**播放器实验报告**

**71115112 余泽晨**

**一、 实验目的**

* 通过自主开发一款安卓多媒体播放器应用程序（Player），实践软件体系结构与设计的方法，体会软件体系结构与设计的思想。

**二、 实验环境**

* Android Stduio(Min Sdk Version 19)+Win10

**三、 软件体系结构设计**

**软件重用**

* 由于android应用的开发已经有一个成熟的框架,因此基于这个框架,在Android Stduio的开发平台上新建项目进行开发,使用大量已经封装好的构件进行功能和用户界面的开发,可以提高开发速度,提高程序的健壮性和可维护性,体现软件重用的思想.

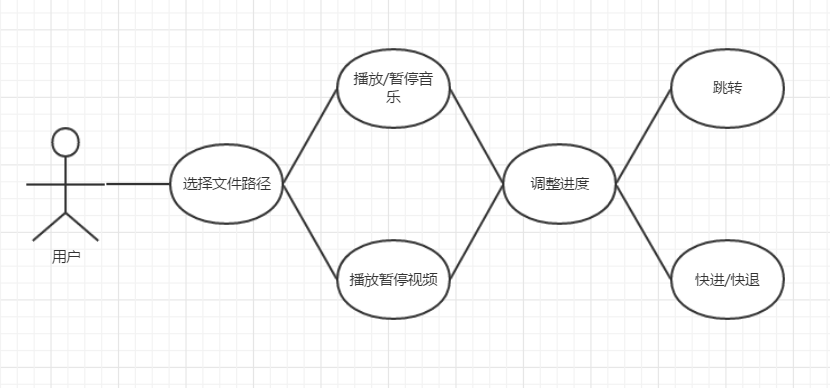
**管道-过滤器风格**

* 过滤出媒体文件,将数据传递给播放层.便于信息隐藏,使得程序符合高内聚,低耦合的设计原则.

**层次系统风格**

* 应用分成文件管理层和播放层.便于开发,易于修改,只要保持层与层的接口不变,修改各个分层的内容不影响其他层次的功能.同时支持重用,可移植性高

**场景视图**

* 

**抽象原理**

* 将各个过滤的过程抽象,作为封装的基础,降低系统复杂性,减少耦合.

**封装原理**

* 将各个过滤器封装成类,为信息隐藏提供支持,保证模块间的相对独立性.

**低耦合高内聚**

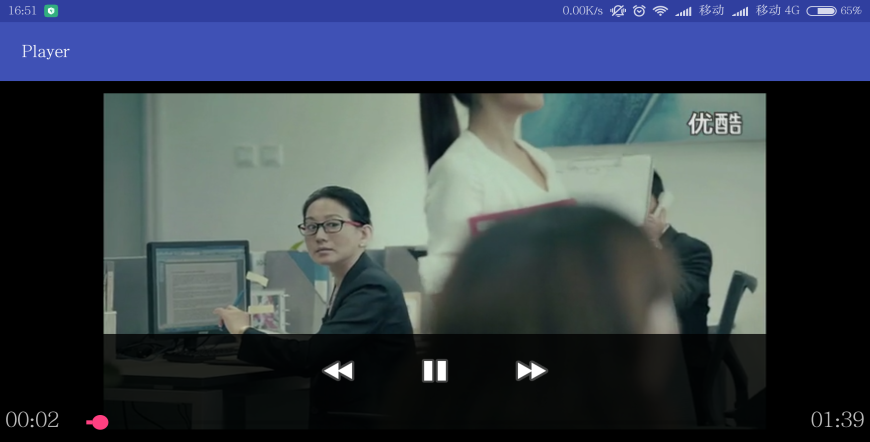
* 文件管理模块,播放模块仅有数据耦合,各个模块的功能内聚,基本符合模块设计的高内聚,低耦合要求.

**信息隐藏**

* 对用户隐藏部件的实现细节，更好的处理系统的复杂性和减少各模块之间的耦合。

**三、运行结果**

****

****

**四、改进思考**

* 增加播放列表,添加用户歌单,改善人机交互,进一步美化界面的UI,使布局更加合理,提高用户体验.
* 增加播放模式设置,添加切换上一首/下一首音乐的功能.

**五、实验总结**

在本门课程的学习中,我学到了许多关于软件体系结构方面的知识.在这次实验中,我加深了对于本门课程的理解和对学到的知识的掌握,尤其是对于软件体系结构设计方面有了更深的认识,对不同的软件体系结构风格也有了进一步的了解.在开发的过程中,我也感受到了软件体系结构的设计对于软件开发的指导意义.没有对开发的应用有足够的认知和准备,就不容易找到合适的软件体系结构风格,这样,在开发的过程中就容易违反软件体系结构的设计原则,不利于开发,导致开发周期延长,程序的可维护性,可移植性,重用性,健壮性等也难以得到保证.

在实验中,我也体会到软件重用的重要性,像开发android应用使用的Android studio提供的框架,构件和函数库使得开发难度大大降低.软件体系结构风格的重用也使得设计更加轻松.而对于我来说,最多的重用可能还是代码级的重用,像是对Filter类的继承和对于过滤过程的封装使得代码更加简洁,也提高了软件的可维护性.

在今后的开发过程中,希望能更多地运用软件体系结构的知识来指导自己的编程开发,提高自己的重用级别,完成一个自己的构件库,便于自己的开发,成为一名合格的软件工程师.

**六、附加(软件难点实现)**

* 文件管理
* private List<Map<String, Object>> getData(String path) {
* List<Map<String, Object>> list = new ArrayList<Map<String,Object>>();
* Map<String, Object> map = new HashMap<String, Object>();
* map.put(COLUMN\_NAME\_NAME, "..");
* list.add(map);
* File file = new File(path);
* if (file.listFiles().length > 0) {
* for (File f : file.listFiles()) {
* if (fileFilter(f)) {
* map = new HashMap<String, Object>();
* String name = f.getName();
* if (f.isDirectory()) {
* name += "/";
* }
* map.put(COLUMN\_NAME\_NAME, name);
* list.add(map);
* Log.d("Scan", " path " + path + f.getName());
* }
* }
* }
* return list;
* }
* @Override
* protected void onListItemClick(ListView l, View v, int position, long id) {
* super.onListItemClick(l, v, position, id);
* String path;
* if (position > 0) {
* pathHistory.push(curPath);
* path = curPath + itemList.get(position).get(COLUMN\_NAME\_NAME);
* } else { // uplevel
* if (!pathHistory.empty())
* path = pathHistory.pop();
* else // root
* path = curPath;
* }
* Log.d("List View Click", " position: " + position + " name: " + path);
* File file = new File(path);
* if (file.isDirectory()) {
* updateList(path);
* curPath = path;
* } else {
* Intent intent = new Intent(MainActivity.this, PlayerActivity.class);
* intent.putExtra("path", path);
* startActivity(intent);
* }
* }
* private void updateList(String path) {
* itemList.clear();
* itemList = getData(path);
* adapter = new SimpleAdapter(this, itemList, R.layout.list\_item, from, to);
* setListAdapter(adapter);
* adapter.notifyDataSetChanged();
* }
* private boolean fileFilter(File file) {
* if(file.isDirectory()) {
* return true;
* } else {
* String name = file.getName();
* if (name.endsWith(".mp3") || name.endsWith(".mp4"))
* return true;
* else
* return false;
* }
* }