00.来源

- 我们希望能够在推荐页中上下滑动切换到新视频时，视频可以自动的播放。因此我们在RecommendFragment中我们注册ViewPager2的OnPageChangeCallback页面改变回调，当上下滑动导致子条目变更时通过playCurVideo自动播放当前视频。
- 这个页面切换监听中我们覆写了onPageSelected方法，在内部调用了playCurVideo(position)逻辑
    - ```
      private val pageChangeCallback = object: OnPageChangeCallback() {
          override fun onPageSelected(position: Int) {
              playCurVideo(position) // 核心播放逻辑
          }
      }
      ```

01.playCurVideo

- playCurVideo接收一个Int类型position用于表示当前是RecyclerView中的第几个ViewHolder。其在内部做了如下事情：获取各视图元素，设置视图元素的回调方法，进行视频播放。
    - 获取视图元素。
        - 通过`recyclerView.findViewHolderForAdapterPosition(position)`查找对应位置的ViewHolder。返回其itemView。
        - 然后根据itemView获取当前ViewHolder的rootView，在从根视图中获取likeView，controllerView，ivPlay，ivCover等视图元素。

> 为什么需要rootView?
>
> ItemView类似于DecorView，rootView类似于DecorView下的最顶级的ViewGroup

    - ```
      /**
       * 通过适配器获取指定位置的根视图itemView，
       * 然后从该视图中查找多个UI组件，如rl_container、LikeView、ControllerView、播放按钮和封面图片。
       * 这些操作是为了获取新位置对应的视图元素，准备进行播放控制。
       */
      val itemView = adapter!!.getRootViewAt(position)
      /**
       * 通过position获取当前item.rootview
       */
      fun getRootViewAt(position: Int): View? {
          val viewHolder =
      recyclerView.findViewHolderForAdapterPosition(position)
          return if (viewHolder != null && viewHolder is VideoViewHolder) {
              viewHolder.itemView
          } else {
              null
          }
      }
      ```

```
val rootView = itemView!!.findViewById<ViewGroup>(R.id.rl_container)
val likeView: LikeView = rootView.findViewById(R.id.likeview)
val controllerView: ControllerView =
rootView.findViewById(R.id.controller)
val ivPlay = rootView.findViewById<ImageView>(R.id.iv_play)
val ivCover = rootView.findViewById<ImageView>(R.id.iv_cover)
```

- 设置likeView的单击播放暂停监听器。
  - likeView会在单击的时候调用该方法，会检查videoView是否在播放，如果是就暂停，显示播放按钮；否则开始播放，隐藏播放按钮。（LikeView不仅处理点赞，还处理播放暂停操作)
  - ```
    //播放暂停事件
        likeView.setOnPlayPauseListener(object:
    LikeView.OnPlayPauseListener {
            override fun onPlayOrPause() {
                if (videoView!!.isPlaying()) {
                    videoView?.pause()
                    ivPlay.visibility = View.VISIBLE
                } else {
                    videoView?.play()
                    ivPlay.visibility = View.GONE
                }
            }

        })
    ```

- 为controllerView设置点击事件监听实例
  - 用户在点击ControllerView中视图时会触发onClick回调，在内部调用事件监听实例的方法。
  - ```
        likeShareEvent(controllerView)

    /**
         * 处理用户的操作事件，比如点击头像、点赞、评论和分享。
         * 这里通过ControllerView设置监听器，并将事件通过RxBus发送出去，或者显示对话框。
         */
        private fun likeShareEvent(controllerView: ControllerView) {
            controllerView.setListener(object :
    OnVideoControllerListener {
                override fun onHeadClick() {
                    RxBus.getDefault().post(MainPageChangeEvent(1))
                }

                override fun onLikeClick() {}
                override fun onCommentClick() {
                    val commentDialog = CommentDialog()
                    commentDialog.show(childFragmentManager, "")
                }

                override fun onShareClick() {
                    ShareDialog().show(childFragmentManager, "")
                }
            })
    ```

```
    }
```

- 通过RxBus发送一个事件，传递上下滑动导致子条目变更时当前新的字条目的信息，然后让个人主页订阅这些数据。这样启动他们的时候就是当前子条目的用户信息了。

```
//切换播放视频的作者主页数据

RxBus.getDefault().post(CurUserBean(DataCreate.datas[position]?.user
Bean!!))
    curPlayPos = position
```

- 将视频播放器加入到RecommendFragment的根视图中。

```
dettachParentView(rootView)

/**
 * 将videoView从原来的父视图中移除，并添加到新的rootView中,我之前还很困惑为什么xml中没有videoView的布局定义呢，原来在这里啊。
 */
private fun dettachParentView(rootView: ViewGroup) {
    //1.添加videoView到当前需要播放的item中,添加进item之前，保证videoView没有父view
    videoView?.parent?.let {
        (it as ViewGroup).removeView(videoView)
    }

    rootView.addView(videoView, 0)
}
```

- 调用autoPlayVideo开始自动播放视频。

02.autoPlayVideo自动播放

- autoPlayVideo会自动播放指定位置的视频，同时隐藏ivCover视频封面。
- 播放视频：调用VideoPlayer的playVideo逻辑，传递媒体源。其在内部会执行设置媒体源，准备播放，开始播放。

```
/**
 * 播放指定位置视频
 */
videoView.playVideo(adapter!!.getDatas()[position].mediaSource!!)

/**
 * 播放视频的方法：本地缓存和直接url
 *
 * 本地缓存：设置媒体源，准备播放，开始播放
 */
fun playVideo(mediaSource: BaseMediaSource) {
    mPlayer.setMediaSource(mediaSource)
    mPlayer.prepare()
    mPlayer.play()
}
```

- 隐藏ivCover视频封面

- 通过设置mPlayer的Listener，在onRenderedFirstFrame，也就是mPlayer渲染第一帧时隐藏封面。

```
videoView.getplayer()?.addListener(object: Player.Listener {
    override fun onPlaybackStateChanged(state: Int) {
        // 播放状态发生变化时的回调
        if (state == Player.STATE_READY) {

        }
    }

    fun onPlayerError(error: ExoPlaybackException?) {
        // 播放发生错误时的回调
    }

    override fun onIsPlayingChanged(isPlaying: Boolean) {
        // 播放状态变为播放或暂停时的回调
    }

    override fun onRenderedFirstFrame() {
        //第一帧已渲染，隐藏封面
        ivCover.visibility = View.GONE
        ivCurCover = ivCover
    }
})
```