修订历史

文档版本	修订日期	修订说明
V1.0.0	2022-03-30	SDK初始版本
v1.0.1	2022-04-19	新增支持广点通信息流模板广告
V1.0.2	2022-04-25	新增支持channel渠道广告，新增广告容器组件
V1.0.3	2022-04-28	新增插屏广告场景参数的设置
V1.0.4	2022-06-06	新增信息流自渲染广告、新增信息流模板插屏样式组件、新增非标准广告样式组件、调整了信息流插屏的样例
V1.0.5	2022-06-13	优化内置的广告落地页
V1.0.6	2022-06-14	支持自行设置oiad
V1.0.7	2022-06-14	修改channel,zhike渠道的点击回调
V1.0.8	2022-06-14	内置广告落地页支持App scheme链接
V1.0.9	2022-06-15	channel自渲染广告限制广告点击上报只进行一次，修复scheme跳转bug
V1.0.10	2022-06-16	广告的跳转逻辑链路
V1.0.11	2022-06-16	修改插屏广告视图组件，修改广告落地页app scheme加载
	2022-	

V1.0.12	06-17	修改showNativeAdConvenient
V1.0.13	2022-06-17	bugfix showNativeAdConvenient
V1.0.14	2022-06-22	支持自定义获取获取广告配置接口的域名
V1.0.15	2022-06-24	修复多次调用onAdFailed问题
V1.0.16	2022-06-24	修改获取oaid的方式
V1.0.17	2022-07-05	修复bug,banner广告和信息流自渲染广告添加场景参数支持
V1.0.18	2022-07-06	修复广告加载失败没有触发回调事件的问题
V1.0.19	2022-07-08	修复channel,zhike的信息流自渲染广告的曝光回调问题
V1.0.20	2022-07-12	修复不能加载 gif 的问题
V1.0.21	2022-07-21	banner 添加关闭按钮逻辑,自渲染广告上报修改默认User-Agent
V1.0.22	2022-07-25	修改 SDK 初始化方式
V1.0.23	2022-07-28	添加 lingu sdk
V1.0.24	2022-07-30	修复 ua 获取方式
V1.0.25	2022-08-02	信息流模板样式修改
V1.0.26	2022-08-04	插屏广告添加自动关闭功能
V1.0.27	2022-08-17	信息流 UI 样式调整

V1.0.28	2022-09-07	广告策略调整
V1.0.29	2022-09-26	请求域名调整
V1.1.0	2022-10-16	新增开屏广告类型
V1.1.1	2022-10-20	更换请求动态域名
V1.1.2	2022-11-03	添加内置混淆方式
V1.1.3	2022-11-14	弹窗广告UI优化
V1.1.4	2022-12-05	优化代码逻辑
V1.1.5	2023-01-30	添加服务类型广告
V1.1.5	2023-01-30	添加服务类型广告
V1.1.6	2023-02-27	增加bidding广告竞价功能,更新 GDTSDK
V1.1.7	2023-03-08	增加停止广告请求功能
V1.1.8	2023-03-20	开屏添加热区和跳过按钮位置配置功能

添加SDK

将广告所需要的依赖集成进去，Sdk可根据接入平台情况进行选择接入。

```
defaultConfig {
  ndk {
     abiFilters 'armeabi'
  }
}


dependencies {
// 必须导入
implementation files("libs/AdBase-1.0.18-release.aar")
implementation files("libs/AdSdk-1.1.8-release.aar")
implementation files("libs/AdapterChannel-1.1.8-release.aar")
implementation files("libs/AdapterZhiKe-1.1.8-release.aar")
implementation 'com.github.bumptech.glide:glide:4.9.0'
}

// 如果接入项目为 AndroidX 项目，需要在 gradle.properties 文件中添加
android.enableJetifier=true
```

权限申请

使用SDK时可能需要以下权限，为了保证使用广告的正确，请在6.0及以上的手机中使用SDK前及时申请

```
<!-- 广告必须的权限，允许网络访问 -->
<uses-permission android:name="android.permission.INTERNET" />
<!-- 广告必须的权限，允许安装未知来源权限（如下载类广告下载完成后唤起安卓） -->
<uses-permission android:name="android.permission.REQUEST_INSTALL_PACK
<!-- 广告必须的权限，地理位置权限，获取位置信息，用于广告投放 -->
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION

<!-- 如果有视频相关的广告播放请务必添加，屏幕保持唤醒不锁屏（部分渠道未添加该权限时
<uses-permission android:name="android.permission.WAKE_LOCK" />

<!-- 如果接入了广点通渠道，必须加入以下权限，不然会导致广点通填充失败 -->
<!-- 允许应用获取 MAC 地址，可选权限，采用 App 内推广时会用到此权限 -->
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<!-- 允许应用检测网络状态，SDK 会根据网络状态选择是否发送数据 -->
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE

<!-- 影响广告填充，强烈建议的权限，获取设备信息，允许应用获取手机状态（包括手机号码
<uses-permission android:name="android.permission.READ_PHONE_STATE" />

<!-- 为了提高广告收益，建议设置的权限，写入权限，用于下载类广告数据写入 -->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORA
<!-- 为了提高广告收益，建议设置的权限，读取权限，用于下载类广告数据读取（如判断是否
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAG

<!-- 为了提高广告收益，建议设置的权限，获取粗略位置信息 -->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATI
```

sdk初始化

> 网络配置，因为上报中有些请求是 http 类型，需要配置允许明文传输 1. 如果 AndroidManifest.xml 的 Application 节点下，有 android:networkSecurityConfig 配置，那么需要在配置文件下添加 2. 如果没有 android:networkSecurityConfig 配置，可以添加在 Application 节点下添加 android:usesCleartextTraffic="true"

```
AdSdk.getInstance().setBaseHttp("http://ad.baihemob.com");
AdSdk.getInstance().setOAID(oaid);
AdSdk.getInstance().init(Activity, "appid");
```

Banner横幅广告示例

Banner横幅广告建议放置在 固定位置，而非ListView、RecyclerView、ViewPager等控件中充当Item，Banner广告支持多种尺寸比例，可在后台创建广告位时配置，Banner广告的宽度将会撑满容器，高度自适应，建议Banner广告容器高度也为自适应。

```
// 创建Banner广告实例，第一个参数可以是Activity或Fragment，第二个参数是广告容器（i
BannerAd BannerAd = new BannerAd(this, flContainer);
bannerAd.setAdId(广告位id);
// 竞标价格，单位为分。 ps:需要在 setAdId() 之后调用
int price=bannerAd.getBidPrice();
//竞赢上报,比价成功后需调用。price 我方出价
bannerAd.sendWinNotice(int price);
//竞败上报,比价失败或者其他情况调用。price 三方竞赢价格(分)
bannerAd.sendLossNotice(int price);

// 设置自刷新时间间隔，0为不自动刷新，其他取值范围为[30,120]，单位秒
bannerAd.setAutoRefreshInterval(30);

// 设置Banner广告监听
BannerAd.setListener(new BannerAdListener() {

            @Override
            public void onAdReceive(AdInfo adInfo) {
                Log.d(TAG, "广告获取成功");
            }

            public void onAdExpose(AdInfo adInfo) {
                Log.d(TAG, "广告曝光");
            }

            @Override
            public void onAdClick(AdInfo ad) {
                Log.d(TAG, "广告点击");
            }

            @Override
            public void onAdClick(AdInfo ad, String skipUrl) {
                Log.d(TAG, "广告点击skipUrl:" + skipUrl);
            }

            @Override
            public void onAdClose(AdInfo ad) {
                Log.d(TAG, "广告关闭");
            }

            @Override
            public void onAdFailed(AdError adError) {
                Log.d(TAG, "广告获取失败");
            }
        });

// 加载Banner广告
bannerAd.loadAd();
```

AdInfo广告信息实现方法 ```java /** * 获取广告来源平台名称 * * @return 广告来源平台名称 */ String getPlatform();

```
/**
 * 获取广告来源平台的广告角标资源ID，角标资源大小为138x54
 *
 * @return 广告来源平台的广告角标资源ID
 */
@DrawableRes
int getPlatformIcon();

/**
 * 获取广告来源平台的广告位ID
 *
 * @return 广告来源平台的广告位ID
 */
String getPlatformPosId();

/**
 * 判断广告是否已经被释放，被释放的广告不可继续使用
 * 可以通过{@link ADAdUtil}的adInfoIsRelease方法便捷判断
 * @return 广告是否被释放
 */
boolean isReleased();

/**
 * 释放广告Info对象，释放后的广告Info对象不可再次使用，各个广告类型对象（{@link ADA
 */
@Override
void release();
/**
 * 获取落地页地址
 *
 * @return 落地页地址
 */
String getAdUrl();
```

```
## 插屏广告
插屏广告是移动广告的一种常见形式，在应用流程中弹出，当应用展示插屏广告时，用户可以选择点
```java
interstitialAd = new InterstitialAd(Activity或Fragment);
interstitialAd.setCloseTime(time);// time ms 后自动关闭
interstitialAd.setAdId(广告位id);
// 竞标价格，单位为分。 ps:需要在 setAdId() 之后调用
int price=interstitialAd.getBidPrice();
//竞赢上报,比价成功后需调用。price 我方出价
```
```

```
interstitialAd.sendWinNotice(int price);
//竞败上报,比价失败或者其他情况调用。price 三方竞赢价格(分)
interstitialAd.sendLossNotice(int price);

Map scenesMap=new TreeMap();
scenesMap.put("station_id","123");
scenesMap.put("in_out","0");
interstitialAd.setScenes(scenesMap);
interstitialAd.setListener(new ADInterstitialAdListener() {
    /**
         * 广告准备完毕回调
         *
         */
    public void onAdReady(InterstitialAdInfo interstitialAdInfo) {
                // 建议在该回调之后展示广告
            Log.d(TAG, "onAdReady----->");
            Log.d(TAG, "广告准备完毕回调... ");
             AdUtil.showInterstitialAd(activity, interstitialAdInfo);
        }
        /**
         * 广告获取成功回调
         *
         */
    public void onAdReceive(InterstitialAdInfo interstitialAdInfo) {

                Log.d(TAG, "广告获取成功回调... ");
     }
      public void onAdExpose(InterstitialAdInfo interstitialAdInfo) {
                Log.d(TAG, "广告展示回调，有展示回调不一定是有效曝光，如网络等情况
      }
      public void onAdClick(InterstitialAdInfo interstitialAdInfo) {
                Log.d(TAG, "广告点击回调，有点击回调不一定是有效点击，如网络等情况
      }
      public void onAdClose(InterstitialAdInfo interstitialAdInfo) {
                Log.d(TAG, "广告关闭回调");
      }
      public void onAdFailed(Error Error) {
                Log.d(TAG, "onAdFailed----->" + Error);
      }
  }
 interstitialAd.loadAd();
```

信息流广告

信息流广告此时只支持模板样式，接口如下： ```java // 创建信息流广告实例 NativeAd adNativeAd = new NativeAd(this); Map scenesMap=new TreeMap(); scenesMap.put("station*id","123"); scenesMap.put("in*out","0");

scenesMap.put("userid","12345"); adNativeAd.setScenes(scenesMap); nativeAd.setAdId(广告位id); // 竞标价格，单位为分。 ps:需要在 setAdId() 之后调用 int price=nativeAd.getBidPrice(); //竞赢上报,比价成功后需调用。price 我方出价 nativeAd.sendWinNotice(int price); //竞败上报,比价失败或者其他情况调用。price 三方竞赢价格(分) nativeAd.sendLossNotice(int price); AdUtil.setCloseTime(time);// time ms 自动关闭(当调用 AdUtil.showNativeAdConvenient，此参数有用)

//请求广告数量 nativeAd.setNativeAdCount(1); // 设置广告监听 adNativeAd.setListener(new NativeAdListener() { @Override public void onAdReceive(List adInfoList) { Log.d(TAG, "广告获取成功 size:" + adInfoList.size()); //展示信息流广告 //AdUtil.showNaiveAd(nativeView, adInfoList.get(0));

```
            //展示信息流广告,且滑动关闭
            //AdUtil.showNaiveAdSlideClose(nativeView, "#fdffff", adInfo

            //信息流广告 作为插屏形式展示
            AdUtil.showNativeAdConvenient(MainActivity.this, adInfoList.
        }

        @Override
        public void onAdExpose(NativeAdInfo adInfo) {
            Log.d(TAG, "广告曝光");
        }

        @Override
        public void onAdClick(NativeAdInfo ad) {
            Log.d(TAG, "广告点击");
        }

        @Override
        public void onAdClick(NativeAdInfo ad, String skipUrl) {
            Log.d(TAG, "广告点击:" + skipUrl);
        }

        @Override
        public void onAdClose(NativeAdInfo ad) {
            Log.d(TAG, "广告关闭");
        }

        @Override
        public void onAdFailed(AdError adError) {
            Log.d(TAG, "广告获取失败");
        }

        @Override
        public void onRenderFailed(NativeAdInfo adNativeAdInfo, AdError a
            Log.d(TAG, "广告渲染失败");
        }
```

});

// 请求广告数据 nativeAd.loadAd(); ```

开屏广告

开屏广告是移动广告的一种常见形式 ```java SplashAd splashAd = new SplashAd(this,splashView); splashAd.setAdId(广告位id); // 竞标价格，单位为分。 ps:需要在 setAdId() 之后调用 int price=splashAd.getBidPrice(); //竞赢上报,比价成功后需调用。price 我

方出价 splashAd.sendWinNotice(int price); //竞败上报,比价失败或者其他情况调用。price 三方竞赢价格(分) splashAd.sendLossNotice(int price);

// 设置开屏广告大小 splashAd.setAdSize(new SplashAd.AdSize(int width,int height)); ps:单位 px // 设置跳过按钮，由外部传入。time : 展示时间，建议取值范围 3000 ms- 5000ms。默认 3000ms // skipView 和 time 都可以不传 splashAd.setSkipView(View skipView,time); splashAd.setAdListener(new SplashAdListener() { @Override public void onAdTick(long countdownSeconds) {

```
            Log.d(TAG, "倒计时 "+ countdownSeconds +" s");
        }

        @Override
        public void onAdSkip(AdInfo adInfo) {
            Log.d(TAG, "点击跳过按钮");
        }

        @Override
        public void onAdReceive(AdInfo adInfo) {
            Log.d(TAG, "广告获取成功");
        }

        @Override
        public void onAdExpose(AdInfo adInfo) {
            Log.d(TAG, "onAdExpose");
        }

        @Override
        public void onAdClick(AdInfo ad) {
            Log.d(TAG, "onAdClick");
        }

        @Override
        public void onAdClose(AdInfo ad) {
            Log.d(TAG, "onAdClose");
            splashView.setVisibility(View.GONE);
        }

        @Override
        public void onAdFailed(AdError adError) {
            Log.e(TAG, "onAdFailed "+adError.errorMsg());
        }
    });
    splashAd.loadAd();
```

原生自渲染广告
自渲染广告布局
创建UnifiedADContainer容器，该容器兼容广点通的NativeAdContainer，UnifiedADCon
```xml
<com.doublejump.top.UnifiedADContainer
        android:id="@+id/unified_container"
        android:layout_width="375dp"
        android:layout_height="123dp"
        android:layout_gravity="center_horizontal">

        <ImageView
            android:id="@+id/unified_container_img"
            android:layout_width="200dp"
            android:layout_height="match_parent" />

        <RelativeLayout
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_gravity="left"
            android:layout_marginLeft="210dp">

            <TextView
                android:id="@+id/unified_container_title_tv"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_alignParentTop="true"
                android:layout_margin="10dp" />

            <TextView
                android:id="@+id/unified_container_desc_tv"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_alignParentBottom="true"
                android:layout_margin="10dp" />
        </RelativeLayout>

    </com.doublejump.top.UnifiedADContainer>
```

自渲染广告加载与监听

```
UnifiedAD unifiedAD = new UnifiedAD(this);
unifiedAD.setAdId(广告id);
// 竞标价格，单位为分。 ps:需要在 setAdId() 之后调用
int price=unifiedAD.getBidPrice();
//竞赢上报,比价成功后需调用。price 我方出价
unifiedAD.sendWinNotice(int price);
//竞败上报,比价失败或者其他情况调用。price 三方竞赢价格(分)
unifiedAD.sendLossNotice(int price);

unifiedAD.setAdListener(new UnifiedADListener() {
    @Override
    public void onADLoaded(UnifiedAdInfo adInfo) {
        Log.d(TAG, "广告数据获取成功时回调");
        showUnified(adInfo);//处理回传广告实体
    }

    @Override
    public void onAdExpose(UnifiedAdInfo adInfo) {
        Log.d(TAG, "广告被曝光回调");
    }

    @Override
    public void onAdClick(UnifiedAdInfo ad) {
        Log.d(TAG, "广告被点击回调");
    }

    @Override
    public void onAdClose(UnifiedAdInfo ad) {
        Log.d(TAG, "广告被关闭");
    }

    @Override
    public void onAdFailed(AdError adError) {
        super.onAdFailed(adError);
        Log.d(TAG, "onAdFailed");
        adError.log();
    }
});
unifiedAD.loadAd();
```

UnifiedAdInfo广告数据

加载自渲染广告返回的广告数据，包含展示自渲染广告所需的所有元素，视图和广告绑定接口。

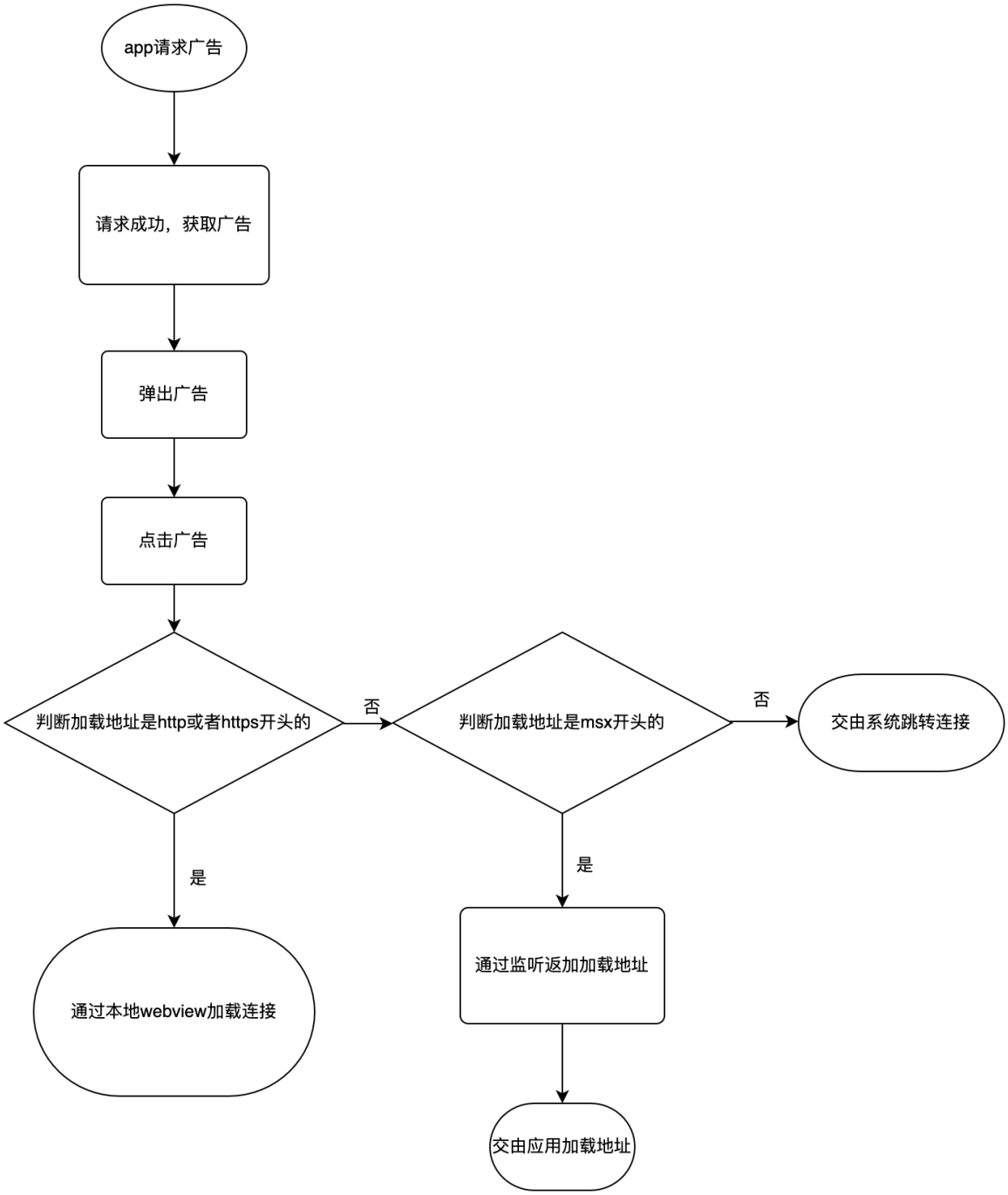
方法名	方法介绍
getTitle()	获取广告标题，短文字
getDesc()	获取广告描述，长文字
getIconUrl()	获取 Icon 图片地址
getImgUrl()	获取大图地址，使素材展示更完整，优化展示效果，建议通过调用bindImageViews由SDK来渲染图片，详细可参考bindImageViews方法使用说明和Demo实现
getAdPatternType()	获取广告样式
getImgList()	获取三小图的地址，使素材展示更完整，优化展示效果，建议通过调用bindImageViews由SDK来渲染图片，详细可参考bindImageViews方法使用说明和Demo实现
bindAdToView(Context context, AdContainer container, List clickViews)	View和广告的绑定，必须设置AdContainer；clickViews为触发广告点击行为的View，必须在container中，不然不会响应点击事件，且开发者不能对clickViews设置OnClickListener，会影响点击事件的上报，点击事件回调可通过ADEventListener中onADClicked()回调监听
bindAdToView(Context context, AdContainer container, List clickViews, List customClickViews)	View和广告的绑定，必须设置AdContainer；clickViews为触发广告点击行为的View，customClickViews点击可以直接下载或进入落地页，两者必须在container中，不然不会响应点击事件，且开发者不能对clickViews和customClickViews设置OnClickListener，会影响点击事件的上报，点击事件回调可通过NativeADEventListener中onADClicked()回调监听

服务类型广告/非标准广告

服务类型广告/非标准广告是指广告尺寸是定制的，与传通的横幅广告、信息流、开屏不一样。

```
// 创建非标准广告实例，第一个参数可以是Activity或Fragment，第二个参数是广告容器（请
ServiceAd serviceAd = new ServiceAd(this, flContainer);
serviceAd.setAdId(广告位id);
 // 竞标价格，单位为分。 ps:需要在 setAdId() 之后调用
int price=serviceAd.getBidPrice();
//竞赢上报,比价成功后需调用。price 我方出价
serviceAd.sendWinNotice(int price);
//竞败上报,比价失败或者其他情况调用。price 三方竞赢价格(分)
serviceAd.sendLossNotice(int price);

// 设置自刷新时间间隔，0为不自动刷新，其他取值范围为[30,120]，单位秒
serviceAd.setAutoRefreshInterval(30);
// 设置Banner广告监听
serviceAd.setListener(new BannerAdListener() {

            @Override
            public void onAdReceive(AdInfo adInfo) {
                Log.d(TAG, "广告获取成功");
            }

            public void onAdExpose(AdInfo adInfo) {
                Log.d(TAG, "广告曝光");
            }

            @Override
            public void onAdClick(AdInfo ad) {
                Log.d(TAG, "广告点击");
            }

            @Override
            public void onAdClick(AdInfo ad, String skipUrl) {
                Log.d(TAG, "广告点击skipUrl:" + skipUrl);
            }

            @Override
            public void onAdClose(AdInfo ad) {
                Log.d(TAG, "广告关闭");
            }

            @Override
            public void onAdFailed(AdError adError) {
                Log.d(TAG, "广告获取失败");
            }
        });

// 加载广告
serviceAd.loadAd();
```

广告的跳转逻辑链路


app请求广告
请求成功，获取广告
弹出广告
点击广告
判断加载地址是http或者https开头的
否
判断加载地址是msx开头的
否
交由系统跳转连接
是
通过本地webview加载连接
是
通过监听返加加载地址
交由应用加载地址