

# TMFMiniAppSDK Android接入

---

- [TMFMiniAppSDK Android接入](#)

- [1 获取容器SDK凭证](#)

- [1.1 创建应用](#)
    - [1.2 获取配置文件](#)

- [2 集成 SDK](#)

- [2.1 前置条件](#)
    - [2.2 集成方式](#)
      - [2.2.1 集成 SDK](#)
      - [2.2.2 扫码能力](#)
      - [2.2.3 定位地图能力](#)

- [3 使用 SDK](#)

- [3.1 调试](#)
    - [3.2 SDK初始化](#)
      - [3.2.1 配置文件获取](#)
      - [3.2.2 配置信息设置](#)
      - [3.2.3 其它初始化动作](#)
    - [3.3 小程序管理API](#)

- [3.3.1 打开小程序](#)
  - [3.3.1.1 小程序版本类型](#)
  - [3.3.1.2 登录和登出](#)
  - [3.3.1.3 打开小程序](#)
  - [3.3.1.4 打开二维码小程序](#)
- [3.3.2 删除小程序](#)
- [3.3.3 获取小程序信息](#)
  - [3.3.3.1 获取最近访问小程序列表](#)
  - [3.3.3.2 搜索正式小程序](#)
- [3.3.4 自定义JSAPI](#)
- [3.3.5 小程序宿主自定义](#)
  - [3.3.5.1 自定义胶囊](#)
  - [3.3.5.2 相册选择与图片预览](#)
  - [3.3.5.3 自定义授权UI](#)
  - [3.3.5.4 自定义授权用户信息](#)
- [3.3.6 分享](#)
- [3.3.7 微信小程序事件](#)
- [3.3.8 自定义权限列表](#)

○ [4 API](#)

- [4.1 MiniCode](#)
- [4.2 MiniApp](#)
- [4.3 MiniStartOptions](#)
- [4.4 MiniScene](#)

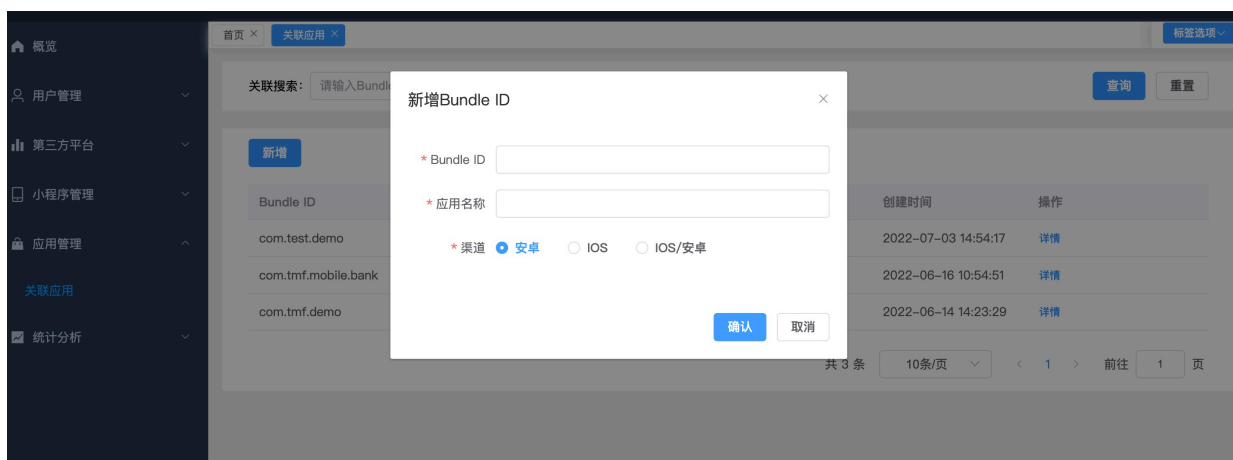
- [4.5 SearchOptions](#)
- [5 错误总结](#)
- [5.1 小程序内webview使用](#)

# 1 获取容器SDK凭证

使用容器SDK需要申请配置信息，只有在SDK初始化的时候配置了正确的配置文件，才能初始化成功并正常使用。

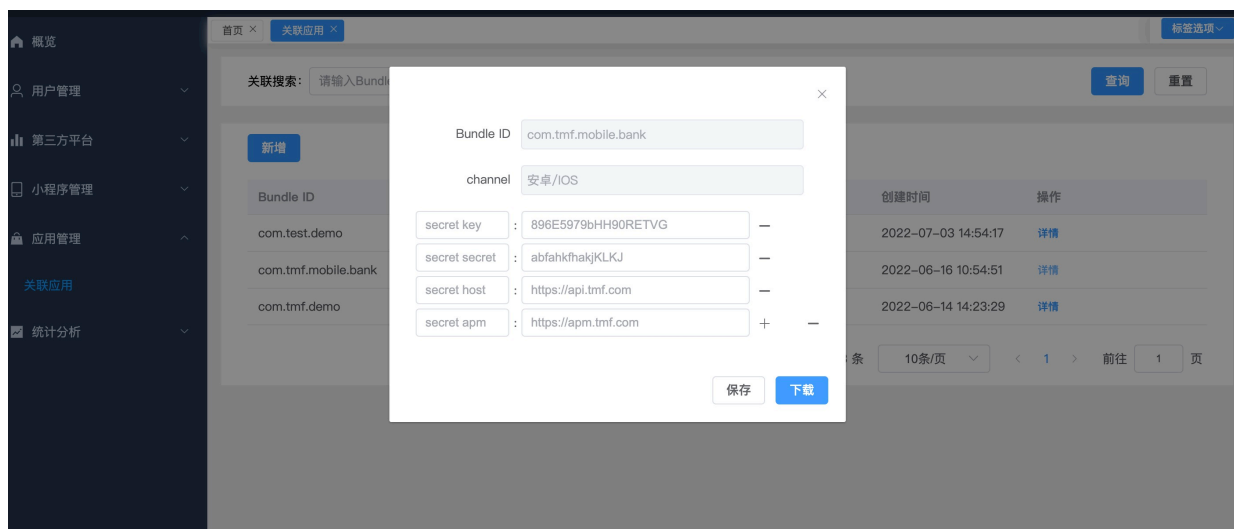
## 1.1 创建应用

需要登录企业运营端平台，输入Bundle ID、应用名称、渠道后完成应用创建：



## 1.2 获取配置文件

若要获取对应应用的配置文件，请选择对应Bundle ID右侧的详情操作，点击下载即可下载应用配置文件到本地：



## 2 集成 SDK

### 2.1 前置条件

- 环境要求
  - `minSdkVersion 21

### 2.2 集成方式

#### 2.2.1 集成 SDK

- 工程根目录添加如下仓库

```
allprojects {
    repositories {
        maven {
```

```
        url
        'https://t.pinpad.qq.com/fHKFBbEjd/repository/maven-public/'
        credentials {
            username ''
            password ''
        }
        authentication {
            basic(BasicAuthentication)
        }
    }
}
```

//maven的username和password参考demo设置

- 项目中添加如下依赖:

```
android {
    defaultConfig {
        packagingOptions {
            pickFirst 'lib/arm64-
v8a/libc++_shared.so'
            pickFirst
            'lib/armeabi/libc++_shared.so'
            pickFirst 'lib/armeabi-
v7a/libc++_shared.so'
            pickFirst 'lib/arm64-
v8a/libmarsxlog.so'
```

```

        pickFirst
        'lib/armeabi/libmarsxlog.so'
        pickFirst 'lib/armeabi-
v7a/libmarsxlog.so'
        pickFirst 'lib/arm64-
v8a/libv8jni.so'
    }
}
}

dependencies {
    implementation
    "com.google.android.material:material:1.3.0-
alpha03"
    // gson
    implementation
    'com.google.code.gson:gson:2.8.6'
    // ok-http
    implementation
    "com.squareup.okhttp3:okhttp:3.12.13"
    // x5内核
    implementation
    'com.tencent.tbs.tbssdk:sdk:43903'
    // mini app start
    kapt
    'com.tencent.tmf.android:mini_annotation_pro
cessor:x.x.x'
    implementation
    'com.tencent.tmf.android:mini_core:x.x.x'

```

```
}
```

如果开发者原来的工程中已集成了腾讯X5内核，则需要去除如下依赖

```
implementation  
'com.tencent.tbs.tbssdk:sdk:43903'
```

- 集成过程可能会遇到如下问题

```
AAPT: error: attribute  
android:requestLegacyExternalStorage not  
found.
```

在application标签下添加如下代码

```
<application  
    android:theme="@style/AppTheme"  
    tools:replace="android:icon"  
  
    tools:remove="android:requestLegacyExternalS  
torage">  
</application>
```

- 出现如下错误

```
Duplicate class
android.support.v4.app.INotificationSideChan
nel found in modules core-1.3.1-runtime
(androidx.core:core:1.3.1) and support-v4-
21.0.3-runtime (com.android.support:support-
v4:21.0.3)
```

在gradle.properties中添加如下代码

```
android.useAndroidX=true
android.enableJetifier=true
```



注意：x.x.x.x为版本号，参考demo工程中最新SDK版本。

## 2.2.2 扫码能力

开发者小程序如果使用了小程序扫码能力，需要添加如下sdk支持扫码功能；如未使用，无需添加，这样可以减小APP包大小

```
//扫码扩展组件
implementation
'com.tencent.tmf.android:mini_extra_qrcode:x
.x.x'
```



注意：x.x.x.x为版本号，参考demo工程中最新SDK版本。



## 2.2.3 定位地图能力

开发者小程序如果使用了小程序地图能力，需要添加如下 Sdk 支持 Google Map 功能；如未使用，无需添加，这样可以减小APP包大小。

```
implementation  
'com.tencent.tmf.android:mini_extra_google_map:1.4.0-SNAPSHOT'
```

您需要在您的 Google Cloud Console 配置 Google Cloud 项目，并获取访问 Google 地图服务所需要的 API 密钥，具体操作步骤请参考 [《在 Google Cloud Console 中进行设置》](#) 以及 [《使用 API 密钥》](#)。

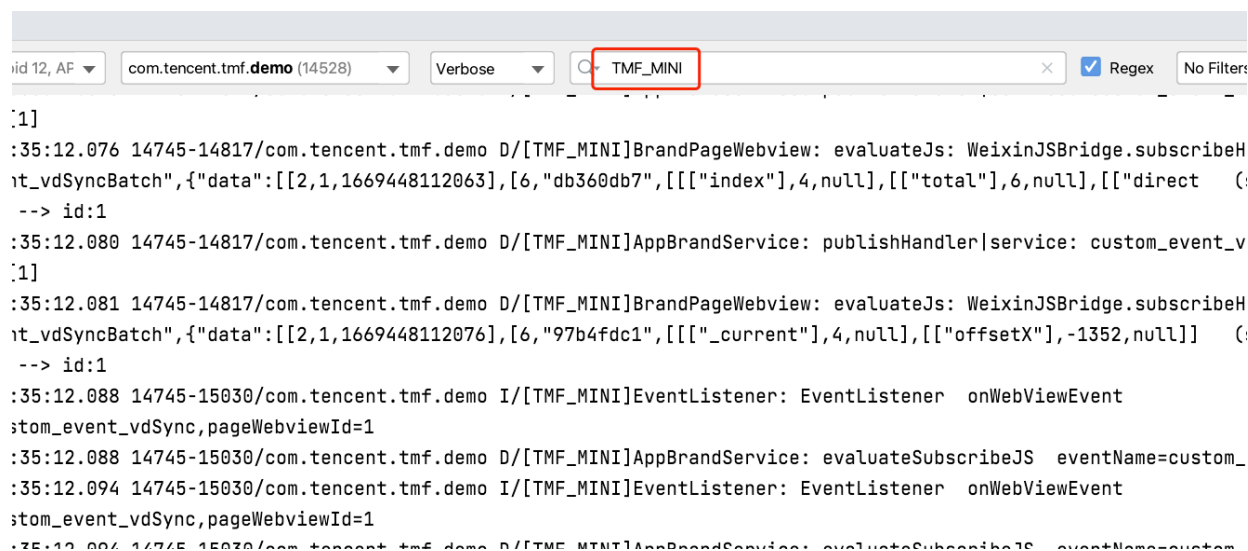
完成上述操作后，您需要在 Android 工程中配置您的 API 密钥。在 AndroidManifest.xml 文件中添加以下 meta-data，并将您的 API 密钥填入 (YOUR\_API\_KEY) 位置：

```
<application  
    ...  
    <meta-data  
  
        android:name="com.google.android.geo.API_KEY"  
        "  
            android:value="(YOUR_API_KEY)" />  
    ...  
</application>
```

# 3 使用 SDK

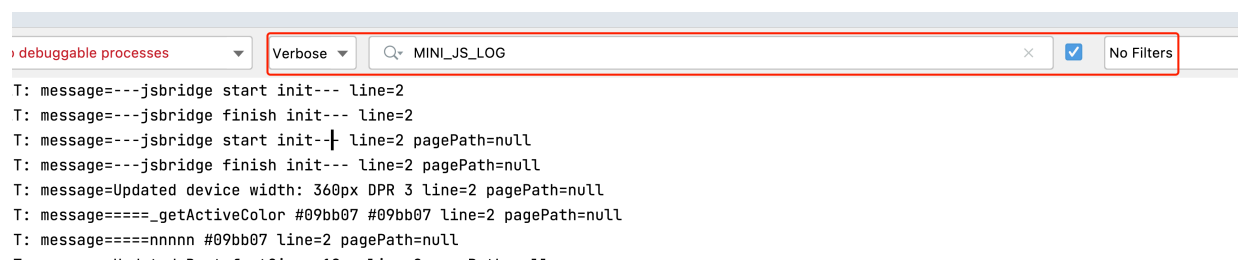
## 3.1 调试

- 通过关键字TMF\_MINI过滤得到SDK日志

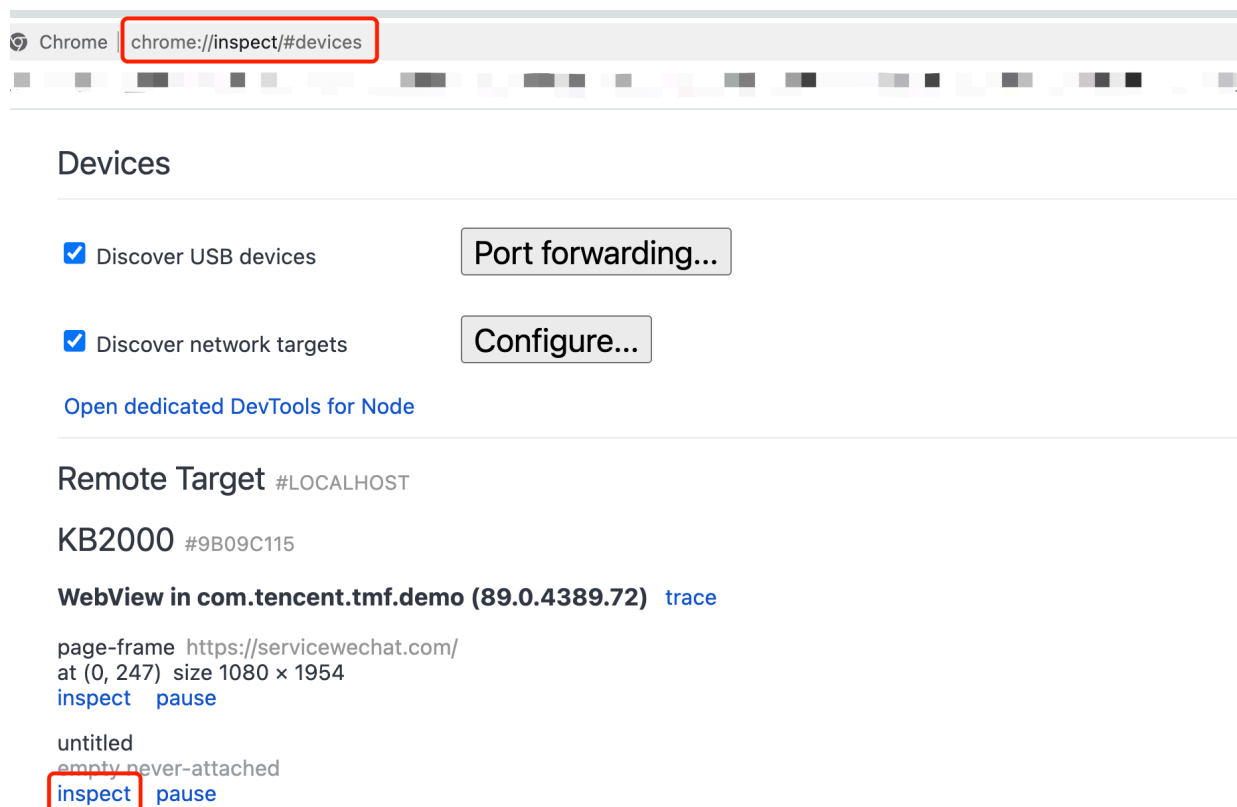


- 查看小程序JS错误日志

方式一：通过关键字MINI\_JS\_LOG过滤得到JS日志



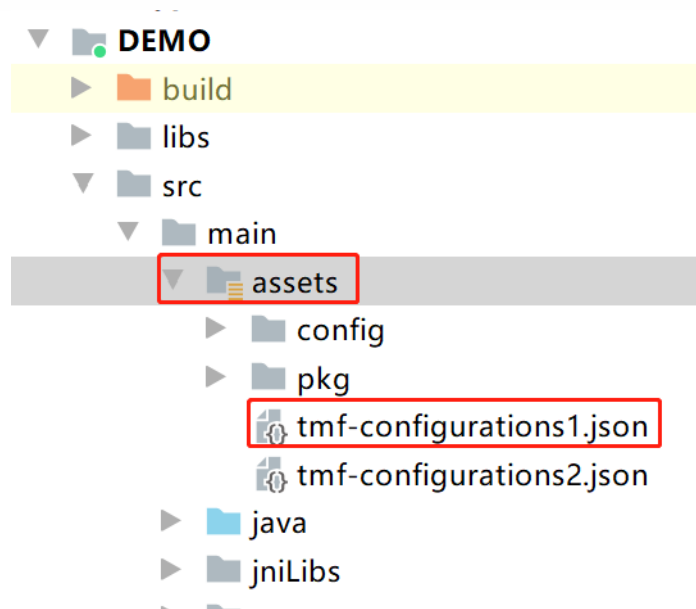
方式二：chrome调试小程序的js是否有错误（如有不懂咨询小程序开发）



## 3.2 SDK初始化

### 3.2.1 配置文件获取

开发人员从开放平台获取对应APP的配置文件，该配置文件是一个json文件，包含该app使用小程序平台的所有信息，将配置文件引入到项目的assets根目录。



注：配置文件中的**packageName**必须与应用的包名保持一致，否则APP运行失败

```
{
  "channel": "Android",
  "customId": "9ab50ac0-be06-11ec-a34e-278d66d394b5",
  "material": "01tGpVhGefdUbCe9JVR1H4b0yWseHioD/zgNFwd4WC03TvNYmT1YxW8LSjEbi3pyyo6A47wF",
  "packageName": "com.tencent.tmf.demo",
  "productId": "3255",
  "qapm": {
    "appKey": "None",
    "url": "https://tmfqapm.qq.com:30026"
  },
  "shark": {
    "appKey": "app-jqvp88v67o",
    "asymEnc": 1,
    "httpUrl": "https://tmfmp.qq.com:30013/",
    "symEnc": 2,
    "tcpHost": "42.194.253.71",
    "tcpPort": 30014
  }
}
```

### 3.2.2 配置信息设置

根据配置文件初始化一下MiniInitConfig对象，并使用MiniInitConfig初始化TMF小程序引擎。

参考代码：

```
MiniInitConfig config = builder

.configAssetName("TMF_CONFIGURATIONS")//assets中配置文件名称

    .imei("IMEI");//配置设备id, 用于在管理平台
    上根据设备标识进行小程序的灰度发布使用(可选)

    .build();
TmfMiniSDK.init(application, config);
```



注意：TmfMiniSDK.init初始化需要在Application中初始化

### 3.2.3 其它初始化动作

设置地区或者账号，方便进行灰度推送时使用；

```
/**
 * 设置账号信息
 * @param userId
 */
public static void setUserId(String userId)

/**
 * 设置位置信息
 * @param country
 * @param province
 * @param city
 */
public static void setLocation(String
country, String province, String city)
```

## 3.3 小程序管理API

### 3.3.1 打开小程序

#### 3.3.1.1 小程序版本类型

小程序版本分为三种：

- 正式版本：已上线对外使用，所有人都可以通过扫描二维码打开使用
- 预览版：必须要登录（运营账号和企业账号登录都可以，企业账号也叫开放平台或开发者账号）后才能扫码预览
- 开发版本：只能使用企业账号登录后才能扫码预览

### 3.3.1.2 登录和登出

```
/**
 * 登录
 *
 * @param userAccount
 * @param password
 * @param isOpenLogin true:企业账号(开发者账号);false:运营账号
 * @param result
 */
public static void login(String userAccount,
String password, final boolean isOpenLogin,
final MiniCallback<Void> result)

/**
 * 登出
 */
public static void logout()

/**
 * 登录是否过期
 * @return
 */
public static boolean isLoginOvertime()
```

### 3.3.1.3 打开小程序

## 正式小程序

打开小程序时，会先判断本地是否有缓存的小程序，如果没有，则会自动从远程服务器上下载小程序，然后打开。如果有缓存的小程序，则会先打开本地小程序，然后在后台校验服务器端是否有新版本。

如果有新版本，则下载新版小程序，下次打开时，就会使用新版小程序；如果没有新版本，则什么也不做。

## 预览和开发小程序

startMiniApp只能打开本地访问过的预览和开发小程序，通过getRecentList可以获得

```
/**
 * 启动小程序
 * @param activity
 * @param appId
 * @param scene 不同场景下打开小程序设置不同参数，
参见MiniScene
 * @param appVerType 小程序的版本类型(正式\预览
\开发版本),参见MiniApp
 * @param options
 */
public static void startMiniApp(Activity
activity, String appId, int scene, int
appVerType, MiniStartOptions options) {
```



miniStartOptions.resultReceiver可用于接收小程序启动错误情况，所以返回的错误码参考MiniCode，里面都有对应的说明

```
private ResultReceiver mResultReceiver = new
ResultReceiver(new Handler()) {
    @Override
    protected void onReceiveResult(int
resultCode, Bundle resultData) {
        if (resultCode != MiniCode.CODE_OK)
        {
            String errMsg =
resultData.getString("errMsg");
            Toast.makeText(mActivity, errMsg
+ resultCode, Toast.LENGTH_SHORT).show();
        }
    }
};
```

### 3.3.1.4 打开二维码小程序

TMF内置扫码模块，通过scan接口启动扫码，在onActivityResult中调用scanResult对扫码结果进行处理

```
/**
 * 启动扫码
 *
 * @param activity
 */
```

```
public static void scan(Activity activity)

/**
 * 获取扫码结果
 *
 * @param requestCode
 * @param intent
 * @return
 */
public static JSONObject getScanResult(int
requestCode, Intent intent)

/**
 * 通过扫码打开小程序，非TMF小程序二维码会返回错误
 *
 * @param activity
 * @param link
 * @param resultReceiver 接收小程序启动过程中错
误情况
 */
public static void
startMiniAppByLink(Activity activity, String
link, ResultReceiver resultReceiver)
```

### 3.3.2 删除小程序

由于小程序的运行，会将小程序包和小程序信息缓存在本地，以后打开时速度会非常快。所以，如果想要将小程序的所有信息都删除，那么可以调用以下api删除某个小程序或者删除所有小程序。

```
/**
 * 根据appId删除小程序(正式、开发、预览版都会删除)
 * @param appId
 */
public static void deleteMiniApp(String
appId)

/**
 * 删除指定类型和版本小程序
 * @param appId
 * @param appVerType
 * @param version
 */
public static void deleteMiniApp(String
appId, int appVerType, String version)
```

## 3.3.3 获取小程序信息

### 3.3.3.1 获取最近访问小程序列表

```

/**
 * 获取最近访问小程序列表
 * @param callback
 */
public static void
getRecentList(IRecentMiniCallback callback)

```

### 3.3.3.2 搜索正式小程序

```

/**
 * 小程序搜索
 *
 * @param searchOptions
 * @param callback
 */
public static void
searchMiniApp(SearchOptions searchOptions,
MiniCallback<List<MiniApp>> callback)

```

### 3.3.4 自定义JSAPI

```

@jsPlugin(secondary = true)
public class CustomPlugin extends
BaseJsPlugin {
    @JsEvent("custom_event")
    public void custom(final RequestEvent
req) {
        //获取参数
        //req.jsonParams
    }
}

```

```

        //异步返回数据
        //req.fail();
        //req.ok();
        req.ok(new JSONObject());
    }

    @JsEvent({"getSystemInfo",
"getSystemInfoSync"})
    public String custom1(final RequestEvent
req) {
        //获取参数
        //req.jsonParams
        //同步返回数据(必须返回json数据)
        return new JSONObject().toString();
    }
}

```

- 继承BaseJsPlugin并用注解进行定义  
@JsPlugin(secondary = true)
- 定义一个方法，方法只能有一个参数且参数必须是RequestEvent类型
- 然后在方法上定义注解@JsEvent("事件名")，当小程序js调用“事件名”时就会调用到@JsEvent修饰的对应方法
- @JsEvent支持定义多个事件名
- 支持同步或异步返回数据（同一事件只能选择一种方式）

### 3.3.5 小程序宿主自定义

通过如下设置，宿主需要自定义小程序的一些定制化功能

```
@ProxyService(proxy = MiniAppProxy.class)
public class MiniAppProxyImpl extends
BaseMiniAppProxy{}
```

定义实现类并继承BaseMiniAppProxyImpl，并使用上面的注解进行修饰。

### 3.3.5.1 自定义胶囊

```
/**
 * 点击胶囊按钮的关闭选项
 *
 * @param miniAppContext 小程序运行环境(小程序
进程，非主进程)
 * @param onCloseClickedListener 点击小程序关
闭时回调
 * @return 不支持该接口，请返回false
 */
public abstract boolean
onCapsuleButtonCloseClick(IMiniAppContext
miniAppContext,
        DialogInterface.OnClickListener
onCloseClickedListener);

/**
 * 返回胶囊更多面板的按钮，扩展按钮的ID需要设置为
[100, 200]这个区间中的值，否则，添加无效
 * @param builder
```

```

    * @return
    */
    public abstract ArrayList<MoreItem>
    getMoreItems(MoreItemList.Builder builder);

    /**
     * 返回胶囊更多面板按钮点击监听器
     *
     * @return 监听器
     */
    public abstract OnMoreItemSelectedListener
    getMoreItemSelectedListener();

```

### 3.3.5.2 相册选择与图片预览

```

    /**
     * 打开选图界面
     *
     * @param context 当前Activity
     * @param maxSelectedNum 允许选择的最大数量
     * @param listner 回调接口
     * @return 不支持该接口，请返回false
     */
    public abstract boolean
    openChoosePhotoActivity(Context context, int
    maxSelectedNum, IChoosePhotoListner
    listner);

    /**

```

```

* 打开图片预览界面
*
* @param context 当前Activity
* @param selectedIndex 当前选择的图片索引
* @param pathList 图片路径列表
* @return 不支持该接口，请返回false
*/
public abstract boolean
openImagePreview(Context context, int
selectedIndex, List<String> pathList);

```

### 3.3.5.3 自定义授权UI

当小程序调用的API需要授权时，SDK提供如下默认的授权UI样式，开发者也可以通过如下方法自定义授权UI样式

```

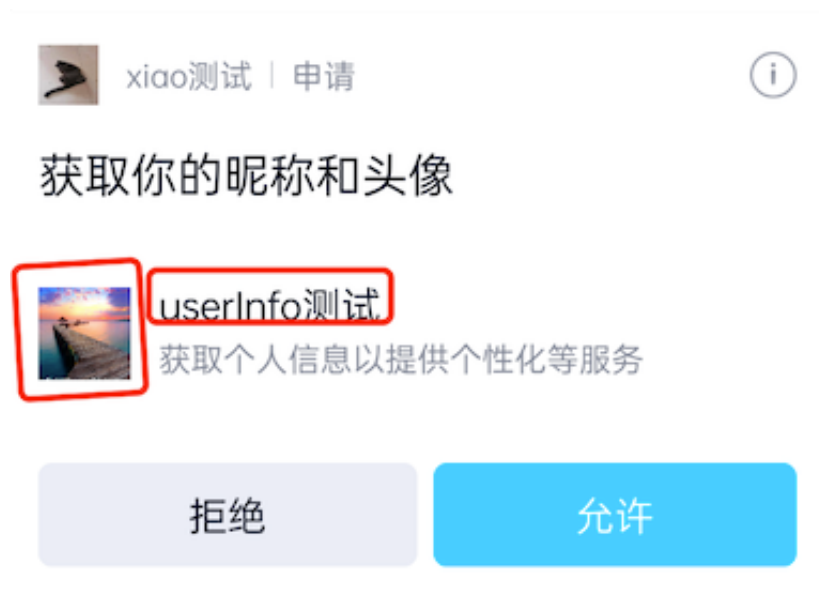
/**
* 自定义授权弹窗view
* @param context
* @param authInfo
* @param authView
* @return true:自定义授权view;false:使用内置
*/
@Override
public boolean authView(Context context,
MiniAuthInfo authInfo, IAuthView authView) {
    return true;
}

```



### 3.3.5.4 自定义授权用户信息

实现BaseMiniAppProxyImpl如下代码,可以自定义用户授权信息中用户昵称和头像



```
/**
 * 获取scope.userInfo授权用户信息
 * @param appId
 * @param result
 */
@Override
public void getUserInfo(String appId,
AsyncResult result) {
    JSONObject jsonObject = new
JSONObject();
    try {
        //返回昵称
        jsonObject.put("nickName", "userInfo
测试");
    }
```

```
//返回头像url
        jsonObject.put("avatarUrl",
            "https://gimg2.baidu.com/image_search/src=ht
            tp%3A%2F%2Fimg.daimg.com%2Fuploads%2Fallimg%
            2F210114%2F1-
            210114151951.jpg&refer=http%3A%2F%2Fimg.daim
            g.com&app=2002&size=f9999,10000&q=a80&n=0&g=
            0n&fmt=auto?
            sec=1673852149&t=e2a830d9fabd7e0818059d92c38
            83017");
        result.onReceiveResult(true,
            jsonObject);
    } catch (JSONException e) {
        e.printStackTrace();
    }
}
```

为了加载头像，还需要实现如下方法

```
public Drawable getDrawable(Context context,
    String source, int width, int height,
    Drawable defaultDrawable)
```

### 3.3.5.5 小程序数据根据账号隔离存储

```
/**
 * 用户账号,必须唯一, 设置后数据会按账号隔离存储,不
 * 建议使用用户敏感信息.(主进程环境执行)
 */
@Override
public String getAccount() {
    return "tmf_test";
}
```

### 3.3.6 分享

可以在胶囊菜单中设置微信、QQ或自定义分享事件，然后按照如下类型接收分享点击事件，在share方法中调用第三方SDK实现分享

```
@ProxyService(proxy = ShareProxy.class)
public class ShareProxyImpl extends
BaseShareProxy {
    /**
     * 分享
     *
     * @param shareData 分享数据
     */
    @Override
    public void share(Activity activity,
ShareData shareData) {
    }
}
```



注意：如果在胶囊菜单自定义按钮，参考"自定义胶囊"小节

### 3.3.7 微信小程序事件

微信小程序如下方法与Native对应关系，当小程序中调用对应方法时，会调用到native对应事件，开发者需要监听事件并返回数据

微信方法	native事件
wx.login	wx.login
wx.getUserInfo	wx.getUserInfo
wx.getUserProfile	wx.getUserProfile

```
@JsPlugin(secondary = true)
public class WxApiPlugin extends
BaseJsPlugin {
    /**
     * 对应小程序wx.login调用
     * @param req
     */
    @JsEvent("wx.login")
    public void login(final RequestEvent
req) {
        //获取参数
        //req.jsonParams
        //异步返回数据
        //req.fail();
        //req.ok();
        JSONObject jsonObject = new
JSONObject();
        try {
            jsonObject.put("key",
"wx.login");
        } catch (JSONException e) {
```

```

        e.printStackTrace();
    }
    req.ok(jsonObject);
}

/**
 * 对应小程序wx.getUserInfo调用
 * @param req
 */
@JsEvent("wx.getUserInfo")
public void getUserInfo(final
RequestEvent req) {
    //获取参数
    //req.jsonParams
    //异步返回数据
    //req.fail();
    //req.ok();
    JSONObject jsonObject = new
JSONObject();
    try {
        jsonObject.put("key",
"wx.getUserInfo");
    } catch (JSONException e) {
        e.printStackTrace();
    }
    req.ok(jsonObject);
}

/**

```

```

    * 对应小程序wx.getUserProfile调用
    * @param req
    */
    @JsEvent("wx.getUserProfile")
    public void getUserProfile(final
RequestEvent req) {
        //获取参数
        //req.jsonParams
        //异步返回数据
        //req.fail();
        //req.ok();
        JSONObject jsonObject = new
JSONObject();
        try {
            jsonObject.put("key",
"wx.getUserProfile");
        } catch (JSONException e) {
            e.printStackTrace();
        }
        req.ok(jsonObject);
    }
}

```

### 3.3.8 自定义权限列表

调用TmfMiniSDK如下方法获取授权列表

```

/**
 * 获取小程序授权列表

```

```
* @param appId
* @param appVerType 小程序版本类型
* @return
*/
public static List<MiniAuthState>
getAuthStateList(String appId, int
appVerType)

/**
* 设置授权状态
* @param appId
* @param appVerType 小程序版本类型
* @param scopeName 权限名
* @param grant 是否授权
*/
public static void setAuthState(String
appId, int appVerType, String scopeName,
boolean grant)
```

## 4 API

---

### 4.1 MiniCode

返回错误码描述

```
/**
* 成功
*/
```



```
public static final int CODE_OK = 0;

//////////////////////////////////////扫码解析
code//////////////////////////////////////
/**
 * requestCode不符
 */
public static final int
CODE_REQUEST_CODE_ERROR = -10000;
/**
 * 扫码返回Intent空
 */
public static final int
CODE_QRCODE_INTENT_NULL = -10001;
/**
 * qrcode空
 */
public static final int CODE_QRCODE_NULL =
-10002;
/**
 * 非tmf小程序二维码
 */
public static final int
CODE_QRCODE_INVALIDATE = -10003;
/**
 * 二维码内容格式错误
 */
public static final int
CODE_QRCODE_FORMAT_ERROR = -10004;
```

```
/**
 * appId空
 */
public static final int CODE_APPID_EMPTY =
-10005;
/**
 * 二维码解析异常
 */
public static final int CODE_EXCEPTION =
-10006;
/**
 * 未找到扫码服务，请检查是否添加了qrcode相关sdk
 */
public static final int
CODE_NOT_FOUND_QRCODE_SERVICE = -10007;
/**
 * http小程序二维码解析错误
 */
public static final int
CODE_HTTP_QRCODE_PARSE_ERROR = -10008;
/**
 * bad query result
 */
public static final int
CODE_HTTP_QRCODE_BAD_QUERY_RESULT = -10009;
/**
 * exception
 */
```

```
public static final int
CODE_HTTP_QRCODE_EXCEPTION = -10010;
/**
 * returnCode error
 */
public static final int
CODE_HTTP_QRCODE_RETURNCODE_ERROR = -10011;
/**
 * data null
 */
public static final int
CODE_HTTP_QRCODE_DATA_NULL = -10012;
/**
 * shark null
 */
public static final int
CODE_HTTP_QRCODE_SHARK_NULL = -10013;
/**
 * businessId null
 */
public static final int
CODE_HTTP_QRCODE_BUSINESSID_NULL = -10014;

////////////////////////////////////小程序信息获取
code////////////////////////////////////
/**
 * 登录类型错误
 */
```

```
public static final int
CODE_APPLET_INFO_LOGINTYPE_ERROR = -10100;

/**
 * 调试小程序只能使用开发者账号
 */
public static final int
CODE_APPLET_INFO_OPERATE_ACCOUNT_PREVIEW_ERROR = -10101;

/**
 * 登录信息空
 */
public static final int
CODE_APPLET_INFO_LOGIN_INFO_EMPTY = -10102;

/**
 * create req error
 */
public static final int
CODE_APPLET_INFO_JSON_EXCEPTION = -10103;

/**
 * shark error
 */
public static final int
CODE_APPLET_INFO_SHARK_ERROR = -10104;

/**
```

```

    * create MiniAppInfo ByQrCode error
    */
public static final int
CODE_APPLET_INFO_CREATE_MINIAPPINFO_ERROR =
-10105;

////////////////////////////////////小程序检查更新
code////////////////////////////////////
/**
    * 小程序下架或无可用小程序
    */
public static final int
STATUS_CODE_SERVER_REQUEST_DELETE = 10200;
/**
    * 无更新
    */
public static final int
STATUS_CODE_NO_UPDATE = 10201;
/**
    * shark实例空
    */
public static final int
STATUS_CODE_SHARK_IS_NULL = -10202;
/**
    * 网络错误
    */
public static final int
STATUS_CODE_NETWORK_CHANNEL_ERROR = -10203;
/**

```

```

    * 检查更新返回类型错误
    */
public static final int
STATUS_CODE_UPDATE_TYPE_ERROR = -10204;

/**
    * 检查更新异常
    */
public static final int
STATUS_CODE_UPDATE_EXCEPTION = -10205;

////////////////////////////////////小程序下载
code////////////////////////////////////
/**
    * 小程序下载信息错误
    */
public static final int
STATUS_CODE_FILE_INFO_ERROR = -10300;

/**
    * 小程序包下载catch异常
    */
public static final int
STATUS_CODE_DOWNLOAD_EXCEPTION = -10301;

/**
    * 下载文件不存在
    */
public static final int
STATUS_CODE_FILE_NOT_EXIST_ERROR = -10302;

```

```
/**
 * MD5校验失败
 */
public static final int
STATUS_CODE_MD5_ERROR = -10303;
/**
 * zip解压错误
 */
public static final int
STATUS_CODE_UNZIP_ERROR = -10304;
/**
 * 移动文件过程文件不存在错误
 */
public static final int
STATUS_CODE_TMFAPKG_FILE_NOT_EXIST_ERROR =
-10305;
/**
 * 移动文件失败
 */
public static final int
STATUS_CODE_MOVE_ERROR = -10306;
/**
 * 小程序包下载后解析catch异常
 */
public static final int
STATUS_CODE_PARSE_PKG_EXCEPTION = -10307;
/**
 * 文件下载器下载失败
 */
```

```
public static final int
STATUS_CODE_DOWNLOAD_ERROR = -10308;

public static final int
STATUS_CODE_UPDATE_EXCEPTION_ERROR = -10309;

//////////////////////////登录和登出
code////////////////////////////////////
/**
 * json异常
 */
public static final int CODE_JSON_EXCEPTION
= -10400;
/**
 * 网关失败
 */
public static final int CODE_SHARK_FAIL =
-10401;
/**
 * 网关返回数据为空
 */
public static final int CODE_RESP_NULL =
-10402;
/**
 * 开放平台登录返回错误
 */
public static final int
CODE_OPEN_LOGIN_RESP_ERROR = -10403;
```



```

/**
 * 开放平台登录返回数据空
 */
public static final int
CODE_OPEN_LOGIN_DATA_NULL = -10404;
/**
 * 开放平台登录返回租户数据空
 */
public static final int
CODE_OPEN_LOGIN_TENANT_NULL = -10405;
/**
 * 运行平台登录返回数据空
 */
public static final int
CODE_OPERATE_LOGIN_DATA_NULL = -10406;
/**
 * logout异常
 */
public static final int
CODE_LOGOUT_EXCEPTION = -10407;
/**
 * 服务端json数据解析异常
 */
public static final int
CODE_PARSE_JSON_EXCEPTION = -10408;

////////////////////////////////////////搜索
code////////////////////////////////////////
/**

```

```
    * 搜索创建req json异常
    */
public static final int
CODE_SEARCH_JSON_EXCEPTION = -10500;
/**
    * 搜索resp为空
    */
public static final int
CODE_SEARCH_RESP_NULL = -10501;
/**
    * 搜索resp.data为空
    */
public static final int
CODE_SEARCH_RESP_DATA_NULL = -10502;
/**
    * 搜索解析resp.data为错误
    */
public static final int
CODE_SEARCH_PARSE_RESP_ERROR = -10503;
/**
    * 搜索网路retCode.errorCode错误
    */
public static final int
CODE_SEARCH_SHARK_ERROR = -10504;
/**
    * 搜索返回JSON_EXCEPTION
    */
```

```

public static final int
CODE_SEARCH_RESP_DATA_JSON_EXCEPTION =
-10505;

////////////////////////////////////启动mini
app////////////////////////////////////
/**
 * doStartMiniApp exception
 */
public static final int
CODE_DO_START_MINI_APP_THROWABLE = -10600;

```

## 4.2 MiniApp

### 小程序信息描述类

```

/**
 * 正式小程序
 */
public static final int TYPE_ONLINE =
MiniSDKConst.ONLINE;
/**
 * 调试小程序
 */
public static final int TYPE_DEVELOP =
MiniSDKConst.DEVELOP;
/**
 * 预览小程序
 */

```

```
public static final int TYPE_PREVIEW =
MiniSDKConst.PREVIEW;
/**
 * 小程序id
 */
public String appId;
/**
 * 小程序版本类型(正式、预览、开发版)
 */
public int appVerType;
/**
 * 小程序版本
 */
public String version;
/**
 * 小程序名
 */
public String name;
/**
 * 小程序图标
 */
public String iconUrl;
/**
 * 小程序简介
 */
public String appIntro;
/**
 * 开发者企业名称
 */
```

```
public String appDeveloper;
/**
 * 时间戳
 */
public long time;
```

## 4.3 MiniStartOptions

```
/**
 * 打开小程序时是否强制检查更新，false: 优先使用本地
缓存，同时异步获取最新数据；true: 待网络返回后才打开
小程序
 */
public boolean isForceUpdate = false;
/**
 * 入口地址
 */
public String entryPath;
/**
 * 接受小程序启动过程中错误情况
 */
public ResultReceiver resultReceiver;
```

## 4.4 MiniScene

```
/**
 * 小程序主入口, 「最近使用」 列表
 */
public static final int
LAUNCH_SCENE_MAIN_ENTRY = 1001;
/**
 * 扫码打开
 */
public static final int
LAUNCH_SCENE_QR_CODE_FROM_SCAN = 1011;
/**
 * 搜索打开
 */
public static final int LAUNCH_SCENE_SEARCH
= 2005;
```

## 4.5 SearchOptions

```
/**
 * 搜索关键字，为空时搜索全部小程序
 */
public String keyWord = "";
/**
 * 暂不支持
 */
public int pageIndex;
/**
 * 暂不支持
 */
public int pageSize;
```

## 5 错误总结

### 5.1 小程序内webview使用

当小程序页面中直接使用webview加载H5页面时，如果页面中有错误或页面中资源有HTTP请求，如下：

```
11757-11757 E/[TMF_MINI]InnerWebView: evaluateJavascript miniAppWebViewStr callback
11757-11757 E/[TMF_MINI]InnerWebView.js: Mixed Content: The page at 'https://wx.vzan.com/plugin-ins/?v=638877544588943418#/FixupIndex/144679432?shareuid=0' was loaded over HTTPS,
but requested an insecure image 'http://static1.weizan.cn/zhibo/livecontent/ent_tv/images/vzanqrcode.jpg'. This content should also be served over HTTPS.
```

加载过程中可能会出现如下错误页面



## 网页无法打开

位于 <https://appservice.qq.com/page-frame.html> 的网页无法加载，因为：

net::ERR\_INVALID\_ARGUMENT