1 功能概述

打赏功能支持观众给主播赠送礼物，观众打赏界面被封装为打赏弹层 `PLVECRewardPopupView`，打赏礼物条被封装为 `PLVECRewardGiftAnimView`

2 使用演示

发送打赏礼物和监听他人打赏事件均依赖于聊天室的 `chatroomPresenter`，需要预先构建该 `Presenter` 对象

发送打赏示例代码如下：

```
// PLVECLiveHomeFragment.java
// 打赏弹层创建
PLVECRewardPopupView rewardPopupView = new PLVECRewardPopupView();

// 显示打赏弹层
private void showRewardLayout(View v) {
    // 显示打赏弹层，并设置点击打赏按钮时的回调响应
    rewardPopupView.showRewardLayout(v, new PLVECRewardGiftAdapter.OnViewActionListener() {
        @Override
        public void onRewardClick(View view, PLVCustomGiftBean giftBean) {
            // 隐藏打赏弹层
            rewardPopupView.hide();

            String nickName = PolyvSocketWrapper.getInstance().getLoginVO().getNickName();
            showRewardGiftAnimView(nickName + "(我)", giftBean);
            addCustomGiftToChatList(nickName + "(我)", giftBean.getGiftName(), giftBean.getGiftType(), true);
            // 通过自定义信息事件发送礼物信息至聊天室
            chatroomPresenter.sendCustomGiftMessage(giftBean, nickName + " 赠送了" + giftBean.getGiftName());
        }
    });
}
```

监听他人打赏事件示例代码如下：

```
// mvp-view 监听礼物事件
private IPLVChatroomContract.IChatroomView chatroomView = new PLVAbsChatroomView() {
    // 其它事件监听...

    // 礼物事件监听
    @Override
    public void onCustomGiftEvent(@NonNull PolyvCustomEvent.UserBean userBean, @NonNull PLVCustomGiftBean
customGiftBean) {
        showRewardGiftAnimView(userBean.getNick(), customGiftBean);
        addCustomGiftToChatList(userBean.getNick(), customGiftBean.getGiftName(), customGiftBean.getGiftType(),
false);
    }
}

// 显示打赏礼物条
private void showRewardGiftAnimView(String userName, PLVCustomGiftBean giftBean) {
    int giftDrawableId = getResources().getIdentifier("plvec_gift_" + giftBean.getGiftType(), "drawable",
getContext().getPackageName());
    rewardGiftAnimView.acceptRewardGiftMessage(
            new PLVECRewardGiftAnimView.RewardGiftInfo(userName, giftBean.getGiftName(), giftDrawableId)
    );
}
```

3 实现介绍

3.1 打赏弹层 - PLVECRewardPopupView

Demo中`PLVECRewardPopupView`使用了本地生成的打赏礼物列表作为演示，也可以接入后端实现动态的礼物列表。

```
// PLVECRewardPopupView.java
// 本地生成礼物列表
private void generateRewardGiftVO();
// 设置礼物列表
private void setRewardGiftVO(List<PLVCustomGiftBean> giftBeanList);
```

3.2 打赏礼物条 - PLVECRewardGiftAnimView

打赏礼物条`PLVECRewardGiftAnimView`通过调用其`acceptRewardGiftMessage(RewardGiftInfo)`方法进行礼物条的显示，会自动隐藏。

在内部方法`showRewardLayout()`中进行了礼物条动画的逻辑处理，可以根据实际需要进行修改。

```
// PLVECRewardGiftAnimView.java
// 外部调用 接收礼物数据
public void acceptRewardGiftMessage(final RewardGiftInfo rewardGiftInfo) {
    post(new Runnable() {
        @Override
        public void run() {
            rewardGiftInfoList.add(rewardGiftInfo);
            //列表最多10条数据。超过10条数据时，如果有新来的礼物数据，则移除旧的数据
            if (rewardGiftInfoList.size() > 10) {
                rewardGiftInfoList.remove(0);
            }
            if (!isStart) {
                isStart = !isStart;
                showRewardLayout();
            }
        }
    });
}

// 内部调用 礼物数据动画显示处理
private void showRewardLayout() {
    if (rewardGiftInfoList.size() < 1) {
        setVisibility(View.INVISIBLE);
        TranslateAnimation animation = new TranslateAnimation(0f, -getWidth(), 0f, 0f);
        animation.setDuration(400);
        startAnimation(animation);
        isStart = !isStart;
        return;
    }

    final RewardGiftInfo rewardGiftInfo = rewardGiftInfoList.remove(0);

    acceptRewardGiftDisposable = Observable.just(1)
            .observeOn(AndroidSchedulers.mainThread())
            .doOnNext(new Consumer<Integer>() {
                @Override
                public void accept(Integer integer) throws Exception {
                    setVisibility(View.VISIBLE);
                    rewardUserNameTv.setText(rewardGiftInfo.getUserName());
                    rewardGiftNameTv.setText("赠送    " + rewardGiftInfo.getGiftName());
                    rewardGiftPicIv.setImageResource(rewardGiftInfo.getGiftDrawableId());
                    TranslateAnimation animation = new TranslateAnimation(-getWidth(), 0f, 0f, 0f);
                    animation.setDuration(400);
                    startAnimation(animation);
                }
            })
            .delay(400 + 1200, TimeUnit.MILLISECONDS)
            .observeOn(AndroidSchedulers.mainThread())
            .subscribe(new Consumer<Object>() {
                @Override
                public void accept(Object o) throws Exception {
                    showRewardLayout();
                }
            }, new Consumer<Throwable>() {
                @Override
```

```
                public void accept(Throwable throwable) throws Exception {
                }
            });
}
```