1功能概述

PPT功能是将讲师端当前正在操作的PPT或者文档显示为一个小的悬浮窗的功能。

由于PPT是嵌入在悬浮窗中显示的，因此该模块将悬浮窗和PPT分开为两个接口，一个接口代表的是嵌入了PPT的悬浮窗，一个接口是PPT。

2 使用演示

```
//初始化
IPLVLCFloatingPPTLayout floatingPPTLayout=findViewById(R.id.plvlc_ppt_floating_ppt_layout);
//设置悬浮窗点击监听器
floatingPPTLayout.setOnFloatingViewClickListener();
//设置关闭悬浮窗的点击监听器
floatingPPTLayout.setOnClickCloseListener();
//设置直播PPT事件监听器
floatingPPTLayout.getPPTView().initLivePPT();
//设置回放PPT事件监听
floatingPPTLayout.getPPTView().initPlaybackPPT();
//销毁
floatingPPTLayout.destroy();
```

详细的调用代码以及和其他模块的通信，请参考PLVLCCloudClassActivity类中的代码调用。

3 接口介绍

3.1 IPLVLCFloatingPPTLayout

`IPLVLCFloatingPPTLayout` 是云课堂场景下定义的PPT悬浮窗布局，定义了：

1. 外部直接调用的方法
2. 需要外部响应的事件监听器

```
public interface IPLVLCFloatingPPTLayout {

    // <editor-fold defaultstate="collapsed" desc="1. 外部直接调用的方法">

    /**
     * 设置服务端的PPT开关
     *
     * @param enable true表示打开PPT，false表示关闭PPT
     */
    void setServerEnablePPT(boolean enable);

    /**
     * 显示
     */
    void show();

    /**
     * 隐藏
     */
    void hide();

    /**
     * PPT是否在悬浮窗中
     */
    boolean isPPTInFloatingLayout();

    /**
     * 获取PPTView
     */
    IPLVLCPPTView getPPTView();

    /**
     * 设置悬浮窗点击监听器
     *
     * @param li listener
     */
    void setOnFloatingViewClickListener(View.OnClickListener li);

    /**
     * 设置显示关闭悬浮窗的监听器
     *
     * @param onClickCloseListener listener
     */
    void setOnClickCloseListener(IPLVOnClickCloseFloatingView onClickCloseListener);

    /**
     * 获取切换View
     */
    PLVSwitchViewAnchorLayout getPPTSwitchView();

    /**
     * 销毁
     */
    void destroy();
    // </editor-fold>
```

```
    // <editor-fold defaultstate="collapsed" desc="2. 需要外部响应的事件监听器">

    interface IPLVOnClickCloseFloatingView {
        /**
         * 点击关闭
         */
        void onClickCloseFloatingView();
    }
    // </editor-fold>

}
```

3.2 IPLVLCPPTView

云课堂场景下，针对 PPT布局 进行封装的interface，定义了：

1. 外部直接调用的方法
2. 需要外部响应的事件监听器

```
public interface IPLVLCPPTView {

    // <editor-fold defaultstate="collapsed" desc="1. 外部直接调用的方法 - live部分，定义 直播PPT独有的方法">

    /**
     * 初始化直播PPT
     *
     * @param onPLVLCPPTViewListener listener
     */
    void initLivePPT(OnPLVLCLivePPTViewListener onPLVLCPPTViewListener);

    /**
     * 移除延迟时间
     */
    void removeDelayTime();

    /**
     * 恢复延迟时间
     */
    void recoverDelayTime();


// </editor-fold>

    // <editor-fold defaultstate="collapsed" desc="1. 外部直接调用的方法 - playback部分，定义回放PPT独有的方法">
    /**
     * 初始化回放PPT
     *
     * @param onPlaybackPPTViewListener listener
     */
    void initPlaybackPPT(OnPLVLCPlaybackPPTViewListener onPlaybackPPTViewListener);

    /**
     * 设置当前回放视频播放的位置
     *
     * @param position 播放位置
     */
    void setPlaybackCurrentPosition(int position);

    /**
     * 获取回放专用的PPTView，用于绑定到播放器内部
     *
     * @return 回放专用的PPTView的接口
     */
    IPolyvPPTView getPlaybackPPTViewToBindInPlayer();
// </editor-fold>

    // <editor-fold defaultstate="collapsed" desc="1. 外部直接调用的方法 - common部分，定义直播PPT和回放PPT通用的方法">

    /**
     * 发送消息到webView
     *
     * @param event   消息的事件
     * @param message 消息内容
```

```
     */
    void sendWebMessage(String event, String message);

    /**
     * 重新加载
     */
    void reLoad();

    /**
     * 销毁
     */
    void destroy();
    // </editor-fold>

    // <editor-fold defaultstate="collapsed" desc="2. 需要外部响应的事件监听器 - PPT直播事件监听器">

    /**
     * PPT直播事件监听器
     */
    interface OnPLVLCLivePPTViewListener {

        /**
         * 直播回调，切换横竖屏
         *
         * @param toLandscape true表示切换到横屏，false表示切换到竖屏
         */
        void onLiveChangeToLandscape(boolean toLandscape);

        /**
         * 直播回调，开始或暂停视频
         *
         * @param toStart true表示开始播放视频，false表示暂停播放视频
         */
        void onLiveStartOrPauseVideoView(boolean toStart);

        /**
         * 直播回调，重播视频
         */
        void onLiveRestartVideoView();

        /**
         * 直播回调，回到上一个Activity
         */
        void onLiveBackTopActivity();
    }
    // </editor-fold>

    // <editor-fold defaultstate="collapsed" desc="2. 需要外部响应的事件监听器 - PPT回放事件监听器">

    /**
     * PPT回放事件监听器
     */
    interface OnPLVLCPlaybackPPTViewListener {

        /**
         * 回放回调，切换PPTView的位置
```

```
         *
         * @param toMainScreen true表示切换到主屏幕，false表示切回到小窗
         */
        void onPlaybackSwitchPPTViewLocation(boolean toMainScreen);
    }
    // </editor-fold>

}
```

4 实现介绍

IPLVLCFloatingPPTLayout的实现类是：PLVLCFloatingPPTLayout。

IPLVLCPPTView的实现类是：PLVLCPPTView。

4.1 PLVLCFloatingPPTLayout

PLVLCFloatingPPTLayout是一个全屏大小的布局，由他的子View，PLVTouchFloatingView作为真正的悬浮窗控件响应悬浮窗的拖动事件。

4.1.2 初始化View

可以在初始化中设置悬浮窗的横屏和竖屏各自的初始化位置，也可以设置在横屏和竖屏下，悬浮窗的大小。

```
//初始化View
private void initView() {
    //...
    //设置初始化位置
    int screenWidth = Math.min(ScreenUtils.getScreenWidth(), ScreenUtils.getScreenHeight());
    int screenHeight = Math.max(ScreenUtils.getScreenWidth(), ScreenUtils.getScreenHeight());
    floatingView.setInitLocation(
            screenWidth - PLVScreenUtils.dip2px(DP_FLOATING_PPT_WIDTH_PORT),
            PLVScreenUtils.dip2px(DP_ORIGIN_MARGIN_TOP_PORTRAIT),
            screenHeight - PLVScreenUtils.dip2px(DP_FLOATING_PPT_WIDTH_LAND) -
PLVScreenUtils.dip2px(DP_ORIGIN_MARGIN_RIGHT_LANDSCAPE),
            PLVScreenUtils.dip2px(DP_ORIGIN_MARGIN_TOP_LANDSCAPE)
    );
}
```

4.1.2 初始化数据

悬浮窗内部使用了MVP模式，需要初始化Presenter。

```
//初始化数据
private void initData() {
    presenter = new PLVLiveFloatingPresenter();
    presenter.init(this);
}
```

而悬浮窗实现的View层接口是：

```
/**
 * 悬浮窗View
 */
interface IPLVLiveFloatingView {
    /**
     * 设置讲师信息
     *
     * @param nick   讲师昵称
     * @param picUrl 讲师图片Url
     */
    void updateTeacherInfo(String nick, String picUrl);
}
```

用于更新悬浮窗上讲师的信息。

4.2 PLVLCPPTView

PLVLCPPTView是PPT的实现。有如下几点需要注意：

1. 它是对SDK中的类：PolyvPPTWebView，的一层业务封装。
2. 它使用mvp模式来处理服务端数据的收发等逻辑。
3. 由于PPT既可以直播，也可以在回放中使用，因此它的接口分为live、playback、common3组，具体可以直接看接口注释。

4.2.1 初始化View

```
//初始View
private void initialView(Context context) {
    //...
    //设置占位图
    pptPlaceHolderView.setPlaceHolderImg(R.drawable.plvlc_ppt_placeholder);
    //设置占位图文本
    pptPlaceHolderView.setPlaceHolderText(getResources().getString(R.string.plv_ppt_no_document));
    //...
    //加载ppt的webView
    pptWebView.loadWeb();
}
```

4.2.2 初始化数据

PPT内部使用了MVP模式，需要初始化Presenter。

```
//初始化数据
private void initData() {
    presenter = new PLVPPTPresenter();
    presenter.init(this);
}
```

而PPT实现的View层接口是：

```
/**
 * PPTView
 */
interface IPLVPPTView {
    /**
     * 发送消息到webView
     *
     * @param msg 消息
     */
    void sendMsgToWebView(String msg);
    /**
     * 发送消息到webView
     *
     * @param msg   消息
     * @param event 事件
     */
    void sendMsgToWebView(String msg, String event);
    /**
     * 隐藏加载中图片
     * 直播时，则聊天室登录后，收到PPT控制消息时，隐藏加载中图片。
     * 回放时，在收到PPT的prepare回调后，异常加载中图片。
     */
    void hideLoading();
}
```