移动互联网技术及应用

大作业报告

姓名	班级	学号
武佳宝	**2017211311**	**2017211540**

2020.6

目录

1. 相关技术

本次作业我设计的视频列表主要采用 recycleview 模块，在项目中建立自定义的 adapter 来与 recycleview 相匹配。在 adapter 中需要使用 glide 进行视频封面的动态加载，用 Pullxml 对 xml 资源文件进行解析。

2. 系统功能需求

需求分析：

在视频列表中，需要对视频的封面截图，描述信息进行展示。

功能描述：

上下滑动展示视频列表，点击对应模块跳转指定视频播放页面。

3. 系统设计与实现

总体设计：

在视频列表中，我主要针对 item 进行设计，建立自定义的 adapter，完成建立，绑定和点击的函数功能，之后再 MainActivity 文件中，对列表进行建立，根据从 xml 资源文件中读取到的信息，创建对应的 item。

系统组成：

工程文件包含下列部分

MainActivity.java

PullParser.java

VideoAdapter.java

Videoitem.xml

Videolist.xml

模块设计：

Videoitem.xml 文件是构成视频列表中的 item 单元，最外层使用 framelayout 用于层叠显示，中间使用 imageView 和 TextView 分别用于展示视频封面和视频介绍。

VideoList.xml 文件是列表文件，主体就是一个 recycleView，中间的 item 选择之前创立的 VideoItem。

VideoAdapter 文件是创建的 adapter 类，继承于 RecycleView 的

adapter，在类中包括创建，绑定和获取数目三个函数。opublic NumberViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int viewType)函数，其中调用 inflate 函数创建 view，再根据 view 创建 viewHolder，最后返回创建的 viewholder。public void onBindViewHolder(@NonNull NumberViewHolder numberViewHolder, int position)绑定函数根据传入的 ViewHolder 和 position 信息进行绑定。public int getItemCount()获取 item 的个数。新建 NumberViewHolder 类，其中包括创建，绑定和点击三部分。创建就是将 xml 文件中的组件与变量名对应。绑定过程中，使用 glide 加载视频的第一帧作为封面截图。点击事件发生，获取对应的 description，在主函数中跳转匹配进行使用。

PullParser 是用来解析 xml 资源文件的类。在所给的资源中，我将其转换成 xml 文件进行解析。public static List<Message> pull2xml(InputStream is)函数将 xml 文件的标签类型进行匹配，调用对应的 message 类函数，将 xml 文件中的信息存储到创建的 List 中，方便之后的调用。

MainActivity 运行时，首先声明 recycleview 等信息，获取 xml 页面信息，创建列表管理，之后将 XML 资源文件中的信息读取到 list 表中，之后根据资源数目创建 adapter，运行成功。

public void onlistItemClick(String description)函数时在点击事件发生后，将 description 信息保存，于 list 列表中的 description 相比对，最后将 list 列表与对应序号存为 intent 传送到视频播放部分。

关键代码：

```
list = (ArrayList<Message>) PullParser.pull2xml(getResources().getAssets().open( fileName: "data.xml"));
List<Message> l = PullParser.pull2xml(getResources().getAssets().open( fileName: "data.xml"));
```

这部分代码是将 data.xml 资源文件读入到建立的 list 中，用于之后的 adapter 建立和绑定，之后会将这部分信息传送到视频播放部分。

```
num_list= list.size();
Log.d(TAG, String.valueOf(num_list));
vadapter = new videoAdapter(num_list,list, listener: this);
mListVideo.setAdapter(vadapter);
```

这部分是根据 list 建立 adapter

```
public void onlistItemClick(String description){
    int pos;
    for(pos=0;pos<list.size();pos++){
        Message ob = list.get(pos);
        if(ob.getDescription() == description)
            break;
    }
    Log.d(TAG,String.valueOf(pos));
    Intent intent = new Intent(getApplicationContext(),PlayerActivity.class);
    intent.putExtra( name: "videoList",list);
    intent.putExtra( name: "position",pos);

    startActivity(intent);

}
```

这部分是首先将之前点击事件留下的 description 与 list 列表中的信息相匹配得到视频的序号 pos，之后将 list 和 pos 封装发送给播放部分。

```
public NumberViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int viewType) {
    context = viewGroup.getContext();
    int layoutID_ListItem = R.layout.videoitem;
    LayoutInflater inflater = LayoutInflater.from(context);
    boolean attachtoparent = false;

    View view = inflater.inflate(layoutID_ListItem, viewGroup, attachtoparent);
    NumberViewHolder viewHolder = new NumberViewHolder(view);

    viewHolderCount++;
    return viewHolder;
```

这部分是 viewholder 的创建，根据 item，viewgroup 等信息建立。

```
public void bind(int position){
    Message messages = message.get(position);
    title_v.setText(messages.getDescription());
    //预览图加载
    Glide.with(context).setDefaultRequestOptions(
            new RequestOptions()
                .frame(1000)
                .centerCrop()
                .error(R.drawable.ic_launcher_background)
                .placeholder(R.drawable.ic_launcher_background)
    ).load(messages.getFeedurl())
            .into(photo_v);
```

这部分是将 xml 文件中的组件信息中变量信息的绑定获取。在这一部分使用 glide 对视频的截图信息进行截取并展示

```
    public void onClick(View v) {
        if (mOnClickListener != null) {
            mOnClickListener.onlistItemClick((String) title_v.getText());
        }
    }
}

public interface ListItemClickListener {
    void onlistItemClick(String description);
}
```

这部分是点击事件的产生后进行的 description 获取。

```
public class PullParser {
    //解析xml文件
    public static List<Message> pull2xml(InputStream is) throws Exception{
        List<Message> list = null;
        Message msg = null;
        //创建xml解析器
        XmlPullParser parser = Xml.newPullParser();
        //初始化解析器
        parser.setInput(is, inputEncoding: "utf-8");
        //读取文件类型
        int type = parser.getEventType();
        while(type!=XmlPullParser.END_DOCUMENT){
            switch(type){
                case XmlPullParser.START_TAG:
                    if("messages".equals(parser.getName())){
                        list = new ArrayList<>();
                    }else if("message".equals(parser.getName())){
                        msg = new Message();
                    }else if("_id".equals(parser.getName())){
                        String id = parser.nextText();
                        msg.setId(id);
                    }else if("feedurl".equals(parser.getName())){
                        String feedurl = parser.nextText();
                        msg.setFeedurl(feedurl);
                    }else if("nickname".equals(parser.getName())){
                        String nickname = parser.nextText();
                        msg.setNickname(nickname);
                    }else if("description".equals(parser.getName())){
                        String des = parser.nextText();
```

这部分代码是将 xml 文件解析并保存在 list，根据标签形式进行循环读取，直至读取完成

4. 系统可能的扩展

之后可以在视频列表中做更进一步的美化，可以对视频信息进行预加载，如展示片段等形式。

5. 总结体会

在最初的理论课上，认识到了互联网的发展历史，了解了很多新兴技术，在之后的实践课上对安卓开发技术有了更深的理解和认识，了解开发的不易。一路实验做过来还是遇到了不少的问题，踩过不少的坑，由于之前缺乏相关知识，入门起来有些吃力。做出来的东西也没有想象中的那么好。经过编写代码后，对安卓的开发有了更深刻的认识，从最初的连文件结构都分不清楚到最后完成大作业的编写，还是经过了不小的努力，回过头来看，之前一直没弄清的也其实很简单。希望自己的努力没有白费，能在每次学习中不断进步，让付出有回报。