",
    "couponDiscount":"优惠券金额",
    "commission":"返给用户的佣金",
    "salesNum":"销量",
    "shareUid":"分享者SSP用户id",
    "longUrl":"领券长链地址",
    "shortUrl":"领券短链地址",
    "pwd":"淘口令（淘宝商品时有效）",
    "searchId":"拼多多search_id",
    "goodsSign":"拼多多goods_sign"
}
```

## 11、注意事项

启用混淆打包的(SDK代码被混淆后会导致广告无法展现、电商优惠券模块异常或者其它异常)，请在混淆配置文件中添加如下代码：

```
#获取OAID的SDK
-keep class XI.CA.XI.**{*;}
-keep class XI.K0.XI.**{*;}
-keep class XI.XI.K0.**{*;}
-keep class XI.xo.XI.XI.**{*;}
-keep class com.asus.msa.SupplementaryDID.**{*;}
-keep class com.asus.msa.sdid.**{*;}
-keep class com.bun.lib.**{*;}
-keep class com.bun.miitmdid.**{*;}
-keep class com.huawei.hms.ads.identifier.**{*;}
-keep class com.samsung.android.deviceidservice.**{*;}
-keep class com.zui.opendeviceidlibrary.**{*;}
-keep class org.json.**{*;}
-keep public class com.netease.nis.sdkwrapper.Utils {public <methods>;}


#基础模块
-keep class com.youxiao.ssp.base.listener.* {
    *;
}
-keep class com.youxiao.ssp.base.bean.ShareData {
    public <methods>;
}
-keepclassmembers class * extends com.youxiao.ssp.base.core.JsBridge {
    public <methods>;
}
-keep class com.youxiao.ssp.ad.core.AdClient {
    public <methods>;
}
-keep class com.youxiao.ssp.ad.listener.* {
    *;
}
-keep class com.youxiao.ssp.ad.bean.SSPAd {
    public <methods>;
}
```

```
-keep class com.youxiao.ssp.ad.bean.NextAdInfo {
    public *;
}
-keep class com.youxiao.ssp.ad.bean.SSPContentItem {
    *** get*();
}
-keep class com.youxiao.ssp.coupon.listener.* {
    *;
}
-keep class com.youxiao.ssp.base.activity.SSPWebActivity {
    public <methods>;
}
-keep class * extends com.youxiao.ssp.base.activity.SSPBaseActivity {
    public static <fields>;
    public static <methods>;
}
-keep class * extends com.youxiao.ssp.base.activity.SSPBaseFragmentActivity {
    public static <fields>;
    public static <methods>;
}
-keep class * extends android.support.v4.app.Fragment {
    public static <fields>;
    public static <methods>;
}
-keep class com.youxiao.ssp.core.SSPSdk {
    public <methods>;
}


#广告模块（*** 纯电商优惠券SDK无需添加广告混淆配置 ***）
-keepclassmembers class * extends android.app.Activity {
 public void *(android.view.View);
}
-keepclassmembers enum * {
 public static **[] values();
 public static ** valueOf(java.lang.String);
}
-keep class com.baidu.mobads.** { *; }
-keep class com.baidu.mobad.** { *; }
-keep class com.qq.e.** {
    *;
}
-keep class android.support.v4.**{
    public *;
}
-keep class android.support.v7.**{
    public *;
}
-keep class com.bytedance.pangle.** {*;}
-keep class com.bytedance.sdk.openadsdk.** { *; }
-keep class com.bykv.vk.** {*;}
-keep class com.ss.android.**{*;}
-keep class com.bytedance.android.**{*;}
-keep class com.byted.live.**{*;}
-keep class org.chromium.** { *; }
```

```
-keep class aegon.chrome.** { *; }
-keep class com.kwai.**{ *; }
-keep class com.kwad.**{ *; }
-keep class com.ksad.**{ *; }
-keep class com.kuaishou.**{ *; }
-keep class com.yxcorp.**{ *; }
-keepclasseswithmembernames class * {
 native <methods>;
}
-dontwarn com.kwai.**
-dontwarn com.kwad.**
-dontwarn com.ksad.**
-dontwarn aegon.chrome.**
-keep class com.yilan.sdk.**{
    *;
}
-dontwarn javax.annotation.**
-dontwarn sun.misc.Unsafe
-dontwarn org.conscrypt.*
-dontwarn okio.**
-keep class com.fun.** {*;}
-keep class com.funshion.** {*;}
-keep class android.support.annotation.Keep
-keep @android.support.annotation.Keep class * {
    @android.support.annotation.Keep <init>(...);
    @android.support.annotation.Keep <methods>;
    @android.support.annotation.Keep <fields>;
}
-keep class okhttp3.** {*;}
-keep class okio. {*;}
-keep class com.alibaba.fastjson.** {*;}
-keep class sun.misc.Unsafe { *; }
-dontwarn com.sigmob.**
-keep class com.sigmob.**.**{*;}
-keepattributes Signature
-keepattributes *Annotation*
-keep class com.mbridge.** {*; }
-keep interface com.mbridge.** {*; }
-keep class android.support.v4.** { *; }
-dontwarn com.mbridge.**
-keep class **.R$* { public static final int mbridge*; }


#厂商广告平台（*** 未接入厂商广告平台或纯电商优惠券SDK无需添加广告混淆配置 ***）
-keep class com.miui.** { *;}
-keep class com.xiaomi.** { *; }
-keep class * extends android.os.IInterface{ *; }

-keep class com.huawei.openalliance.ad.** { *; }
-keep class com.huawei.hms.ads.** { *; }

-keepattributes SourceFile,LineNumberTable
-dontwarn com.squareup.okhttp.**
-dontwarn okhttp3.**
-keep class com.vivo.*.** { *; }
```

```
-dontwarn com.bytedance.article.common.nativecrash.NativeCrashInit

-keep class com.androidquery.callback.** {*;}
-keep class com.ss.sys.ces.* {*;}
-keep class com.opos.** { *;}
-keep class com.heytap.msp.mobad.api.** {*;}
-keep class com.heytap.openid.** {*;}


#电商模块（*** 纯广告SDK无需添加电商混淆配置 ***）
-keepattributes Signature
-ignorewarnings
-keep class javax.ws.rs.** { *; }
-keep class com.alibaba.fastjson.** { *; }
-dontwarn com.alibaba.fastjson.**
-keep class sun.misc.Unsafe { *; }
-dontwarn sun.misc.**
-keep class com.taobao.** {*;}
-keep class com.alibaba.** {*;}
-keep class com.alipay.** {*;}
-dontwarn com.taobao.**
-dontwarn com.alibaba.**
-dontwarn com.alipay.**
-keep class com.ut.** {*;}
-dontwarn com.ut.**
-keep class com.ta.** {*;}
-dontwarn com.ta.**
-keep class org.json.** {*;}
-keep class com.ali.auth.**  {*;}
-dontwarn com.ali.auth.**
-keep class com.taobao.securityjni.** {*;}
-keep class com.taobao.wireless.security.** {*;}
-keep class com.taobao.dp.**{*;}
-keep class com.alibaba.wireless.security.**{*;}
-keep interface mtopsdk.mtop.global.init.IMtopInitTask {*;}
-keep class * implements mtopsdk.mtop.global.init.IMtopInitTask {*;}
-keep class com.kepler.jd.**{ public <fields>; public <methods>; public *; }


# gson（未使用该库可忽略）
-keepattributes Signature
-keepattributes *Annotation*
-dontwarn sun.misc.**
-keep class com.google.gson.examples.android.model.** { <fields>; }
-keep class * implements com.google.gson.TypeAdapterFactory
-keep class * implements com.google.gson.JsonSerializer
-keep class * implements com.google.gson.JsonDeserializer
-keepclassmembers,allowobfuscation class * {
  @com.google.gson.annotations.SerializedName <fields>;
}


# glide（未使用该库可忽略）
-keep public class * implements com.bumptech.glide.module.GlideModule
-keep class * extends com.bumptech.glide.module.AppGlideModule {
```

```
 <init>(...);
}
-keep public enum com.bumptech.glide.load.ImageHeaderParser$** {
  **[] $VALUES;
  public *;
}
-keep class
com.bumptech.glide.load.data.ParcelFileDescriptorRewinder$InternalRewinder {
  *** rewind();
}
```