猎宝 SDK 对接说明文档

SDK 版本号：5.1.6(4)

修改记录

修订人	版本号	备注	修订时间
韦家郑	5.1.0	增加功能（切换帐号、退出、注销帐号、数据统计、升级）	2016.01.01
韦家郑	5.1.0(3)	修改数据统计接口中的 ServerId 为 String 类型	2016.02.20
王东方	5.1.4 (4)	1、增加登录/支付取消回调 2、废除切换帐号回调 3、修改自动登录逻辑	2016.06.01
韦家郑	5.1.5(4)	1、在登录之后增加贴面广告 2、优化浮点显示	2016.07.22
韦家郑	5.1.6(4)	1、增加微信支付方式（不需要重新接入，直接替换资源即可）	2016.09.13

目录

1、概述

该文档主要对 SDK 接口进行描述，方便游戏开发者快速接入，详情请参照 demo 接入。

1.1、注意事项（重要）

- gameId、appId、appKey、agent 需由猎宝平台统一申请，确保唯一
- 所有参数统一配置在 assets 目录下的 lb_config.properties 文件中
- 包名后缀需要添加.lb
- 游戏强更时，游戏内无需提供升级
- 使用 SDK 时，需要在 AndroidManifest.xml 中配置相应的权限和属性
- 游戏接入 SDK 后的游戏母包不能做加固处理，猎宝需要重新签名/改包名
- 游戏必须继承 LBApplication
- 调用 SDK 所有接口，必须在 UI 线程中进行

2、接入配置

猎宝 SDK 主要包括 demo、assets、libs、res、AndroidManifest.xml、SDK 接入文档等几部分。assets 包含 SDK 接入所需配置文件，libs 包含接入 SDK 接入所依赖的 jar 文件，res 是所需要的资源文件，AndroidManifest.xml 中是需要添加的权限和属性。

2.1、继承 LBApplication

若游戏没有自定义 application，则在 AndroidManifest.xml 中配置 application 节点，如下：

```
<applicationandroid:name="com.lb.sdk.LBApplication"></application>
```

若游戏有自定义 application，则需继承 LBApplication，并在 AndroidManifest.xml 中配置 application 节点，如下：

```
<applicationandroid:name="xxx.xxx.xxx.XXApplication"></application>
```

2.2、配置参数

在 assets 目录下的 lb_config.properties 中配置，参数由猎宝运营人员提供

```
appID=322                    //应用 ID
gameID=336                   //游戏 ID
agent=default                //渠道号
```

2.3、配置 AndroidManifest

添加权限

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission  android:name="android.permission.CALL_PHONE" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.SYSTEM_OVERLAY_WINDOW" />
<uses-permission android:name="cn.swiftpass.wxpay.permission.MMOAUTH_CALLBACK" />
<uses-permission android:name="cn.swiftpass.wxpay.permission.MM_MESSAGE" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.FLASHLIGHT" />
<uses-feature android:name="android.hardware.camera" />
<uses-feature android:name="android.hardware.camera.autofocus" />
<uses-permission android:name="xvtian.gai.receiver" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.SYSTEM_OVERLAY_WINDOW" />
```

添加必须声明的 Activity 和 Service

```
 <activity android:name="com.liebao.sdk.ui.LoginActivity"
    android:theme="@android:style/Theme.Translucent.NoTitleBar.Fullscreen"
    android:launchMode="singleTask"
    android:screenOrientation="behind"
    android:configChanges="screenSize|keyboardHidden|orientation"/>
 <activity android:name="com.liebao.sdk.ui.PayActivity"
    android:theme="@android:style/Theme.Translucent.NoTitleBar.Fullscreen"
    android:screenOrientation="behind"
    android:launchMode="singleTask"
    android:configChanges="screenSize|keyboardHidden|orientation"/>
 <activity android:name="com.liebao.sdk.ui.NotificationActivity"
    android:theme="@android:style/Theme.Translucent.NoTitleBar.Fullscreen"
    android:screenOrientation="behind"
    android:launchMode="singleTask"
    android:configChanges="screenSize|orientation|keyboardHidden"/>
```

```
<activity android:name="com.liebao.sdk.ui.FloatWebActivity"
  android:theme="@style/lb_Translucent_NoTitle_WebView"
  android:screenOrientation="behind"
  android:launchMode="singleTask"
  android:configChanges="screenSize|orientation|keyboardHidden"/>
<activity android:name="com.alipay.sdk.auth.AuthActivity"
  android:configChanges="orientation|keyboardHidden|navigation"
  android:exported="false"
  android:screenOrientation="behind" />
<activity android:name="com.alipay.sdk.app.H5PayActivity"
  android:configChanges="orientation|keyboardHidden|navigation"
  android:exported="false"
  android:screenOrientation="behind"
  android:windowSoftInputMode="adjustResize|stateHidden" />
<activity  android:theme="@android:style/Theme.Translucent.NoTitleBar"
        android:name="com.switfpass.pay.activity.QQWapPayWebView"
        android:screenOrientation="portrait"/>
<service android:name="com.liebao.def.sdk.LBAppService"
  android:exported="false" />
<service android:name="com.liebao.def.sdk.util.DownloadService"
  android:exported="false" />
<receiver android:name="com.liebao.def.sdk.code.utils.DownloadReceiver">
  <intent-filter>
    <action android:name="android.intent.action.DOWNLOAD_COMPLETE"  />
  </intent-filter>
</receiver>
<meta-data android:name="DEFAULT_LB_LOGIN_PLUGIN"
  android:value="com.liebao.def.sdk.DefaultLBUser" />
<meta-data android:name="DEFAULT_LB_VERSION_PLUGIN"
  android:value="com.liebao.def.sdk.DefaultLBVersion" />
<meta-data android:name="DEFAULT_LB_PAY_PLUGIN"
  android:value="com.liebao.def.sdk.DefaultLBPay" />
<meta-data android:name="DEFAULT_LB_DATA_PLUGIN"
  android:value="com.liebao.def.sdk.DefaultLBData" />
```

屏幕适配

```
<supports-screens android:smallScreens="true" android:normalScreens="true"
  android:largeScreens="true" android:xlargeScreens="true"
  android:anyDensity="true" tools:ignore="ManifestOrder"/>
```

2.4、引入资源文件

将目录 libs 、res、assets 中的文件，引入或拷贝到工程中

3、接口描述

3.1、添加 sdk 生命周期与 ActivityResult 回调

3.1.1、添加生命周期

sdk 生命周期需要与游戏的生命周期绑定在一起，所以需要在 activity 的个各个生命周期中调用 sdk 生命周期。如下：

```
@Override
protected void onCreate() {
    LBSDK.getInstance().onCreate();
    super.onCreate();
}

@Override
protected void onResume() {
    LBSDK.getInstance().onResume();
    super.onResume();
}

@Override
protected void onRestart() {
    LBSDK.getInstance().onRestart();
    super.onRestart();
}

@Override
protected void onStart() {
    LBSDK.getInstance().onStart();
    super.onStart();
}

@Override
protected void onPause() {
    LBSDK.getInstance().onPause();
    super.onPause();
}

@Override
protected void onStop() {
    LBSDK.getInstance().onStop();
    super.onStop();
}

@Override
protected void onDestroy() {
    LBSDK.getInstance().onDestroy();
    super.onDestroy();
}
```

3.1.2、重写 onActivityResult、onConfigurationChanged、onNewIntent、onBackPressed 函数

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
   LBSDK.getInstance().onActivityResult(requestCode, resultCode, data);
   super.onActivityResult(requestCode, resultCode, data);
}

@Override
public void onConfigurationChanged(Configuration newConfig) {
   LBSDK.getInstance().onConfigurationChanged(newConfig);
   super.onConfigurationChanged(newConfig);
}

@Override
protected void onNewIntent(Intent intent) {
   LBSDK.getInstance().onNewIntent(intent);
   super.onNewIntent(intent);
}

@Override
public void onBackPressed() {
   LBSDK.getInstance().onBackPressed();
}
```

3.2、初始化平台

首先，需要在程序开始的地方初始化平台，主要通过调用 init 方法，并设置 context 对象。在 init 方法调用之后，需调用 checkVersion 方法检测升级。

3.2.1、方法定义

```
void init(Activity activity);
void checkVersion();
```

3.2.2、参数说明

参数	说明
activity	Activity 对象

3.2.3、代码示例

```
// 初始化猎宝 SDK
LBSDK.getInstance().init(this);
// 升级检测
LBVersion.getInstance().checkVersion();
```

3.3、设置监听 SDK 各事件回调

对登录、支付、注销、退出回调的处理

3.3.1、方法定义

```
void setSDKListener(ILBSDKListener listener);
```

3.3.2、接口参数说明

参数	说明
listener	各回调事件监听器

3.3.3、登录回调 LoginResult 参数说明

参数	类型	说明
code	int	值为 1：登录成功 值为 0：登录失败 值为-1：登录取消
msg	String	登录反馈提示信息
username	String	用户名
logintime	String	登录时间
sign	String	签名字符串，用于验证签名比对，规则： MD5("username=t315688&appkey=91bac46a9b70bd2db563cc483d443ba3&logintime=1395634100")

3.3.4、其他回调 Response 参数说明

参数	类型	说明
code	int	值为 1：成功 值为 0：失败 值为-1：取消
msg	String	反馈提示信息

3.3.5、代码示例

```
LBSDK.getInstance().setSDKListener(new ILBSDKListener() {
    @Override
    public void onResult(Response res) {
            // 普通回调，一般信息反馈回调，除非特殊情况，无需处理
    }
    @Override
    public void onLoginResult(LoginResult logRes) {
            if (logRes.getCode() == Constants.RESULT_CODE_SUCCESS){
                    // 登录成功回调后需验证签名
                    String appKey = "926db50e6808210e6ce9e3bc19de019c";
                    // 验签规则见登录部分
                    String mySign = Md5Util.md5("username=" +
                            logRes.getUsername()+ "&appkey=" + appKey +
                            "&logintime=" + logRes.getLogintime());
                    if (logRes.getSign().equals(mySign)){
                            // 验证签名成功，需获取参数进行服务器端再次验证
                    }else{
                            // 验证签名失败
                    }
            }else if(logRes.getCode() == Constants.RESULT_CODE_FAIL){
                    // 登录失败回调
            }else if(logRes.getCode() == Constants.RESULT_CODE_CANCEL){
                    // 登录取消回调
            }
    }
    @Override
    public void onPayResult(Response res) {
            if (res.getCode() == Constants.RESULT_CODE_SUCCESS){
                    // 支付成功回调
            }else if(res.getCode() == Constants.RESULT_CODE_FAIL){
                    // 支付失败回调
            }else if(res.getCode() == Constants.RESULT_CODE_CANCEL){
                    // 支付取消回调
            }
    }
    @Override
    public void onLogout(Response res) {
            if (res.getCode() == Constants.RESULT_CODE_SUCCESS){
                    // 注销成功回调
            }else{
                    // 注销失败回调
            }
    }
    @Override
    public void onExit(Response res) {
            if (res.getCode() == Constants.RESULT_CODE_SUCCESS){
                    // 退出成功回调
            }else{
                    // 退出取消回调
            }
    }
});
```

3.4、登录

此方法必须在 UI 线程中调用

3.4.1、方法定义

```
void login();
```

3.4.2、代码示例

```
LBUser.getInstance().login();
```

3.4.3、签名验证

在登录成功回调后，客户端需进行签名比对，比对通过后需获取参数通过服务器端再次验证，增加安全性。代码示例及验签规则详见目录3.3.3 / 3.3.5

3.5、充值

此方法必须在 UI 线程中调用

3.5.1、方法定义

```
void pay(PayParams params);
```

3.5.2、PayParams 参数说明

注：以下参数均为必填参数，不可为空

参数	类型	说明
roleId	String	角色编号
roleName	String	角色名称
roleLevel	String	角色等级，请传数字
serverId	String	区服编号
serverName	String	区服名称
productId	String	商品编号
productName	String	商品名称
productDesc	String	商品描述
price	String	充值金额（单位：元）
attach	String	透传字段，此处传游戏方订单号

3.5.3、代码示例

```
PayParams params = new PayParams();
params.setRoleId("rid_001");
params.setRoleName("花无缺");
params.setRoleLevel("99");
params.setServerId("sid_001");
params.setServerName("江湖一区");
params.setProductId("pid_101");
params.setProductName("飞羽扇");
params.setProductDesc("花无缺专属武器");
params.setPrice("1");
params.setAttach("cp_orderId_001");
LBPay.getInstance().pay(params);
```

3.5.4、充值回调

见目录 3.3.5onPayResult 回调方法

3.6、数据统计

3.6.1、方法定义

```
// 提交角色数据
void submitUserData(UserExtraDatauserData)
```

3.6.2、UserExtraData 参数说明

注：以下参数均为必填参数，不可为空

参数	类型	说明
dataType	int	1：登录 **（必接）** 2：注册 3：登出 4：创建角色 5：角色升级 需在以上五种情景下分别调用数据统计接口
roleId	String	角色编号
roleName	String	角色名称
roleLevel	String	角色等级，请传数字
serverId	String	区服编号
serverName	String	区服名称
moneyNum	String	角色剩余金币数量
uid	String	游戏方的帐号唯一标识，非角色 id
attach	String	冗余字段，如有其他信息可传入此字段，没有请传“0”

3.6.3、代码示例

```
UserExtraData userData = new UserExtraData();
userData.setDataType(1);
userData.setRoleID("rid_001");
userData.setRoleName("花无缺");
userData.setRoleLevel("99");
userData.setServerID("sid_001");
userData.setServerName("江湖一区");
userData.setMoneyNum(9099);
userData.setUid("hwq2355617592");
userData.setAttach("0");
LBData.getInstance().submitUserData(userData);
```

3.7、注销

由于猎宝SDK浮窗中有注销账号的操作，所以用户有可能在游戏过程中注销账号，因此需要游戏在onLogout 的回调中保存游戏数据，以及将游戏跳转到登录界面，且处于“未登录”状态。

3.7.1、方法定义

```
void logout();
```

3.7.2、代码示例

```
LBUser.getInstance().logout();
```

3.7.3、注销回调

见目录 3.3.5onLogout 回调方法

3.8、退出

由于猎宝 SDK 退出确认弹窗是后台可配置，所以在调用前需判断是否支持退出，猎宝提供 isSupport()方法进行判断，如不支持，需要执行游戏方自己的退出弹窗。

3.8.1、方法定义

```
void isSupport(“exit”);
void exit();
```

3.8.2、代码示例

```
if (LBUser.getInstance().isSupport("exit")){
   LBUser.getInstance().exit();
}else{
   //游戏方自己定义的退出确认弹窗
}
```

3.8.3、退出回调

见目录 3.3.5onExit 回调方法

4、服务器端 — 登录验证接口

游戏服务端通过移动端登陆获取的登陆信息调用此接口对登陆信息进行校验，校验成功后即可让玩家登陆。

4.1、描述

- 验证地址：http://sdk.51508.com/sdk/login/information.action
- 提交方式：POST

4.2、请求参数说明

参数	类型	说明
gameid	String	游戏编号
username	String	猎宝登录帐号
logintime	String	登录时间
sign	String	签名字符串，用于验证签名比对，规则如下： MD5("gameid="+gameid+"&username="+username+"&logintime="+logintime+"&appkey="+appkey) 签名算法为 MD5，统一使用 32 位(小写)UTF-8 加密算法。 规则为 key=value 的形式，参数之间以“&”符号相连，组成一串字符串。目前共 4 个签名字段。 签名使用的 appkey 由猎宝提供。

4.3、返回值

参数	类型	说明
code	String	200：操作成功 501：参数有误 502：appkey 不存在 503：签名已失效 504：签名错误 99999：系统错误
msg	String	描述信息
status	boolean	true：成功 false：失败

返回值以JSON格式进行返回，如下：

```
{"code":"200","msg":"操作成功","status":true}
```

5、服务器端—充值回调接口

游戏方需开发一个充值回调接口，用于接收猎宝支付结果通知，开发完毕后将该接口 URL 提供给猎宝运营人员进行配置。

5.1、描述

- 当用户在对应平台购买成功后，猎宝把支付结果以POST方式调用该接口。
- 游戏服务端通知接口在接收到通知消息后，若没有成功响应给猎宝，猎宝会进行 5 次每隔 2 秒回调游戏方，满 5 次若仍未成功，则每隔 15 分钟进行一次回调，3 小时内仍未成功，系统停止自动回调，该笔订单会被列入猎宝异常订单。
- 游戏服务端需要做参数签名验证，否则存在安全风险。
- 游戏服务端
  接收到请求后，需要对所有String类型的参数值做URLDecode处理，URLDecode编码统一为UTF-8。

5.2、请求参数说明

参数	类型	说明
orderid	String	猎宝订单编号
username	String	猎宝登录帐号
gameid	int	游戏编号
roleid	String	游戏角色编号
serverid	String	游戏区服编号
paytype	String	支付类型
amount	int	充值金额（单位：元）
paytime	int	充值时间
attach	String	透传字段
sign	String	签名字符串，用于验证签名比对，规则如下： MD5("orderid="+orderid+"&username="+username+"&gameid="+gameid+"&roleid="+roleid+"&serverid="+serverid+"&paytype="+paytype+"&amount="+amount+"&paytime="+paytime+"&attach="+attach+"&appkey="+appkey) 签名算法为 MD5，统一使用 32 位(小写)UTF-8 加密算法。 规则为 key=value 的形式，参数之间以“&”符号相连，组成一串字符串。目前共 10 个签名字段。 签名使用的 appkey 由猎宝提供。 签名后与参数中的 sign 进行对比，如果相同则签名通过，否则失败。

5.3、返回值

参数	类型	说明
success	String	处理成功
errorSign	String	验签失败
error	String	其他错误

注：返回值以字符串形式进行返回，**返回值不得为空**