

1. 接入前项目检查

根据游戏接入后出现的问题，对游戏项目做以下几点要求：

- a) AndroidManifest.xml 中 android:targetSdkVersion="19"
- b) 如果 AndroidManifest.xml 有 installLocation 选项时，参数设置为 auto
- c) AndroidManifest.xml 中的 Activity 需要进行如下配置
android:configChanges="keyboardHidden|orientation|screenSize"
- d) 根据游戏的横竖屏，将 activity 做横竖屏设置
android:screenOrientation="landscape"
android:screenOrientation="portrait"
- e) AndroidManifest.xml 中的 Activity 等组件路径为完整路径。

2. 接入流程

2.1 添加资源

下载 SDK 资源，将 files 文件夹下的文件 assets、libs 和 res 拷贝到项目对应的文件夹中。

2.2 修改 AndroidManifest

添加权限

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

添加 Application

在 AndroidManifest.xml 中的 Application 节点，修改 android:name 属性值为

com.zmapp.mzsdk.MZApplication。如果游戏有自己的 Application 需求，需要在 Application 的生命周期方法中做一些操作。可以实现 IApplicationListener 里面的接口。

添加 meta-data

```
<meta-data
    android:name="MZ_APP_ID"
    android:value="联系商务(不填或者错误无法获取登陆信息)" />
```

2.3 生命周期处理

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    com.zmapp.mzsdk.MZSDK.getInstance().onCreate();
}
@Override
protected void onStart() {
    super.onStart();
    com.zmapp.mzsdk.MZSDK.getInstance().onStart();
}
@Override
```

```
protected void onRestart() {
    super.onRestart();
    com.zmapp.mzsdm.MZSDK.getInstance().onRestart();
}
@Override
protected void onPause() {
    super.onPause();
    com.zmapp.mzsdm.MZSDK.getInstance().onPause();
}
@Override
protected void onResume() {
    super.onResume();
    com.zmapp.mzsdm.MZSDK.getInstance().onResume();
}
@Override
protected void onStop() {
    super.onStop();
    com.zmapp.mzsdm.MZSDK.getInstance().onStop();
}
@Override
protected void onDestroy() {
    super.onDestroy();
    com.zmapp.mzsdm.MZSDK.getInstance().onDestroy();
}
@Override
protected void onNewIntent(Intent intent) {
    super.onNewIntent(intent);
    com.zmapp.mzsdm.MZSDK.getInstance().onNewIntent(intent);
}
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    com.zmapp.mzsdm.MZSDK.getInstance().onActivityResult(this, requestCode, resultCode, data);
}
```

2.4 SDK 初始化设置

2.4.1 设置初始化回调接口

```
MZSDK.getInstance().setSDKInitListener(new IMZSDKInitListener() {
    @Override
    public void onInitSuccess(final InitResult result) {
    }

    @Override
    public void onInitFail(final int code, final String msg) {
```

```
        //code:初始化失败错误码, msg:初始化失败错误信息
    }
});
```

2.4.2 设置用户回调接口

```
MZSDK.getInstance().setSDKUserListener(new IMZSDKUserListener() {
    @Override
    public void onLoginFail(final int code, final String msg) {
        //code:登录失败错误码, msg:登录失败错误信息
    }
    @Override
    public void onLoginSuccess(final String userInfoJJsonStr) {
        //登录成功, userInfoJJsonStr 参数说明参照下面表格
    }
    @Override
    public void onSwitchAccount() {
    }
    @Override
    public void onSwitchAccount(final String userInfoJJsonStr) {
        //切换账号成功, userInfoJJsonStr 参数说明参照下面表格
    }
    @Override
    public void onLogout() {
    }
});
```

userInfoJJsonStr 是返回的用户信息，是 JSON 字符串格式：

status	"0" 表示成功获取到了用户信息，其它值都为失败	
userdata	uid	用户唯一码
	t	时间戳，数字字符串
	sign	校验方法是：sign=md5(secret_key=1&t=2&uid=3)，参数首字母升序排列，md5 是 32 小写

2.4.3 设置支付回调接口

```
MZSDK.getInstance().setSDKPayListener(new IMZSDKPayListener() {
    @Override
    public void onPaySuccess(final PayResult result) {
        //支付成功
        //道具计费点编号: result.getProductID()
        //道具名称: result.getProductName()
        //透传字符串: result.getExtension()
    }
    @Override
    public void onPayFail(final int code, final String msg) {
```

```
        //code: 支付失败错误码, msg: 支付失败错误信息  
    }  
});
```

接入要求：客户端的支付回调仅作为通知使用，以服务端的支付回调为最终发货依据。

2.5 SDK 接口调用

接入要求：SDK 的接口需在 UI 主线程下调用

2.5.1 SDK 初始化

类名：com.zmapp.mzsdk.MZSDK

方法：public void init(Activity activity)

功能：渠道 SDK 初始化

参数：activity 必填 当前 Activity

案例：

```
com.zmapp.mzsdk.MZSDK.getInstance().init(activity);
```

接入要求：初始化接口，应在 Activity 中 onCreate 方法中调用。

2.5.2 登录

类名：com.zmapp.mzsdk.MZUser

方法：public void login()

功能：渠道 SDK 登录

案例：

```
com.zmapp.mzsdk.MZUser.getInstance().login();
```

2.5.3 支付

类名：com.zmapp.mzsdk.MZPay

方法：public void pay()

功能：渠道 SDK 支付

参数：params 必填 订单信息

案例：

```
PayParams params = new PayParams();  
params.setProductId("1");//productId  
params.setProductName("道具 1");//productName  
params.setProductDesc("道具 1 的描述");//productDesc  
params.setPrice(100);//price  
params.setRatio(1);//ratio  
params.setServerName("服务器 1");//serverName  
params.setServerId("123");//数字字符串  
params.setRoleId("123456");//数字字符串  
params.setRoleName("速易大侠");//角色名称  
params.setRoleLevel(100);//角色等级  
params.setVip("0");//默认值是 0  
params.setExtension("透传参数");  
params.setCheck("服务端生成签名");  
MZPay.getInstance().pay(params);
```

支付信息参数 PayParams 的格式说明：

字段	类型	是否必填	说明
----	----	------	----

productID	String	是	道具计费点编号, 数字字符串
productName	String	是	道具名称
productDesc	String	否	道具描述
price	int	是	道具价格, 单位为分
serverName	String	是	服务器名称
serverID	String	是	服务器编号, 数字字符串
roleID	String	是	角色编号, 数字字符串
roleName	String	是	角色名称
roleLevel	int	否	角色等级
VIP	String	否	角色 VIP 等级
extension	String	否	透传字符串, 支付回调会原样返回给游戏
check	String	是	check=md5(fee+feeid+seceret_key), 其中 md5 为小写, “+” 不参与加密。注意加密必须在服务端完成。check 生成示例: fee=600, feeid=123, secret_key=abcde123456, check=31a482a2b53dde3bdb30815b6af79b72

2.5.4 注销

类名: com.zmapp.mzsdk.MZUser

方法: public void logout()

功能: 渠道 SDK 登录

案例:

```
com.zmapp.mzsdk.MZUser.getInstance().logout();
```

2.5.5 退出

类名: com.zmapp.mzsdk.MZUser

方法: public void exit()

功能: 渠道 SDK 退出

案例:

```
MZPay.getInstance().pay(params);
if (MZUser.getInstance().isSupport("exit")) {
    MZSDK.getInstance().runOnMainThread(new Runnable() {
        @Override
        public void run() {
            MZUser.getInstance().exit();
        }
    });
} else {
    // 游戏自己的退出确认框
}
```

2.5.6 角色信息上报

类名: com.zmapp.mzsdk.MZUser

方法: public void exit()

功能：渠道 SDK 角色信息上报

案例：

```

UserExtraData extraData = new UserExtraData();
extraData.setDataType(1);
extraData.setServerID(10);
extraData.setServerName("server_10");
extraData.setRoleID("role_100");
extraData.setRoleName("test_112");
extraData.setRoleLevel("10");
extraData.setMoneyNum(100);
extraData.setPartyName("帮派");
extraData.setRoleCTime("1510759822");
extraData.setRoleLevelMTime("1510771935");
MZUser.getInstance().submitExtraData(extraData);
    
```

角色信息参数表:

UserExtraData

字段	类型	是否必填	说明
dataType	int	是	1.选择服务器 2.创建角色 3.进入游戏 4.等级提升 5.退出游戏
serverID	int	是	服务器 ID
serverName	String	是	服务器名称
roleID	String	是	角色 ID
roleName	String	是	角色名称
roleLevel	String	是	角色等级
fightValue	String	是	战力值
moneyNum	int	否	游戏币
partyName	String	否	所在帮派名称
roleCreateTime	String	否	角色创建时间, 10 位整数字符串, 秒为单位, 格式参照 http://tool.chinaz.com/Tools/unixtime.aspx?qq-pf-to=pcqq.group
roleLevelMTime	String	否	角色升级时间, 10 位整数字符串, 秒为单位, 格式参照 http://tool.chinaz.com/Tools/unixtime.aspx?qq-pf-to=pcqq.group
gender	String	否	角色性别, 可传 "男"、"女"、"无"
professionID		否	职业 ID
profession	String	否	职业名称
VIP		否	VIP 等级
partyID		否	所在帮派 ID
partyName	String	否	所在帮派名称
partyroleid		否	帮派称号 id
partyrolename	String	否	帮派称号名称
friendlist	String	否	角色相互关系, 例如[{"roleid": "关系角色 id","intimacy": "亲密度", "nexusid": "关系 id,可填数字 1:夫妻 2:结拜 3:情侣 4:师徒 5:仇人 6:其它"}...]

接入要求：SDK 会根据渠道不同进行上报，渠道存在不上报不给上线情况，如果个别参数没有传空值，建议都传值。

3. 其它

3.1 关于混淆

SDK 已经经过了混淆，如果要混淆 java 代码，请不要混淆联编的 jar 包中的类。可以添加以下类到 proguard 配置，排除在混淆之外：

```
-keep class com.zmapp.** { *; }
```