金山广告SDK

简介

本文将指导您在Android项目中快速集成广告SDK。
SDK支持Android API 9及以上的版本。完整代码已通过示例代码项目sample-app提供。

- android support-v4

构建开发环境

- 如果是Eclipse环境，打开工作空间，复制SDK库文件sdk-ad-{version}.jar ,
- 如果是AndroidStudio环境，打开工作空间，复制SDK库文件sdk-ad-{version}.jar,

在代码中使用sdk

修改manifest配置文件

1.添加权限

```
<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.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_SETTINGS" />

<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" />
```

2.注册Activity组件

```
<activity
    android:name="com.ks.client.ads.AdActivity"
    android:configChanges="keyboardHidden|orientation|screenSize"
    android:screenOrientation="sensor"
    android:theme="@android:style/Theme.Black.NoTitleBar.Fullscreen" />

<activity
    android:name="com.ks.permission.RequestPermissionActivity"
    android:configChanges="keyboardHidden|orientation|screenSize"
    android:screenOrientation="sensor"
    android:theme="@android:style/Theme.Translucent.NoTitleBar.Fullscreen" />
```

说明：如果奖励视频不需要随机器旋转而旋转的请将AndroidManifest里面的Activity设置android:screenOrientation="sensor"修改横屏或竖屏

3.注册Service组件

```
 <service
      android:name="com.ks.client.ads.DownloadService"
      android:exported="false" />
```

4.添加应用id

```
 <meta-data
            android:name="adsdk_appid"
            android:value="6bq3bpop" />
```

特注: - meta-data标签一定要在application标签下; - name必须为 adsdk_appid,value为您在您的APPID值 - 只有您提交的应用通过审核后，才会获得有 效的APP ID

5.build添加第三方依赖

```
dependencies {
    compile 'com.github.bumptech.glide:glide:3.7.0'
}
```

Application 中初始化

为了让用户能够获取更精准的数据，因此sdk在客户端app集成广告之前需要初始化数据,一般都是在客客户端 application 的 onCreate 中进行 sdk 初始化:

```
@Override
public void onCreate() {
    super.onCreate();
    ADApplication.getInstance().init(this, true,true);
    //应用id, 当meta-data也有adsdk_appid,以方法中的参数为准.
    ADApplication.getInstance().init(this,"appid", true,true);

}
```

参数 init(Context context,boolean isDebug,boolean logSwitch)
上下文 Context ,用户测试时 isDebug 为 true,正式上线时可以改为 false;
logSwitch 日志是否开启

开屏广告(SplashAd)

初始化

目前只支持代码的形式嵌入,在 activity 的 onCreate 方法中集成 SplashAd

```
// 重要：请填上您的广告位ID，代码位错误会导致无法请求到广告 debug
String adPlaceId = "sx8d0lkj";
plashAd = new SplashAd(this, rl_splash_adsParent, listener, adPlaceId, 8, true);
```

SpalshAd 的构造函数参数解释如下:
1、上下文，
2、需要添加的广告占位容器，
3、广告加载成功，失败，消失监听，
4、广告位 id, 5、默认显示时长，单位为 s，
6、是否可以被点击

生命周期

Activity 和 SplashAd 生命周期绑定，防止内存泄漏

```
@Override
  protected void onDestroy() {
  if (splashAd != null) {
         splashAd.destroy();
     }
   super.onDestroy();

  }
```

横幅 Banner 广告(SplashAd)

初始化

目前只支持在代码中集成，在 activity 的 onCreate()方法中植集成横幅 Banner 广告

```
String adId="vtt8ro1o"; //测试
  BannerAd bannerAd = new BannerAd(this, adId, AbScreenUtils.getRealWidth(this, 640),
   AbScreenUtils.getRealHeight(this, 100),
   new BannerAdListener() {
      @Override
      public void onAdPresent() {
          Log.i("ManActivity", "onAdPresent");

      }

      @Override
      public void onAdDismissed() {
          Log.i("ManActivity", "onAdDismissed");

      }

      @Override
      public void onAdFailed(String var1) {
          Log.i("ManActivity", "onAdFailed");

      }

      @Override
      public void onAdClick() {
          Log.i("ManActivity", "onAdClick");

      }
  });
      this.bannerAd = bannerAd;
      View bannerView = this.bannerAd.getBannerView();
      RelativeLayout.LayoutParams layoutParams =
      (RelativeLayout.LayoutParams) bannerView.getLayoutParams();
      layoutParams.addRule(RelativeLayout.ALIGN_PARENT_BOTTOM);
      layoutParams.addRule(RelativeLayout.CENTER_HORIZONTAL);
      ll_main_container.addView(bannerView, layoutParams);
```

直接创建出一个 BannerAd 对象，通过 BannerAd 实例对象，调用 getBannerView()方法 将其横幅 Banner 广告添加到客户端指定的容器中。
BannerAd 构造函数参数解释如下:
1.上下文
2.广告位 id
3 横幅广告的宽
4.横幅广告的高
5.横幅 banner 的展示，关闭，加载失败，点击监听

生命周期

Activity 和横幅 banner 广告生命周期绑定，防止内存泄漏:

```
@Override
 protected void onDestroy() {
     if (bannerAd != null) {
         bannerAd.destroy();
     }
     super.onDestroy();
 }
```

插屏广告(InterstitialAd)

初始化

目前 InterstitialAd 广告只支持代码集成，
在 Activity 的 onCreate 方法中进行插屏广告集成:

```
String adId="oc1lyk7w";//测试
  ad = new InterstitialAd(this, adId, 600, 500, new InterstitialAdListener() {
        /**
         * 广告准备好时回调
         */
        @Override
        public void onAdReady() {
            Toast.makeText(InterstitialAdtivity.this,"onAdReady",Toast.LENGTH_SHORT).shc
        }

        /**
         * 广告展示时回调
         */
        @Override
        public void onAdPresent() {
            Toast.makeText(InterstitialAdtivity.this,"onAdPresent",Toast.LENGTH_SHORT).s

        }
        /**
         * 广告点击时回调
         */
        @Override
        public void onAdClick() {
            Toast.makeText(InterstitialAdtivity.this,"onAdClick",Toast.LENGTH_SHORT).shc

        }
        /**
         * 广告关闭时回调
         */
        @Override
        public void onAdDismissed() {
            Toast.makeText(InterstitialAdtivity.this,"onAdDismissed",Toast.LENGTH_SHORT)

        }
        /**
         * 广告请求失败时回调
         */
        @Override
        public void onAdFailed(String var1) {
            Toast.makeText(InterstitialAdtivity.this,"onAdFailed",Toast.LENGTH_SHORT).sh

        }
    });
```

InterstitialAd 的构造函数参数解释如下:
1.上下文
2.广告位 id
3.插屏广告的宽
4.插屏广告的高

此外，通过调用 InterstitialAd 的实例对象的 loadAd() 来加载广告。
showAd()和 hideAd()来控制插屏广告的显示 和隐藏，
实例代码如下:

```
public void loadAd(View view) {
    if (ad != null)
         ad.loadAd();
}
public void showAd(View view) {
    if (ad != null)
        if (ad.isAdReady())
         ad.showAd();
}


@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
    if (keyCode == KeyEvent.KEYCODE_BACK && !ad.isAdClosed()) {
        ad.hideAd();
        return false;
    } else {
        return super.onKeyDown(keyCode, event);
    }
}
```

生命周期

此外 InterstitialAd 和 Activity 的生命周期绑定在一起，防止内存泄漏

```
@Override
protected void onDestroy() {
    if (ad != null) {
        ad.destroy();
    }
    super.onDestroy();
}
```

奖励视频

初始化SDK

注意：所有接口请在主线程调用

i.设置Debug模式

```
ADSDK.getInstance().setDebug(false).setLogSwitch(true);
```

setDebug():True为debug模式，false为release模式，默认false
当设置为true时将链接测试环境
设置为false是将链接正式环境
注意：上线时务必修改为false
setLogSwitch():true为开启日志开关，false为关闭日志开关。

ii.初始化

请在Activity onCreate方法调用

```
ADSDK.getInstance().init(this, mAppId, mAdSlotId, new AdEventListener() {});
```

参数说明：

字段名	类型	说明
activity	Activity	Android上下文
mAppId	String	应用ID，在MSSP后台或向相关人员申请
adSlotId	String	广告位ID，在MSSP后台或向相关人员申请
adEventListener	AdEventListener	广告事件回调

回调说明

```
public interface AdEventListener {
/**
 * 是否有广告
 *
 * @param isAdExist 有广告为true，反之false
 * @param code      错误码
 */
void onAdExist(boolean isAdExist, long code);

/**
 * 广告视频是否缓存了
 *
 * @param isCached 已缓存为true，反之false
 */
void onVideoCached(boolean isCached);

/**
 * 开始播放视频
 */
void onVideoStart();

/**
 * 视频播放完成
 *
 * @param isLookBack 是否回看，true为回看，不发奖励；false为第一次看，发奖励
 */
void onVideoCompletion(boolean isLookBack);

/**
 * 关闭广告
 *
 * @param currentPosition 当前播放进度
 */
void onVideoClose(int currentPosition);

/**
 * 视频播放错误
 *
 * @param reason 错误原因
 */
void onVideoError(String reason);

/**
 * 落地页关闭
 *
 * @param status 直接关闭为false，点击下载关闭为true
 */
void onLandingPageClose(boolean status);

/**
 * 开始下载
 */
void onDownloadStart();

/**
 * 网络错误
 *
 * @param error 错误原因
 */
void onNetRequestError(String error);
```

注意： onVideoCompletion(boolean isLookBack) 注意: 当isLookBack为true时表示重新观看完毕，不发奖励 为false时表示第一次观看完毕，发奖励

播放视频

```
if (ADSDK.getInstance().getAdStatus(MainActivity.this)
!= VideoStatus.NO_AD) {
    ADSDK.getInstance().showAdVideo(MainActivity.this);
} else {
    ADSDK.getInstance().load(MainActivity.this);
}
```

视频状态:

状态	说明
VideoStatus.NO_AD	服务器没有广告返回
VideoStatus.HAVE_AD_NO_LOCAL_CACHE	服务器返回广告，视频缓存失败
VideoStatus.HAVE_AD_AND_LOCAL_CACHE	服务器返回广告，视频缓存成功

```
//如果对视频播放条件有要求，只想播放本地视频。
if (ADSDK.getInstance().getAdStatus(MainActivity.this)
== VideoStatus.HAVE_AD_AND_LOCAL_CACHE) {
    ADSDK.getInstance().showAdVideo(MainActivity.this);
} else {
    ADSDK.getInstance().load(MainActivity.this);
}
```

生命周期

```
protected void onResume() {
    super.onResume();
    ADSDK.getInstance().onResume(MainActivity.this);
}

@Override
protected void onPause() {
    super.onPause();
    ADSDK.getInstance().onPause(MainActivity.this);
}

@Override
protected void onDestroy() {
    ADSDK.getInstance().release(MainActivity.this);
    super.onDestroy();
}
```

原生广告

获取原生广告

初始化对象AdNative(Activity activity, String adId, AdNativeNetworkListener listener)
通过adNative.loadNativeAd()异步获取原生广告。
广告结果会通过回调传入AdNativeNetworkListener通知开发者。

```
//信息流广告
AdNative adNative = new AdNative(activity, YOUR_AD_PLACE_ID,
 new AdNativeNetworkListener() {

    @Override
    public void onResponse(NativeBaseResponse nativeBaseResponse) {

        NativeAdResponse adResponse = (NativeAdResponse) nativeBaseResponse;
        if (adResponse.errorCode == 200) {
            nrAdList = adResponse.nativeAds;
            showAdList();
        }

    }

});
adNative.loadNativeAd();
```

渲染广告

通过adNative.getData()方法可获取对应广告的原生素材。

可用的素材信息如下表:

字段名	类型	说明
title	String	广告标题。
description	String	广告描述。
logoUrl	String	广告Logo图片地址。
imageUrl	String	广告大图图片地址。
mIsDownloadApp	boolean	是否是下载类广告
brandName	String	商家名。可空。

显示广告并处理广告点击

当您将adNative渲染完成并在界面上展示后，务必调用nativeAd.onShown(View)方法同时SDK已完成广告展示。 View为被展示的广告视图。

通过调用nativeAd.onClicked(view, jsonObject);调用SDK来处理广告点击。jsonObject 放置点击坐标详细参考demo

混淆配置

如果您需要使用proguard混淆代码，请不要混淆SDK代码。 请在您的proguard.cfg文件(或其他混淆文件)尾部添加如下配置:

```
-keep class com.ks.** {*;}
-keep class com.afk.** {*;}
-keep class com.google.protobuf.** {*;}
-keepattributes *Annotation*
-keepattributes *JavascriptInterface*
```

注意：SDK代码被混淆之后会导致广告无法显示或其他异常

注意事项

- 请务必使用在后台申请的广告参数，否则无法请求到广告
- 播放广告时，请关闭App音效
- 错误提示：打开debug模式在Logcat会输出错误信息，可以根据错误信息排查对应的错误
- 如果接入方不使用提供的android-support-v4.jar包，使用别的jar包的使用，建议使用最新版本的android-support-v4.jar包

兼容7.0的应用安装

媒体需要在AndroidManifest中的application标签中添加provider,如代码所示：

```
 <application ...
 <provider
   android:name="android.support.v4.content.AdFileProvider"
   android:authorities="${applicationId}.fileprovider"
   android:exported="false"
   android:grantUriPermissions="true">
   <meta-data
       android:name="android.support.FILE_PROVIDER_PATHS"
       android:resource="@xml/provider_paths" />
   </provider>
 </application>
```

在res 中创建xml路径，并添加文件provider_paths;

```
<?xml version="1.0" encoding="utf-8"?>
<paths xmlns:android="http://schemas.android.com/apk/res/android">
    <external-path name="external_files" path="."/>
</paths>
```