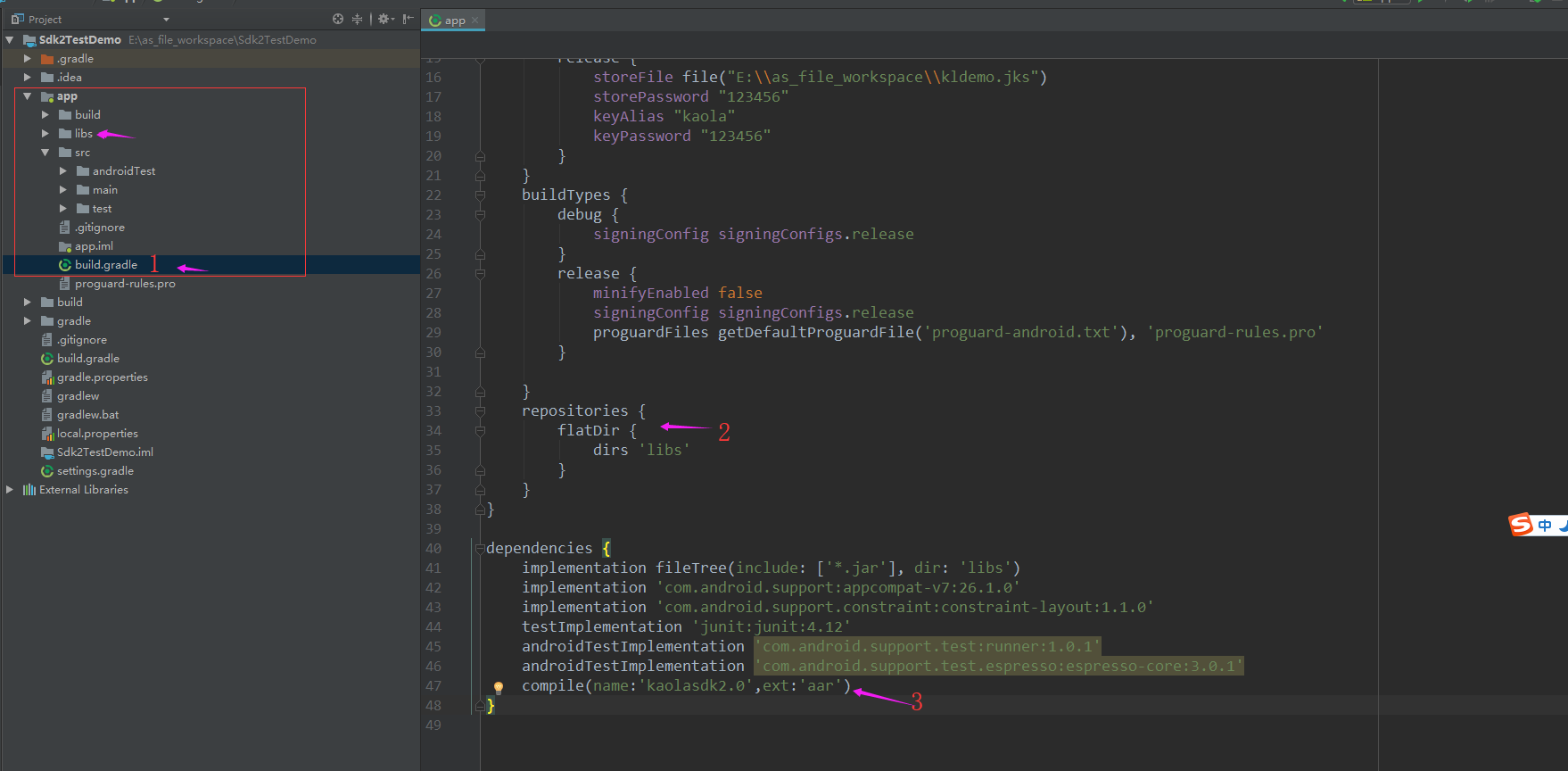
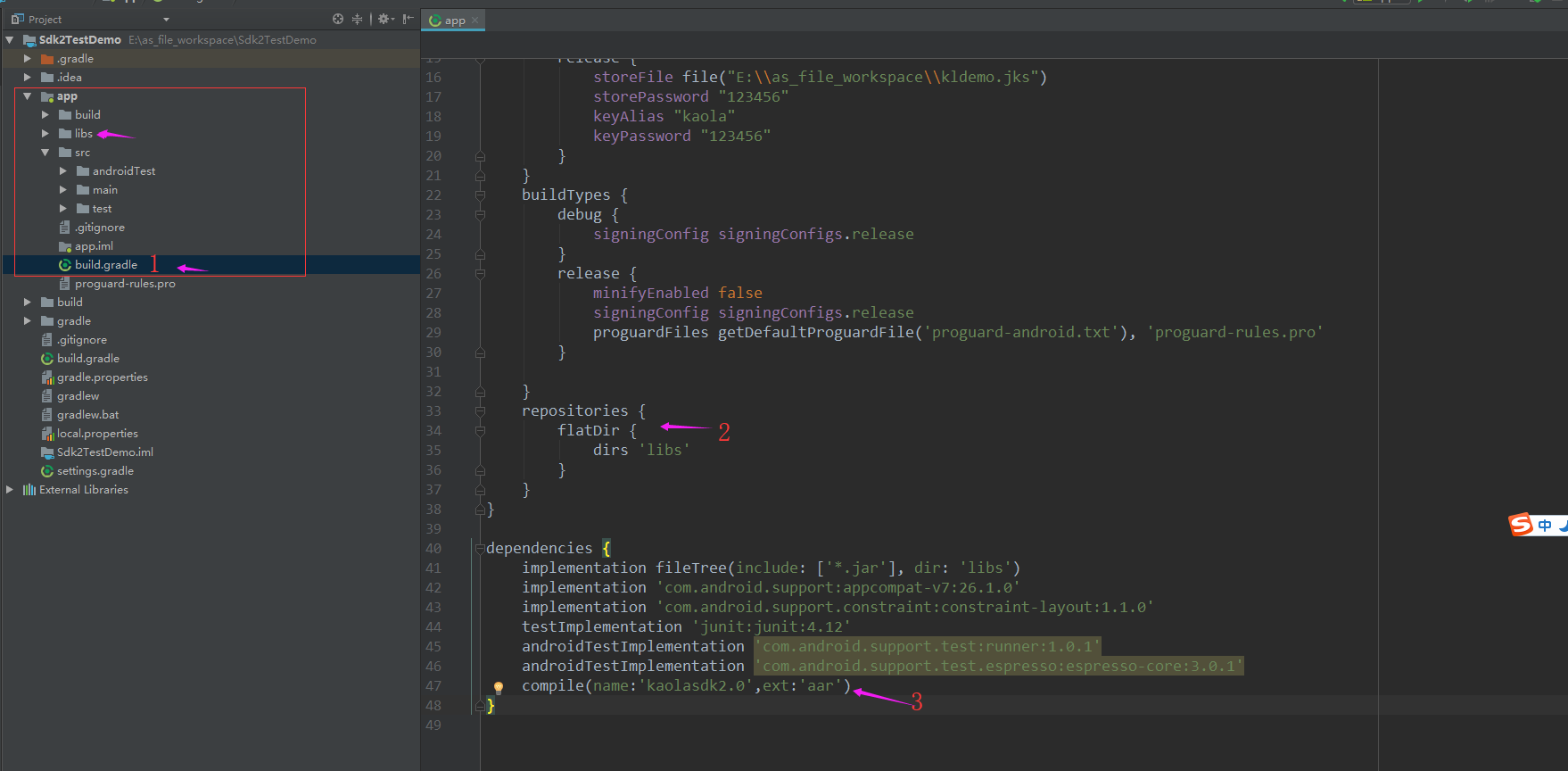
**sdk对接文档**

1. 接入流程
2. 生命周期接口
3. 初始化接口🡪登录接口（初始化成功后调用）
4. 角色信息收集接口
5. 切换账号/登出监听器
6. 切换账号/登出接口
7. 充值接口
8. 退出接口
9. 关于库，lib和其他配置文件的引入
10. android studio打开Sdk2TestDemo，作为接入参考



1. 关于工程中需要引入的库





1. 关于功能清单文件（其中的配置信息，请拷贝到自己工程中）

权限配置

|  |
| --- |
| <uses-permission android:name="android.permission.INTERNET" />  <uses-permission android:name="android.permission.CALL\_PHONE" />  <uses-permission android:name="android.permission.SEND\_SMS" />  <uses-permission android:name="android.permission.READ\_SMS" />  <uses-permission android:name="android.permission.WRITE\_SMS" />  <uses-permission android:name="android.permission.GET\_TASKS" />  <uses-permission android:name="android.permission.REORDER\_TASKS" />  <uses-permission android:name="android.permission.READ\_PHONE\_STATE" />  <uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />  <uses-permission android:name="android.permission.RECEIVE\_BOOT\_COMPLETED" />  <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />  <uses-permission android:name="android.permission.CHANGE\_NETWORK\_STATE" />  <uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />  <uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />  <uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE" />  <uses-permission android:name="android.permission.READ\_LOGS" />  <uses-permission android:name="android.permission.SYSTEM\_ALERT\_WINDOW" /> |

修改Application

1. 若无自定义的Application,则修改AndroidManifest.xml的Application，如下

|  |
| --- |
| <application android:name=*"com.klsdk.common.KLApplication"*> |

1. 若开发者自定义

application，则自定义的application需继*com.klsdk.common.KLApplication，AndroidManifest.xml修改如下*

|  |
| --- |
| <application android:name=*”自定义Application"*> |

Activity、service、receiver声明

|  |
| --- |
| <activity  android:name="com.klsdk.activity.KLLoginActivity"  android:configChanges="orientation|keyboardHidden|screenSize"  android:screenOrientation="landscape"  android:theme="@style/kl\_Transparent" />  <activity  android:name="com.klsdk.activity.KLRegisterActivity"  android:configChanges="orientation|keyboardHidden|screenSize"  android:screenOrientation="landscape"  android:theme="@style/kl\_Transparent" />  <activity  android:name="com.klsdk.activity.KLPaymentActivity"  android:configChanges="orientation|keyboardHidden|screenSize"  android:screenOrientation="landscape"  android:theme="@style/kl\_Transparent" />  <activity  android:name="com.klsdk.activity.KLpayWebActivity"  android:configChanges="orientation|keyboardHidden|screenSize"  android:screenOrientation="landscape"  android:theme="@style/kl\_Transparent" />  <activity  android:name="com.klsdk.activity.KLUserinfoActivity"  android:configChanges="orientation|keyboardHidden|screenSize"  android:screenOrientation="landscape"  android:theme="@style/kl\_Transparent" />  <activity  android:name="com.klsdk.activity.KLForgetpasswordActivity"  android:configChanges="orientation|keyboardHidden|screenSize"  android:screenOrientation="landscape"  android:theme="@style/kl\_Transparent" />  <service android:name="com.klsdk.push.PushService" />  <!-- 开机启动监听 -->  <receiver android:name="com.klsdk.push.BootReceive" >  <intent-filter android:priority="1000" >  <action android:name="android.intent.action.BOOT\_COMPLETED" />  <category android:name="android.intent.category.LAUNCHER" />  <action android:name="android.intent.action.USER\_PRESENT" />  </intent-filter>  </receiver> |

1. 生命周期接口（必接）

代码如下：

|  |
| --- |
| @Override  **Protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);    KLSDK.*onCreate*(**this**);  }  @Override  **Protected void** onActivityResult(**int**requestCode, **int**resultCode, Intent data) {  // **TODO** Auto-generated method stub  **super**.onActivityResult(requestCode, resultCode, data);  KLSDK.*onActivityResult*(MainActivity.**this**, requestCode, resultCode, data);  }  @Override  **protected void** onDestroy() {  // **TODO** Auto-generated method stub  **super**.onDestroy();  KLSDK.*onDestroy*(**this**);  }  @Override  **Protected void** onPause() {  // **TODO** Auto-generated method stub  **super**.onPause();  KLSDK.*onPause*(**this**);  }  @Override  **Protected void** onRestart() {  // **TODO** Auto-generated method stub  **super**.onRestart();  KLSDK.*onRestart*(**this**);  }  @Override  **Protected void** onResume() {  // **TODO** Auto-generated method stub  **super**.onResume();  KLSDK.*onResume*(**this**);  }  @Override  **Protected void** onStop() {  // **TODO** Auto-generated method stub  **super**.onStop();  KLSDK.onStop (**this**);  }  @Override  **Protected void** onNewIntent(Intent intent) {  // **TODO** Auto-generated method stub  **super**.onNewIntent(intent);  KLSDK.*onNewIntent*(intent);  } |

1. 初始化接口（必接）
2. 代码示例

|  |
| --- |
| // 初始化  KLSDK.*initInterface*(MainActivity.**this**, appid, appkey,  **New** InitListener() {  @Override  **Public void** fail(String msg) {  // **TODO** Auto-generated method stub    }  @Override  **Public void** Success(String msg) {  // **TODO** Auto-generated method stub    }  }); |

1. 参数说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **参数** | **参数名称** | **类型** | **参数说明** | **是否为空** |
| Context | 上下文对象 | Context | 上下文对象 | 否 |
| appId | 游戏id | int | 游戏方的id | 否 |
| appKey | 游戏密钥 | String | 加密密钥 | 否 |
| InitListener | 回调接口 |  | 初始化回调 | 否 |

注：Appid, Appkey：在接入游戏的时候，平台方会发给游戏方

1. 接口描述

接口会返回两种结果

Success的方法体返回的信息是success和update

success，证明初始化成功

update，则需要游戏等待，正在更新游戏

Fail的方法体返回的信息是fail;

Fail说明初始化不成功

1. 登录接口（必接）

注：必须初始化成功后调用

1. 代码示例

|  |
| --- |
| // 登录  KLSDK.*login*(MainActivity.**this**, appid, appkey, **new** ApiListenerInfo() {  @Override  **Public void** onSuccess(Object obj) {  // **TODO** Auto-generated method stub  **if** (obj != **null**) {  LoginMessageinfo data = (LoginMessageinfo) obj;  String result = data.getResult();  String msg = data.getMsg();  String gametoken = data.getGametoken();  String time = data.getTime();  String uid = data.getUid();  String sessid = data.getSessid();  Log.*i*("kk", "登录结果:" + result + "|msg" + msg  + "|gametoken" + gametoken + "" + "|time"  + time + "|uid" + uid + "|sessid" + sessid);  }  }  }); |

1. 参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **参数** | **参数名称** | **类型** | **参数说明** | **是否为空** | **实例** |
| Appid | appid | int | 游戏id | 否 | 固定 |
| appkey | appkey | String | 游戏密钥 | 否 | 固定 |
| ApiListenerInfo | 回调接口 |  | 登录回调 | 否 |  |

1. 返回结果参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **参数** | **参数名称** | **类型** | **参数说明** | **是否为空** | **实例** |
| result | 登陆结果 | String | success/fail  （成功返回success其他返回fail） | 否 | success/fail |
| msg | 登陆额外信息 | String | 额外信息 | 否 | 提示对应的信息 |
| uid | 用户uid | String | 用户uid | 否 | 10010 |
| tiem | 时间戳 | String | 用在登陆二次验证 | 否 |  |
| gametoken | token | String | 用在登陆二次验证 | 否 |  |
| sessid | 验证id | String | 用在登陆二次验证 | 是 |  |

1. 切换账号/登出监听器（必接）

注：必须保证玩家任何时候点击切换账号按钮，都能正常切换账号

代码示例

|  |
| --- |
| KLSDK.*setUserListener*(**new** UserApiListenerInfo() {  @Override  **Public void** onLogout(Object obj) {  // **TODO** Auto-generated method stub  **super**.onLogout(obj);  // 切换账号，处理自己的逻辑，比如重新登录，进行选服进入游戏    }  }); |

1. 切换账号/登出接口（必接）

说明：用于用户切换账号等事件的处理

a）若游戏中存在登出或者切换帐号的按钮，则可在点击按钮时进行登出接口调用，在登出回调中进行重新登录等操作；

b）若游戏中不存在登出或者切换帐号的按钮时，建议修改游戏添加登出或切换帐号按钮。

代码示例

|  |
| --- |
| KLSDK.switchAccount(); |

1. 角色信息提交接口
2. 代码示例

|  |
| --- |
| //进入服务器  KLSDK.*setExtData*(**this**, "enterServer", "11", "kk", "99", "1", "1区","80", "81", "逍遥", "21322220", "54456588");  //创建角色  KLSDK.*setExtData*(**this**, "createRole", "11", "kk", "99", "1", "1区",  "80", "81", "逍遥", "21322225", "54456890");  //角色升级  KLSDK.*setExtData*(**this**, "levelUp", "11", "kk", "99", "1", "1区",  "80", "8", "逍遥", "21322333", "54457990"); |

1. 参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 参数 | 参数名称 | 类型 | 参数说明 | 是否为空 | 示例 |
| context | 上下文 | Context | 上下文 | 否 | this |
| scene\_Id | 场景 | String | 玩家场景 | 否 | 分别为进入服器(enterServer)、玩家创建用户角色(createRole)、玩家升级(levelUp) |
| roleId | 角色id | String | 角色id | 否 | 112 |
| roleName | 角色名 | String | 角色名 | 否 | 盟友 |
| roleLevel | 角色等级 | String | 角色等级 | 否 | 99 |
| zoneId | 服务器id | String | 服务器id | 否 | 1 |
| zoneName | 服务器名称 | String | 服务器名称 | 否 | 剑魂 |
| balance | 游戏币余额 | String | 游戏币余额 | 是 | 12 |
| Vip | Vip等级 | String | Vip等级 | 否 | 1 |
| partyName | 所属帮派 | String | 所属帮派 | 是 | 醉江湖 |
| roleCtiem | 创建角色时间 | String | 单位为秒，创建角色的时间 | 是 | 21322222 |
| roleLeveMTimE | 角色等级变化时间 | String | 单位为秒，角色等级变化时间 | 是 | 54456588 |

1. 接口描述

请根据相应的scene\_Id（场景）调用接口

scene\_Id 的值请填写对应的值：进入服器=enterServer、玩家创建用户角色=createRole玩家升级=levelUp；

若某些参数获取不到，请填默认值""

1. 充值接口（必接）
2. 代码示例

|  |
| --- |
| PaymentInfo paymentInfo = **new** PaymentInfo();  paymentInfo.setAppid(appid);  paymentInfo.setAppKey(appkey);  paymentInfo.setAgent("");  paymentInfo.setAmount("1");  paymentInfo.setBillno("");  paymentInfo.setExtrainfo("");  paymentInfo.setSubject("元宝");  paymentInfo.setIstest("1");  paymentInfo.setRoleid("1111");  paymentInfo.setRolename("hh");  paymentInfo.setRolelevel("100");  paymentInfo.setServerid("8888");  paymentInfo.setUid("");  KLSDK.*payment*(**this**, paymentInfo, **new** ApiListenerInfo() {  @Override  **Public void** onSuccess(Object obj) {  // **TODO** Auto-generated method stub  **super**.onSuccess(obj);  **if** (obj != **null**) {  Log.*i*("kk", "充值界面关闭" + obj.toString());  }  }  }); |

1. 参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **参数** | **参数名称** | **类型** | **参数说明** | **是否为空** | **实例** |
| setAppid | 游戏id | int | 游戏id | 否 | 固定 |
| setAppkey | 游戏密钥 | String | 游戏密钥 | 否 | 固定 |
| setAgent | 渠道 | String | 渠道 | 是 | “” |
| setBillno | 订单号 | String | 订单号 | 否 | 查看备注 |
| setAmount | 订单金额 | String | 订单金额 | 是 | 查看备注 |
| setSubject | 订单主题 | String | 订单主题 | 否 | 元宝 |
| setExtraInfo | 额外参数 | String | 订单额外信息 | 否 | 自定义 |
| setUid | 用户信息 | String | 用户信息 | 是 | 查看备注 |
| setIsTest | 是否测试 | String | 测试参数 | 否 | “”：正式  1：测试 |
| setRoleid | 角色id | String | 角色id | 否 | 111 |
| setRolename | 角色名 | String | 角色名 | 否 | 哈哈 |
| setRolelevel | 角色等级 | String | 角色等级 | 否 | 9999 |
| SetServerid | 服务器id | String | 服务器名 | 否 | 哈哈 |

1. setBillNo:游戏订单号（商户订单号(12-32位数字、字母),不要使用特殊字符）
2. setExtraInfo：如果没有透传参数，直接传订单号
3. setAmount: 传整数，不要带小数点；
4. setUid: 传“”就行
5. 接口描述

回调接口只会返回支付界面关闭状态close

1. 退出接口（必接）
2. 示例代码

|  |
| --- |
| KLSDK.*exit*(**this**, **new** ExitListener() {  @Override  **Public void** fail(String msg) {  // **TODO** Auto-generated method stub  }  @Override  **Public void** ExitSuccess(String msg) {  // **TODO** Auto-generated method stub  System.*exit*(0);  }  }); |

1. 参数说明

activity 当前Activity

ExitListener退出回调

Fail方法体取消的回调；

ExitSuccess方法体确定退出的回调