智能分包开发文档 V3.0

一、概述

用于读取智能分包参数

二、具体实现

商店为开发者提供了两种方式读取智能分包参数，推荐使用 ContentProvider 接入方案；以下是每种方案的接入代码示例，请仔细阅读相关源码以及注释，按 MS 规范接入

注意：

在读取到分包参数后，需要将分包参数存储到本地缓存！

由于智能分包在安装后就不会产生变化 ，建议在读取成功后将智能分包存储到本地缓存，有利于提高参数读取效率

广告主从智能分包服务获取到的参数信息；只需要回传完整的智能分包参数 json 格式内容，不需要额外的处理

2.1 AIDL 代码接入

首先，创建接口 IAppStoreOpenService 的 aidl 文件，放置在 com.bbk.appstore.openinterface 包路径下，并按以下步骤进行接入：

（1）将以下内容复制到 aidl 文件中。

```
package com.bbk.appstore.openinterface;
import com.bbk.appstore.openinterface.IChannelDataCallback;
// Declare any non-default types here with import statements

interface IVAppStoreOpenService {
       /**
        * 读取分包参数
        * requestType: "readChannelInfer"
        * input:  json 数据格式
        * 消息格式：{"packageName":"要查询的包名"}
        * dataCallback：回调接口
```

```
        * handler:用户传入子线程的 handler，保证其在子线程中进行
        */
        oneway void request(in String requestType, in String inputJson, in
IChannelDataCallback callback);
}
package com.bbk.appstore.openinterface;

// Declare any non-default types here with import statements

interface IChannelDataCallback {
       /**
        * 批量查询下载状态回调
        * dataType 为调用方传入的值返回，智能分包参数
        * jsonStr 数据格式：json 格式
        */
        void onResponse(String requestType, String input, String jsonStr);
}
```

（2）创建一个类，实现 Android 原生的 ServiceConnection 接口。

a. 实现 ServiceConnection 的 onServiceConnected 方法。
b. 调用 Android 原生的 IPPSChannelInfoService.Stub.asInterface 方法获取 IPPSChannelInfoService。
c. 调用 getChannelInfo 方法获取转化跟踪参数。

```
public class InstallReferrerServiceConnection implements ServiceConnection {

    @Override
    public void onServiceConnected(ComponentName name, IBinder iBinder) {
        Log.i(TAG, "onServiceConnected");
        IVAppStoreOpenService service =
IVAppStoreOpenService.Stub.asInterface(iBinder);
        if (null != service) {
            try {
                JSONObject jsonObject = new JSONObject();
                jsonObject.put("packageName", packageName);//三方需要读取的包
名
                service.request("readChannel", jsonObject.toString(), new
com.bbk.appstore.openinterface.IChannelDataCallback.Stub() {
                    @Override
```

```
                    public void onResponse(String requestType, String input,
String jsonStr) throws RemoteException {
                                                //获取智能分包参数信息
                        String installReferrer = parseChannelJson(jsonStr);
                                                //通过监听回调分包参数，用于归因
和缓存
                        mCallBackListener.setChannelStr(installReferrer);
                                                //解绑服务

        getApplicationContext().unbindService(this);
                    }
                });
            } catch (RemoteException | JSONException e) {
                Log.e(TAG, "getChannelInfo Exception");
            } finally {

            }
        }
    }

    @Override
    public void onServiceDisconnected(ComponentName name) {
        Log.i(TAG, "onServiceDisconnected");
    }

    private String parseChannelJson(String jsonStr){
        String installReferrer= null;
        try {
            Log.i("channel", "channelValue:" + jsonStr);
            JSONObject jsonObject = new JSONObject(jsonStr);
            int code = jsonObject.optInt("code");
            if (code == 0) {
                // 获取应用智能分包参数及相关参数
                installReferrer = jsonObject.optString("value");
                //*installReferrer 即为智能分包参数,三方可以对之进行相关操作
                Log.i("appInfo", "value:" + installReferrer);
            } else {
                // 失败，看 message 信息
                installReferrer = jsonObject.optString("message");
                Log.i("appInfo", "message:" + installReferrer);
            }
```

```
        } catch (Exception e) {
            e.printStackTrace();
        }
        return installReferrer;
    }
}
```

（3）连接转化跟踪参数的 AIDL 服务。

```
    private boolean bindService() {
        // 创建一个 InstallReferrerServiceConnection 实例
        InstallReferrerServiceConnection serviceConnection = new
InstallReferrerServiceConnection();
        // 创建一个 Intent，Action 是“com.bbk.appstore.CHANNEL_SERVICE”
        Intent intent = new Intent("com.bbk.appstore.CHANNEL_SERVICE");
               intent.setPackage("com.bbk.appstore");
        // 调用 bindService 连接转化跟踪参数的 AIDL 服务
        boolean result = getApplicationContext().bindService(intent,
serviceConnection, Context.BIND_AUTO_CREATE);
        Log.i(TAG, "bindService result: " + result);
        return result;
    }

    //* 定义一个读取 SP 文件的方法，从本地缓存中获取分包参数
    private static String read(Context context) {
        String data = null;
        SharedPreferences sp =
context.getSharedPreferences("com.bbk.appstore_install_referrer",
Context.MODE_PRIVATE);
        data = sp.getString("channel", null);
        return data;
    }
```

2.2 Contentprovider 代码接入

（1）读取示例

```
String installReferrer  = readChannel(mContext,"com.ss.android.article.news");
//"com.ss.android.article.news"为要读取的包名
```

（2）读取实现

```
    public static String readChannel(Context context, String packageName) {
                //从本地缓存中获取智能分包参数
        String installReferrer = read(context);
        //如果该应用已获取到智能分包参数则直接返回
        if (!TextUtils.isEmpty(installReferrer)) {
            return  installReferrer;
        }
        try {
            Bundle inputBundle = new Bundle();
            inputBundle.putString("package_name", packageName);
            Uri uri =
Uri.parse("content://com.bbk.appstore.provider.appstatus");
            Bundle bundle = context.getContentResolver().call(uri,
"read_channel", null, inputBundle);
            if (bundle != null) {
                String channelValue = bundle.getString("channelValue");
                Log.i("channel", "channelValue:" + channelValue);
                JSONObject jsonObject = new JSONObject(channelValue);
                int code = jsonObject.optInt("code");
                if (code == 0) { //
                    // 获取应用智能分包参数
                   installReferrer= jsonObject.optString("value");
                    // 存储在 sp 缓存中
                    save(context, installReferrer);
                    Log.i("appInfo", "value:" + installReferrer);
                } else {
                    // 失败，看 message 信息
                   installReferrer= jsonObject.optString("message");
                    Log.i("appInfo", "message:" + installReferrer);
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return installReferrer;
    }

    //定义一个保存数据的方法
    private static void save(Context context, String channel) {
```

```
        SharedPreferences sp =
context.getSharedPreferences("com.bbk.appstore_install_referrer",
Context.MODE_PRIVATE);
        SharedPreferences.Editor editor = sp.edit();
        editor.putString("channel", channel);
        editor.apply();
    }

    //* 定义一个读取 SP 文件的方法，从本地缓存中获取智能分包参数
    private static String read(Context context) {
        String data = null;
        SharedPreferences sp =
context.getSharedPreferences("com.bbk.appstore_install_referrer",
Context.MODE_PRIVATE);
        data = sp.getString("channel", null);
        return data;
    }
```

2.3 读取智能分包参数返回示例

广告主从智能分包服务获取到的参数信息；只需要回传完整的智能分包参数 json 格式内容，不需要额外的处理


```
{
        "referrer_click_timestamp_seconds":  "1658541060088",
        "install_referrer":  "task_id%3Dafaf4f65f45f445d5f%26channel_id%3Dapps
tore_001%26request_id%3Dasdffsafsf54fasfsdfsdfsdfsdfsfadfs%26ad_id%3D125556454
%26ext_info%3Dafasfasfsdfsdfsdfsdf%26",
        "package_name":  "com.dudu.flashlight",
        "vivo_ext_referrer":  "BdAwWkjLXRw2s3QUg_7_reZBeahaPY59klNFQYbYxJwYTOA
_g803WnUmShsAqlTXHzoE8yTdxRBaM-Y-
8PELuaP3CBcY7UuooOusQLrYhdPZACAswxUdT3h1k0111SOx9b-8UJci"
}
```


相关字段说明：

参数	含义	说明
referrer_click_timestamp_seconds	引荐来源网址点击事件发生时的客户端时间戳	单位（毫秒时间戳），如 "1632811393"
install_referrer	智能分包参数	归因信息 install_referrer（json 结构，可扩展），当前包含： • taskid：任务 ID，如 "afaf4f65"。 • channel_id：由**广告主运营**在 Vivo 营销平台投放系统针对广告任务绑定的**智能分包渠道号**，如 "channel12345"。 • request_id：广告请求 ID，如"1234567890" • ad_id：广告创意 ID，如"1234567890"。 • ext_info：回传参数，用于 oCPX 对接，如 "Dafaasda"。
package_name	应用包名	广告主应用包名，如 "com.XXX.XXXX"
vivo_ext_referrer	vivo 广告参数	广告信息回传补充参数，如 "BdAwWkjLXRw2s3QUg_7"。 **注：非广告主设置出现的乱码可忽略**