Android 点播回放 UI SDK

- 源码地址
- Platform: API 21+
- CPU: ARM-v7a, ARM64-v8a
- IDE: **Android Studio** Recommend
- Change Log
- 点播 3.X UI SDK 文档、回放 3.X UI SDK

1. 简介

带 UI 的点播 SDK 基于点播 Core SDK，提供标准的 UI 实现，方便用户快速集成投入使用。此 SDK 代码开源，开发者可以自己建立分支进行开发，也欢迎给我们提 issue。

2. 快速集成

2.1. 添加 maven 仓库

从 4.0 SDK 起，引入了新的 `nexus.baijiayun.com` 仓库。

gradle 7.1 及以上

```
maven {
        allowInsecureProtocol true
```

```
        url 'http://nexus.baijiayun.com/nexus/content/groups/public/'
    }
maven { url 'https://git2.baijiashilian.com/open-android/maven/raw/master/' }
```

gradle 7.1 以下

```
maven {url 'http://nexus.baijiayun.com/nexus/content/groups/public/'}
maven { url 'https://git2.baijiashilian.com/open-android/maven/raw/master/' }
```

2.2. 版本号说明

版本号格式为 `大版本.中版本.小版本[-alpha(测试版本)/beta(预览版本)]` ：

- 测试版本和预览版本可能不稳定，请勿随意尝试。
- 小版本升级只改 BUG、UI 样式优化，不会影响功能。
- 中版本升级、修改功能，更新 UI 风格、布局，会新增 API、标记 API 即将废弃，但不会导致现有 API 不可用。
- 大版本任何变化都是有可能的。

首次集成建议选择最新正式版本(版本号中不带有 `alpha` 、 `beta` 字样)，**版本升级后请仔细阅读**

2.3. 添加依赖

最新版本请点击自取，Releases · Open Android / VideoplayerUI2.0Demo · GitLab

```
implementation'com.baijia.player:VideoplayerUI:4.2
```

3. 快速集成

3.1. 初始化 SDK

参考 点播 Core SDK 初始化

3.2. 调起点播播放页面

`VideoPlayActivity` 为标准的点播播放界面，该布局为全屏居中播放，用户可以参考实现任意组合下的点播播放。

`PBRoomUI` 为 UI SDK 入口工具类，提供一系列 static 函数方便用户调用。

在线播放

```
/**
    * 进入标准在线点播播放界面
    * @param context       context
    * @param videoId       点播vid
    * @param token         点播token
    * @param playerConfig  创建播放器的配置项，null则取默认配置项
    */
   public static void startPlayVideo(Context context, long videoId, String token, VideoPlayerConfig playerConfig) {
        Intent intent = new Intent(context, VideoPlayActivity.class);
        intent.putExtra(ConstantUtil.IS_OFFLINE, false);
```

```
        intent.putExtra(ConstantUtil.VIDEO_ID, videoId);
        intent.putExtra(ConstantUtil.TOKEN, token);
        if (playerConfig == null) {
            playerConfig = new VideoPlayerConfig();
        }
        intent.putExtra(ConstantUtil.VIDEO_PLAYER_CONFIG, playerConfig);
        context.startActivity(intent);
    }
```

点播三分屏

```
    /**
     * 进入标准在线点播三分屏播放界面
     * @param context       context
     * @param videoId       点播vid
     * @param token         点播token
     * @param playerConfig  创建播放器的配置项，null则取默认配置项
     */
    public static void startPlayVideoTriple(Context context, long videoId, String token, VideoPlayerConfig playerConfig) {
        Intent intent = new Intent(context, VideoPlayTripleActivity.class);
        intent.putExtra(ConstantUtil.IS_OFFLINE, false);
        intent.putExtra(ConstantUtil.VIDEO_ID, videoId);
        intent.putExtra(ConstantUtil.TOKEN, token);
        if (playerConfig == null) {
            playerConfig = new VideoPlayerConfig();
        }
```

```
intent.putExtra(ConstantUtil.VIDEO_PLAYER_CONFIG,
 playerConfig);
        context.startActivity(intent);
    }
```

离线播放

```
/**
     * 进入标准离线点播播放界面
     * @param context         context
     * @param downloadModel   下载model(包含离线视频的全部信息)
     * @param playerConfig    创建播放器的配置项，null则取默认配置项
     */
    public static void startPlayLocalVideo(Context context, DownloadModel downloadModel, VideoPlayerConfig playerConfig) {
        Intent intent = new Intent(context, VideoPlayActivity.class);
        intent.putExtra(ConstantUtil.IS_OFFLINE, true);
        intent.putExtra(ConstantUtil.VIDEO_DOWNLOAD_MODEL,
 downloadModel);
        if (playerConfig == null) {
            playerConfig = new VideoPlayerConfig();
        }
        intent.putExtra(ConstantUtil.VIDEO_PLAYER_CONFIG,
 playerConfig);
        context.startActivity(intent);
    }
```

3.3. 调起回放播放页面

标准回放

```
/**
     * 进入回放标准UI界面
     *
     * @param context                    Activity
     * @param roomId                     房间id
     * @param roomToken                    token
     * @param sessionId                    sessionId 长期课才需要填，非长期课默认传-1
     * @param onEnterPBRoomFailedListener 进房间错误监听，可为null
     */
    public static void enterPBRoom(Context context, String roomId, String roomToken, String sessionId,
                                   OnEnterPBRoomFailedListener onEnterPBRoomFailedListener)
```

标准回放 with videoConfig

```
/**
     * 进入回放标准UI界面
     *
     * @param context                    Activity
     * @param roomId                     房间id
     * @param roomToken                    token
     * @param sessionId                    sessionId 长期课才需要填，非长期课默认传-1
     * @param playerConfig                 创建播放器的配置项， null则取默认配置项
```

```
     * @param onEnterPBRoomFailedListener 进房间错误监听，可为null
     */
    public static void enterPBRoom(Context context, String roomId, String roomToken, String sessionId, VideoPlayerConfig playerConfig, OnEnterPBRoomFailedListener onEnterPBRoomFailedListener)
```

裁剪回放

```
    /**
     * 进入回放标准UI界面
     *
     * @param context                  Activity
     * @param roomId                   房间id
     * @param roomToken                token
     * @param sessionId                sessionId 长期课才需要填，非长期课默认传-1
     * @param version                  裁剪版本号 （不传主版本，传0原视频）
     * @param playerConfig             创建播放器的配置项， null则取默认配置项
     * @param onEnterPBRoomFailedListener 进房间错误监听，可为null
     */
    public static void enterPBRoom(Context context, String roomId, String roomToken, String sessionId, int version, VideoPlayerConfig playerConfig, OnEnterPBRoomFailedListener onEnterPBRoomFailedListener)
```

合并回放

```
/**
```

```
    * 合并回放，多个回放合并在一起
    * @param context
    * @param mixedId
    * @param mixedToken
    * @param onEnterPBRoomFailedListener
    */
   public static void enterPBRoom(Context context, String mixedId, String mixedToken, OnEnterPBRoomFailedListener onEnterPBRoomFailedListener)
```

合并回放 with videoConfig

```
   /**
    * 合并回放
    * @param context
    * @param mixedId
    * @param mixedToken
    * @param playerConfig
    * @param onEnterPBRoomFailedListener
    */
   public static void enterPBRoom(Context context, String mixedId, String mixedToken, VideoPlayerConfig playerConfig, OnEnterPBRoomFailedListener onEnterPBRoomFailedListener)
```

离线回放

```
   /**
    * 进入离线回放标准UI界面
    *
    * @param context     activity
    * @param videoModel  视频 DownloadModel
    * @param signalModel 信令 DownloadModel
```

```
    */
    public static void enterLocalPBRoom(Context context, DownloadModel videoModel, DownloadModel signalModel)
```

离线回放 with videoConfig

```
 /**
    * 进入离线回放标准UI界面
    * @param context        activity
    * @param videoModel     视频 DownloadModel
    * @param signalModel    信令 DownloadModel
    * @param playerConfig   创建播放器的配置项，null则取默认配置项
    */
    public static void enterLocalPBRoom(Context context, DownloadModel videoModel, DownloadModel signalModel, VideoPlayerConfig playerConfig)
```

3.4. VideoPlayerConfig

`VideoPlayerConfig` 提供一系列配置参数供外部控制播放器行为，如下

```
// 是否后台播放
    public boolean supportBackgroundAudio = false;
    // 是否循环播放
    public boolean supportLooping = false;
    // 是否记忆播放
    public boolean supportBreakPointPlay = true;
    // 用户名
    public String userName;
```

```
    // 用户ID
    public String userId;
    // 跑马灯
    public LPHorseLamp horseLamp;
     // 是否支持进度条拖动
    public boolean supportSeek = true;
    // 最大可拖拽观看时间
    public int maxWatchTime =
Integer.MAX_VALUE;
    // 是否支持倍速播放
    public boolean supportVideoRate = true;
    // 是否以视频为主 (true为以视频为主)
    public Boolean isVideoMain;
    // 是否默认横屏
    public Boolean isLandscape;
    // 是否允许切换横竖屏
    public boolean enableToggleScreen = true;
     // 点播评论 token
    public String loginToken;
    // 点播评论 用户avatar链接，null则显示本地默认头
像
    public String avatar;
    // 签到
    public List<SignModel> signModelList;

    /**
     * 点播合集 or 回放合集 数据源
     */
    public List<AlbumItemData>
albumItemDataList = new ArrayList<>();

    /**
     * 合集默认 index
     */
    public int defaultAlbumIndex;
```

其中 `userName` 和 `userId` 用户播放行为上报，建议设置。

3.5. 回放合集

外界通过 `VideoplayerConfig.setAlbumItemDataList()` 传入合集列表，`VideoplayerConfig.defaultAlbumIndex` 为默认播放合集的下标。

```
/**
 * 回放专辑：roomId + sessionId + version + token + prefaceUrl + name
 * 点播合集：videoId + token + prefaceUrl + name
 */
data class AlbumItemData(
    @SerializedName("room_id")
    val roomId: Long,
    @SerializedName("session_id")
    val sessionId: Long,
    val version: Int,
    @SerializedName("play_token")
    val token: String,
    @SerializedName("vid")
    val videoId: Long,
    // 封面
    @SerializedName("preface_url")
    val prefaceUrl: String?,
    // 标题
    val name: String,
    @SerializedName("is_effective_expired")
    val isExpired: Int
)
```

3.6. 点播签到

点播签到功能支持在指定时间点显示指定文案，并且回调签到事件。

外界通过 `VideoplayerConfig.setSignModelList()` 传入签到列表。

每一个 `SignModel` 代表一个签到打点，字段如下

```
public class SignModel implements Serializable {
    /**
     * 显示签到的时间点，单位秒
     */
    public int seconds;
    /**
     * 签到弹窗 logo，null 则为默认图标
     */
    public String logo;
    /**
     * 签到标题
     */
    public String title;
    /**
     * 签到内容
     */
    public String content;
    /**
     * 签到按钮文案
     */
    public String btnText;
}
```

设置点播签到按钮点击回调

```
CallbackManager.getInstance().setSignConfigListen
    Consumer<SignModel>() {
```

```
            @Override
            public void accept(SignModel signModel) {
                LPLogger.d(TAG, "sign:" + PBJsonUtils.toString(signModel));

                Toast.makeText(LauncherActivity.this, PBJsonUtils.toString(signModel), Toast.LENGTH_SHORT).show();
            }
        });
```

3.7. 点播分享

`CallbackManager.setShareListener(ShareListener shareListener)` 设置点播分享回调后，UI 界面上会显示分享按钮（默认不显示），点击回调 `onShareClicked(String videoId)` 。

```
public interface ShareListener {
    /**
     * 点击分享回调
     *
     * @param videoId 点播视频 id
     */
    void onShareClicked(String videoId);
}
```

3.8. 点播/回放试看

`VideoPlayerConfig.visitorModel` 设置试看信息，包括允许试看的时长和弹框文案。

```
public class VisitorModel extends CustomUserModel {
    /**
     * 试看时长，单位秒
     */
    public int seconds = Integer.MAX_VALUE;
}
```

`CustomUserModel` 字段如下

```
public class CustomUserModel implements Serializable {
    /**
     * logo url
     */
    public String logoUrl;
    /**
     * logo resource id，与 logoUrl 互斥
     */
    public int logoResId;
    /**
     * 弹窗提示标题
     */
    public String title;
    /**
     * 弹窗提示正文
     */
    public String content;
    /**
     * 确认按钮文案
     */
    public String positiveText;
    /**
     * 取消按钮文案
     */
```

```
25.     public String navigateText;
26. }
```

3.9. 点播/回放禁止拖拽

`VideoPlayerConfig.enableDragCOntroller = false`，默认值为 true，允许拖拽。

```
1.  /**
2.      * 是否允许拖拽
3.      */
4.     public boolean enableDragController = true;
```

禁止拖拽后，用户拖拽会弹框提示，设置 `VideoPlayerConfig.dragControllerModel` 自定义文案。`DragControllerModel` 继承自 `CustomUserModel`，详见 3.8 节。

4. 功能介绍

包结构

4.1. 自定义点播UI

component包下提供一系列标准播放器UI组件，用户可以在ComponentManager管理类中配置需要显示的组件。

```
1. /**
```

```
 * 默认组合的组件
 */
public void generateDefaultComponentList(){
    componentMap.clear();
    componentChangeListeners.clear();

addComponent(UIEventKey.KEY_LOADING_COMPON
 new LoadingComponent(context));

addComponent(UIEventKey.KEY_GESTURE_COMPON
 new GestureComponent(context));
    //controller 需在gesture布局上方，否则会有事件冲突

addComponent(UIEventKey.KEY_CONTROLLER_COM
 new ControllerComponent(context));

addComponent(UIEventKey.KEY_ERROR_COMPONEI
 new ErrorComponent(context));

addComponent(UIEventKey.KEY_MENU_COMPONEN
 new MenuComponent(context));
    if(BJYPlayerSDK.IS_DEVELOP_MODE){

addComponent(UIEventKey.KEY_VIDEO_INFO_COMF
 new MediaPlayerDebugInfoComponent(context));
    }
  }
```

4.2. 自定义组件

1）ComponentContainer

ComponentContainer extends FrameLayout，持有ComponentManager初始化各个播放器组件，实现手势监听，

并且控制自定义事件和触摸事件的分发。

2）BaseVideoView

BaseVideoView extends FrameLayout，持有IBJYVideoPlayer引用获取播放器回调，持有ComponentContainer通知各个component组件更新状态，内部实现网络监听逻辑。

3）BJYVideoView

BJYVideoView extends BaseVideoView,初始化BJYPlayerView和各个component,监听播放器回调通知各个component。