虎牙 Berry SDK

Android 版本接口文档

概述

虎牙 Berry SDK 主要提供一套 Android 游戏直播技术方案及接口，接入方只需调用少数几个接口就可以实现复杂的游戏直播功能，并拥有很高的可重用性。另外，虎牙 Berry SDK 提供丰富的互动方式，如弹幕，礼物等，增强主播和粉丝之间的联系和互动。

1.1 SDK 接入流程

1.2 支持平台

Android 5.0 及以上（5.0 以下请屏蔽直播入口）

1.3 支持开发环境

Android Studio 1.4 及以上

1.4 Android 版本配置要求

编译版本 compileSdkVersion 25 及以下

目标版本 targetSdkVersion 25 及以下（低于 Android 8.0，悬浮窗要求）

最小版本 minSdkVersion 21（Android 5.0）

1.5 如何接入

1 初始化（必须调用，只需 1 次，有回调）

初始化 Berry SDK，所有功能的基础。

调用时机：在应用一开始的地方（建议 Application.OnCreate 或游戏入口的 Activity.OnCreate）。

```
HuyaBerryConfig huyaBerryConfig = new HuyaBerryConfig.Builder()
        .appId("13486878")  // 后台申请
        .appKey("1f7ce3cc55184019")  // appKey 是跟 appId 匹配自动生成的
        .gameId(2168)          // gameId 是跟 appId 匹配自动生成的
        .debugMode(false)  // 一般传 false，是否用测试模式（需要有虎牙测试环境）
        .landscapeMode(true)       //横竖屏设置（默认为横屏）
        .isOpenBugly(true)         //是否开启 SDK 中的 bugly（默认开启）
        .build();
    HuyaBerry.instance().init(getApplication(), huyaBerryConfig);
```

2 处理 Android 录屏权限请求回调（必须调用，只需 1 次）

录屏权限的回调。

调用时机：在直播界面 Activity 的 onActivityResult 里。

```
HuyaBerry.instance().onActivityResult(requestCode , resultCode, data);
```

3 开启直播首页（必须调用，可多次，有回调）

开启直播首页。

调用时机：直播入口处（初始化之后）。

```
    StartLiveConfig startLiveConfig = new StartLiveConfig.Builder()
            .landscapeMode(true) //横竖屏设置（如果应用初始化已传入，这里可以不传）
            .build();
    HuyaBerry.instance().startLive(MyActivity.this, startLiveConfig);
```

4 反初始化（必须调用，可多次）

结束直播并完成虎牙 Berry SDK 的资源回收工作。

调用时机：在应用结束的地方调用或者需要结束直播的地方。

```
HuyaBerry.instance().uninit();
```

5 修改横竖屏配置（可选，可多次）

在直播中可修改横竖屏配置，会影响之后显示的界面。

调用时机：任意。

```
HuyaBerry.instance().changeLandscapeMode(false);
//横竖屏设置，true 为横屏，false 竖屏
```

6 设置玩家的唯一标识（可选，可多次）

游戏厂商传入的唯一标识，对应某区某服的某个特定角色，如果没有设置，将收不到首页开启，直播开始，直播结束的事件回调。

调用时机：在开启直播首页前调用。

```
HuyaBerry.instance().setGameAccountID("123456");        //唯一角色标识
```

7 设置 SDK 事件监听（可选，只需 1 次）

设置 SDK 相关事件的回调，当前版本一共 5 个回调事件：初始化，首页开启，直播开始，直播结束，透传数据。回调的事件为<key,value>形式，定义及相关键值在

HuyaBerry.BerryEvent 类中。其中，首页开启，直播开始，直播结束这 3 个事件需要设置了玩家唯一标识才会回调。

调用时机：在初始化方法之前（可以收到所有的事件回调）。

```
HuyaBerry.instance().setBerryEventDelegate(new HuyaBerry.BerryEvent() {
    @Override
    public void onEventCallback(Map<String, String> dataMap) {
        if (dataMap == null) {
            return;
        }
        Log.i (TAG, dataMap.toString());            //事件的回调信息
     }
});
```

8 数据发送通道（可选，可多次）

游戏厂商传入的数据，可以实现向观众推荐道具、皮肤，显示玩家战绩等之类的合作功能。

调用时机：直播中。

```
HuyaBerry.BerryPlayerDataHelper dataHelper = new
HuyaBerry.BerryPlayerDataHelper();
dataHelper.roleName = "Tmac50";
dataHelper.serverID = "和平服-1";
dataHelper.playerLevel = "10";
dataHelper.customJson = "{\"action\":\"5 杀\",\"description\":\"超神\"}";
HuyaBerry.instance().sendPlayerData(dataHelper);
```

1.6 特殊接口

以下为特殊接口，游戏厂商自定义界面需要参考的相关接口

1 自定义界面开播接口（可选，只需 1 次）

游戏厂商直接调用 SDK 的开播界面，前提：未开播

```
HuyaBerry.instance().customUIStartLive(MyActivity.this, new CustomUICallback() {
    @Override
    public void onResultCallback(int status, BaseCallback baseCallback) {
        if (status == BaseCallback.SUCCESS) {
            mTvCustomUICallback.setText("调用成功");
        } else {
            mTvCustomUICallback.setText("调用失败");
        }
    }

    @Override
    public void onResultListCallback(int status, List list) {}
});
```

2 自定义界面获取主播信息接口（可选，可多次）

游戏厂商直接调用 SDK 获取主播信息，前提：已登录

```
HuyaBerry.instance().customUIGetAuthorInfo(this, new
CustomUICallback<AuthorInfo>() {
    @Override
    public void onResultCallback(int status, AuthorInfo authorInfo) {
        if (status == BaseCallback.SUCCESS) {
            mTvCustomUICallback.setText(authorInfo.toString());
        } else {
            mTvCustomUICallback.setText("调用失败");
        }
    }

    @Override
    public void onResultListCallback(int status, List<AuthorInfo> authorInfos) {}
});
```

3 自定义界面修改昵称接口（可选，可多次）

游戏厂商直接调用 SDK 的修改昵称界面，前提：已登录

```
HuyaBerry.instance().customUIModifyNickname(this, new CustomUICallback() {
    @Override
    public void onResultCallback(int status, BaseCallback baseCallback) {
        if (status == BaseCallback.SUCCESS) {
            mTvCustomUICallback.setText("调用成功");
        } else {
            mTvCustomUICallback.setText("调用失败");
        }
    }

    @Override
    public void onResultListCallback(int status, List list) {}
});
```

4 自定义界面修改直播标题接口（可选，可多次）

游戏厂商直接调用 SDK 修改直播标题，前提：已登录

```
HuyaBerry.instance().customUIModifyTitle(this, new CustomUICallback<ErrorInfo>()
{
    @Override
    public void onResultCallback(int status, ErrorInfo errorInfo) {
```

```
            if (status == BaseCallback.SUCCESS) {
                mTvCustomUICallback.setText("调用成功");
            } else {
                if (errorInfo == null) {
                    mTvCustomUICallback.setText("调用失败");
                } else {
                    mTvCustomUICallback.setText(errorInfo.errorMsg);
                }
            }
        }

        @Override
        public void onResultListCallback(int status, List list) {}
    },”newTitle”);
```

5 自定义界面修改直播公告接口（可选，可多次）

游戏厂商直接调用 SDK 修改直播公告，前提：已登录

```
    HuyaBerry.instance().customUIModifyAnnouncement(this, new
    CustomUICallback<ErrorInfo>() {
        @Override
        public void onResultCallback(int status, ErrorInfo errorInfo) {
            if (status == BaseCallback.SUCCESS) {
                mTvCustomUICallback.setText("调用成功");
            } else {
                if (errorInfo == null) {
                    mTvCustomUICallback.setText("调用失败");
                } else {
                    mTvCustomUICallback.setText(errorInfo.errorMsg);
                }
            }
        }

        @Override
        public void onResultListCallback(int status, List list) {}
    }, ”newAnnouncement”);
```

6 自定义界面登录接口（可选，可多次）

游戏厂商直接调用 SDK 的登录界面，前提：未登录

```
HuyaBerry.instance().customUILogin(this, new CustomUICallback() {
    @Override
    public void onResultCallback(int status, BaseCallback baseCallback) {
        if (status == BaseCallback.SUCCESS) {
            mTvCustomUICallback.setText("调用成功");
        } else {
            mTvCustomUICallback.setText("调用失败");
        }
    }

    @Override
    public void onResultListCallback(int status, List list) {}
});
```

7 自定义界面登出接口（可选，可多次）

游戏厂商直接调用 SDK 修改直播标题，前提：已登录

```
HuyaBerry.instance().customUILogout(this, new CustomUICallback() {
    @Override
    public void onResultCallback(int status, BaseCallback baseCallback) {
        if (status == BaseCallback.SUCCESS) {
            mTvCustomUICallback.setText("调用成功");
        } else {
            mTvCustomUICallback.setText("调用失败");
        }
    }

    @Override
    public void onResultListCallback(int status, List list) {}
});
```

8 自定义界面获取清晰度接口（可选，可多次）

游戏厂商直接调用 SDK 获取清晰度，前提：已登录

```
HuyaBerry.instance().customUIGetResolution(this, new
CustomUICallback<OptionalResolution>() {
    @Override
    public void onResultCallback(int status, OptionalResolution optionalResolution) {}

    @Override
    public void onResultListCallback(int status, List<OptionalResolution>
optionalResolutions) {
        if (status == BaseCallback.SUCCESS) {
            StringBuffer sb = new StringBuffer();
            for (OptionalResolution optionalResolution : optionalResolutions) {
                sb.append(optionalResolution.toString()).append("  ,  ");
            }
            mTvCustomUICallback.setText(sb.toString());
        } else {
            mTvCustomUICallback.setText("调用失败");
        }
    }
});
```

9 自定义界面设置清晰度接口（可选，可多次）

游戏厂商直接调用 SDK 设置清晰度，前提：已登录

```
HuyaBerry.instance().customUISetResolution(this, new CustomUICallback() {
    @Override
    public void onResultCallback(int status, BaseCallback baseCallback) {
        if (status == BaseCallback.SUCCESS) {
            mTvCustomUICallback.setText("调用成功");
        } else {
            mTvCustomUICallback.setText("调用失败");
        }
    }

    @Override
    public void onResultListCallback(int status, List list) {
    }
}, resolution);
```

具体的接口调用参见 demo。

接口说明

HuyaBerry 类

2.1.1 init

字段	类型	说明
传入参数		
application	Application	接入方 app 的实例，主要用于系统上下文环境
huyaBerryConfig	HuyaBerryConfig	初始化参数，详情见下
返回值		
无		

HuyaBerryConfig

字段	类型	说明
appId	String	申请的 appId
appKey	String	与 appId 对应的 appKey
gameId	int	游戏 id，跟 appId 对应分配的
debugMode	boolean	是否用测试模式，如果是，则接入的是虎牙的测试环境
landscapeMode	boolean	是否用横屏模式，如果不设置默认是横屏
isOpenBugly	boolean	是否开启 SDK 中的 bugly，如果游戏中已有，传入 false，避免 bugly 实例的冲突

2.1.2 startLive

字段	类型	说明
传入参数		
activity	Activity	接入方当前 Activity
startLiveConfig	StartLiveConfig	开播参数，详情见下
返回值		
无		

StartLiveConfig

isHasLiveList	boolean	首页是否包含列表
landscapeMode	boolean	是否用横屏模式，如果不设置

		默认是横屏（会覆盖初始化的设置）

2.1.3 onActivityResult

录屏权限回调，在接入方的当前 Activity 的 onActivityResult 回调加入即可，参数都用 Activity 的回调参数。

字段	类型	说明
传入参数		
requestCode	int	系统回调
resultCode	int	系统回调
data	Intent	系统回调
返回值		
无		

2.1.4 uninit

反初始化，建议在 Application.OnDestroy 里面调用

字段	类型	说明
传入参数		
无		
返回值		
无		

2.1.5 changeLandscapeMode

字段	类型	说明
传入参数		
landscapeMode	boolean	是否用横屏模式，如果不设置默认是横屏（会覆盖初始化的设置）
返回值		
无		

2.1.6 setGameAccountID

字段	类型	说明
传入参数		
gameAccountID	String	玩家唯一标识
返回值		

无		

2.1.7 setBerryEventDelegate

字段	类型	说明
传入参数		
eventDelegate	BerryEvent	事件回调 void onEventCallback(Map<String, String> dataMap)
返回值		
无		

回调的事件 key 值在 BerryEvent 的定义值

暂时回调事件有 5 个：

1) 初始化事件，返回包括：事件类型、结果码、信息。

   示例：{eventType=init, resultCode=0, msg=}

2) 首页启动事件，返回包括：事件类型、虎牙账号唯一标识<未登录为 0>、游戏账号唯一标识、SDK 启动时间、房间号<未认证主播房间号为 0>。

   示例：{eventType=startUp, huyaUid=1491839133, gameAccountID=test, startUpTime=1534922566744, roomId=0}

3) 主播开播事件，返回包括：事件类型、虎牙账号唯一标识、游戏账号唯一标识、开播时间、房间号。

   示例：{eventType=startLive, huyaUid=1491839133, gameAccountID=test, startLiveTime=1534922739599, roomId=15341663}

4) 主播停播事件，返回包括：事件类型、虎牙账号唯一标识、游戏账号唯一标识、结束时间、本次直播时长、房间号。

   示例：{eventType=endLive, huyaUid=1491839133, gameAccountID=test, endLiveTime=1534922893694, duration=00:02:35, roomId=15341663}

5) 透传数据事件，返回包括：事件类型、结果码、信息。

   示例：{eventType=sendPlayerData, resultCode=0, msg=}

2.1.8 sendPlayerData

字段	类型	说明
传入参数		
dataHelper	BerryPlayerDataHelper	透传的数据，包括角色名，服务器 id，玩家等级，自定义

		json
返回值		
无		

透传数据到观众端

2.2.1 customUIStartLive

字段	类型	说明
传入参数		
activity	Activity	接入方当前 Activity
callback	CustomUICallback	调用的回调结果
返回值		
无		

2.2.2 customUIGetAuthorInfo

字段	类型	说明
传入参数		
activity	Activity	接入方当前 Activity
callback	CustomUICallback	调用的回调结果
返回值		
authorInfo	AuthorInfo	返回的主播信息

2.2.3 customUIModifyNickname

字段	类型	说明
传入参数		
activity	Activity	接入方当前 Activity
callback	CustomUICallback	调用的回调结果
返回值		
无		

2.2.4 customUIModifyTitle

字段	类型	说明
传入参数		
activity	Activity	接入方当前 Activity
callback	CustomUICallback	调用的回调结果

newTitle	String	修改后的标题
返回值		
errorInfo	ErrorInfo	若失败，具体的错误信息

2.2.5 customUIModifyAnnouncement

字段	类型	说明
传入参数		
activity	Activity	接入方当前 Activity
callback	CustomUICallback	调用的回调结果
newAnnouncement	String	修改后的公告
返回值		
errorInfo	ErrorInfo	若失败，具体的错误信息

2.2.6 customUILogin

字段	类型	说明
传入参数		
activity	Activity	接入方当前 Activity
callback	CustomUICallback	调用的回调结果
返回值		
无		

2.2.7 customUILogout

字段	类型	说明
传入参数		
activity	Activity	接入方当前 Activity
callback	CustomUICallback	调用的回调结果
返回值		
无		

2.2.8 customUIGetResolution

字段	类型	说明
传入参数		
activity	Activity	接入方当前 Activity
callback	CustomUICallback	调用的回调结果

返回值		
optionalResolution	OptionalResolution	可选的清晰度

2.2.9 customUISetResolution

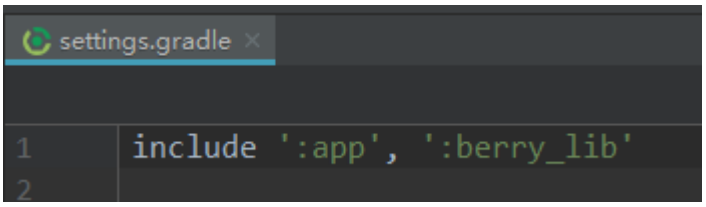
字段	类型	说明
传入参数		
activity	Activity	接入方当前 Activity
callback	CustomUICallback	调用的回调结果
返回值		
selectedResolution	Int	可选清晰度返回的清晰度数值

具体接入步骤（分版本）

接入之前最好先参照 demo 的依赖方式

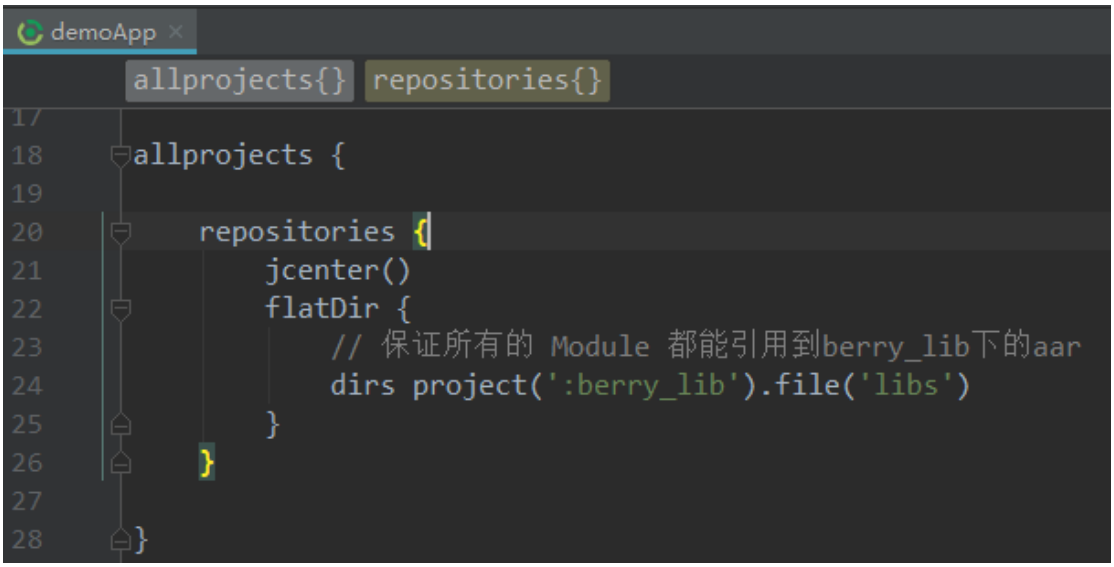
3.1 Android Studio 版本接入

1. 将 berry_lib 文件复制到项目中（项目根目录）
2. 在 ./setting.gradle 中增加':berry_lib'，引入 berry 库



3. 在 ./build.gradle 中 allprojects{ repositories{ 后插入以下代码：

```
flatDir {
    dirs project(':berry_lib').file('libs')
}
```



4. 在 ./app/build.gradle 中 android{ 后插入以下代码：

```
sourceSets {
    main.jniLibs.srcDirs = ['../berry_lib/libs']
}
```

在文件引入：

```
dependencies {
    compile project(':berry_lib')
}
```

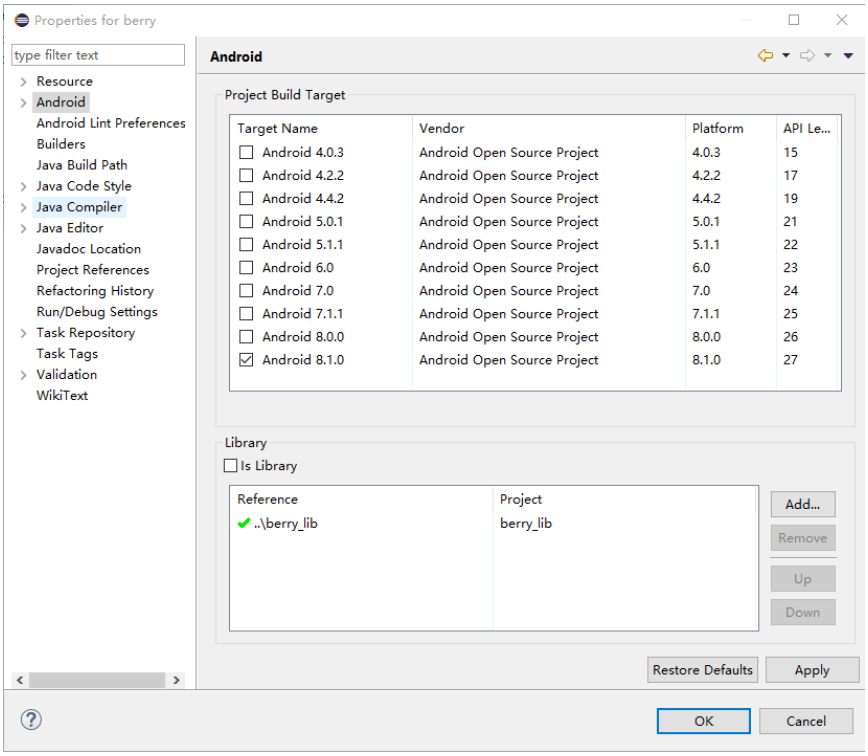
5. 在主项目的 string.xml 中加入 hyberry_webview_scheme 键值，填入值为 huyagame + appid（虎牙分配的），如下图：（gameid 假设为 huya_app_123）

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hyberry_webview_scheme">huyagamehuya_app_123</string>
</resources>
```

6. 在引入的 module（如 app）编写代码，检查引入是否成功，引入 HuyaBerry 类没有报错即引入库成功

3.2 Unity/Eclipse 版本接入

1. 将 berry_lib 文件复制到项目中（Eclipse 与主项目同目录下，Unity 项目是 Plugins/Android 目录下）
2. 在 Eclipse 中 import berry_lib 的库到工程中
3. 引入 berry_lib 库，Eclipse 引入界面如下：



4. 在主项目的 string.xml 中加入 hyberry_webview_scheme 键值，填入值为 huyagame + appid（虎牙分配的），如下图：（gameid 假设为 huya_app_123）

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3     <string name="hyberry_webview_scheme">huyagamehuya_app_123</string>
4 </resources>
5
```

5. 编写代码，检查引入是否成功，引入 HuyaBerry 类没有报错即引入库成功

常见问题

4.1 接入相关

1.找不到 HuyaBerry 类：检查 SDK 版本和接入流程，是否跳过了某个流程

4.2 崩溃相关

1.如果崩溃中有 berry 或者 hy 相关的信息，请上报给相关的对接人员

4.3 回调相关

1.收不到初始化回调：回调只能在设置之后才生效，即设置之后才会接到后续事件的回调，要接到初始化的回调，需要在 init 方法之前就设置回调

2.收不到首页启动，主播开播，主播停播事件，是否未设置回调或者设置游戏唯一标识（setBerryEventDelegate 和 setGameAccountID）

4.4 全面屏适配

打出的包，有些机型不能显示充满整个屏幕，需添加全面屏适配代码如下：

```
<meta-data
    android:name="android.max_aspect"
    android:value="2.2"/>

<meta-data
    android:name="android.vendor.full_screen"
    android:value="true" />

<meta-data
    android:name="android.notch_support"
    android:value="true" />

<meta-data
    android:name="notch.config"
    android:value="portrait|landscape" />
```