简易Android音视频播放器的开发

71Y15102 孟越

1. 实验要求

1、界面美化

2、链接音视频

3、已有曲目或自选曲目的加入

1. 实验内容

实验要求能基本实现，另外添加少许特色功能。

软件结构为：

直接进入主界面（打开按钮，播放屏幕，文字）；

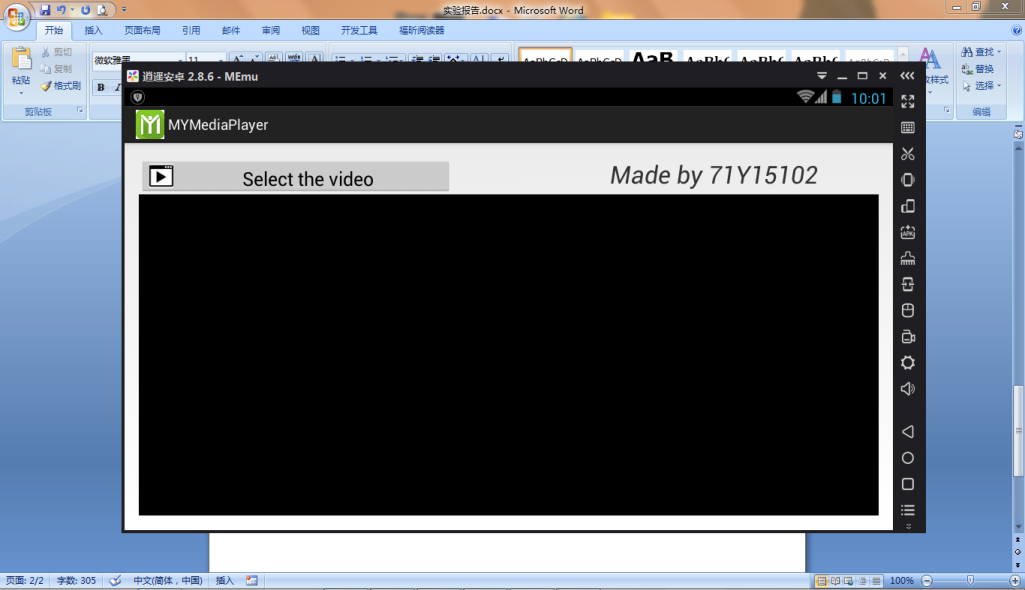
点击打开按钮后，会进入系统文件管理器，直接在相关文件的路径下，选择音视频双击自动转到主界面进行播放；

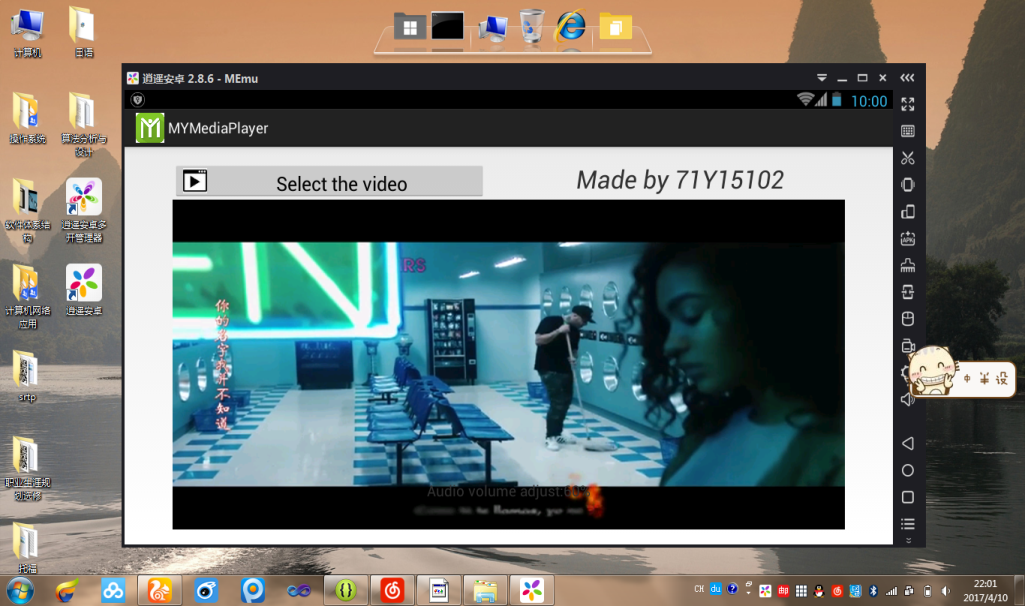
添加播放控制器，可随时暂停,快进，后退等操作，另外可拖动滑块进行随意的进度调节；

添加触屏功能，屏幕左半部分拖滑可调节屏幕亮度，屏幕右半部分上下拖滑可调节音视频音量，并把拖动比例在屏幕中下部分显示出来。

1. 实验步骤及结果
2. 配置相关环境，搜集资料，为开发Android软件做准备
3. 使用逍遥安卓模拟器，更加方便地对自己的软件进行测试
4. 分析自己所想要开发的播放器的功能，并将功能整合进一个完整的软件结构中
5. 参考别人的播放器，完成整体的架构设计
6. 写出播放框架，能够播放res中的音视频并对其进行基本操作
7. 添加从sdcard 中打开音视频的功能
8. 添加滑屏控制亮度及音量大小的功能
9. 界面美化，插入相关的图标和文字
10. 完成实验报告，对实验过程进行分析与反思

播放器部分截图：





1. 实验过程分析及反思

A.在项目初期一筹莫展的时候，应用到课上的相关知识：

由于软件系统比较简单，于是采用简单的管道-过滤器风格和数据抽象和OO风格；

画出4+1视图，明确软件开发方向：

1. 逻辑视图：不同功能封装起来，根据播放器逻辑进行调用，关注了各部分不同的功能，如不同控件的逻辑，按钮打开文件管理器，触屏时间控制音量大小等等；
2. 开发视图：表现出播放器整体的层次结构，关注每一部分具体代码的实现，与功能一一映射；

c. 进程视图：整体软件运行时的过程，软件运行时动态结构的变化；

d. 物理视图：软件与硬件的部署，例如触屏操作与电容屏及其电路信号相关；

e. 场景：汇总，画出最终想要实现播放器效果的大致框架。

B．相关技术具体代码实践：

a.触屏操作:重写触屏事件函数，对不同位置滑屏进行判断并量化；

@Override

**public** **boolean** onTouchEvent(MotionEvent event) {

DisplayMetrics screen = **new** DisplayMetrics();

getWindowManager().getDefaultDisplay().getMetrics(screen);

**if** (mSurfaceYDisplayRange == 0)

mSurfaceYDisplayRange = Math.*min*(screen.widthPixels, screen.heightPixels);

**float** x\_changed, y\_changed;

**if** (mTouchX != -1f && mTouchY != -1f) {

y\_changed = event.getRawY() - mTouchY;

x\_changed = event.getRawX() - mTouchX;

} **else** {

x\_changed = 0f;

y\_changed = 0f;

}

**float** coef = Math.*abs* (y\_changed / x\_changed);

**float** xgesturesize = ((x\_changed / screen.xdpi) \* 2.54f);

**float** delta\_y = Math.*max*(1f,((mInitTouchY - event.getRawY()) / screen.xdpi + 0.5f)\*2f);

**switch** (event.getAction())

{

**case** MotionEvent.*ACTION\_DOWN*:

// Audio

mTouchY = mInitTouchY = event.getRawY();

mVol = mAudioManager.getStreamVolume(AudioManager.*STREAM\_MUSIC*);

mTouchAction = *TOUCH\_NONE*;

// Seek

mTouchX = event.getRawX();

**break**;

**case** MotionEvent.*ACTION\_MOVE*:

**if** (mTouchAction != *TOUCH\_SEEK* && coef > 2)

{

**if** (Math.*abs*(y\_changed/mSurfaceYDisplayRange) < 0.05)

**return** **false**;

mTouchY = event.getRawY();

mTouchX = event.getRawX();

**if** ( (**int**)mTouchX > (3 \* screen.widthPixels / 5)){

doVolumeTouch(y\_changed);

}

**if** ( (**int**)mTouchX < (2 \* screen.widthPixels / 5)){

doBrightnessTouch(y\_changed);

}

}

**break**;

**case** MotionEvent.*ACTION\_UP*:

**if** (mTouchAction == *TOUCH\_SEEK*)

mTouchX = -1f;

mTouchY = -1f;

**break**;

}

**return** mTouchAction != *TOUCH\_NONE*;

}

b.调节音量：从触屏事件中量化滑动距离，通过AudioManager设置音量大小；

**private** **void** doVolumeTouch(**float** y\_changed) {

**if** (mTouchAction != *TOUCH\_NONE* && mTouchAction != *TOUCH\_VOLUME*)

**return**;

**float** delta = - ((y\_changed / mSurfaceYDisplayRange) \* mAudioMax);

mVol += delta;

**int** vol = (**int**) Math.*min*(Math.*max*(mVol, 0), mAudioMax);

**if** (delta != 0f) {

setAudioVolume(vol);

}

}

**private** **void** setAudioVolume(**int** vol) {

mAudioManager.setStreamVolume(AudioManager.*STREAM\_MUSIC*, vol, 0);

**int** newVol = mAudioManager.getStreamVolume(AudioManager.*STREAM\_MUSIC*);

**if** (vol != newVol)

mAudioManager.setStreamVolume(AudioManager.*STREAM\_MUSIC*, vol, AudioManager.*FLAG\_SHOW\_UI*);

mTouchAction = *TOUCH\_VOLUME*;

vol = vol \* 100 / mAudioMax;

//**TODO**

Log.*i*("LanSoSdk","Audio valume adjust:"+Integer.*toString*(vol) + '%');

tvSlideHint.setText("Audio volume adjust:"+Integer.*toString*(vol) + '%');

}

1. 打开文件管理器：点击按钮后直接调用系统文件管理器，获取选取的文件即可。

**public** **void** OpenSystemFile(View v) {

Intent intent = **new** Intent(Intent.*ACTION\_GET\_CONTENT*);

intent.setType("video/\*");

intent.addCategory(Intent.*CATEGORY\_OPENABLE*);

**try** {

startActivityForResult(Intent.*createChooser*(intent, "请选择文件!"),

*FILE\_SELECT\_CODE*);

} **catch** (android.content.ActivityNotFoundException ex) {

Toast.*makeText*(**this**, "请安装文件管理器", Toast.*LENGTH\_SHORT*).show();

}

}

1. 实验改进

遗憾的是，还有一些最初想要实现的功能未能完美实现。

可改进的地方：

触屏左右滑动调节视频进度；

从网络直接搜索有关音视频（不仅仅是输入http），下载并打开；

双击全屏，单击暂停/播放；

丰富软件体系结构层次，是软件更具有实用性。

1. 实验收获

软件重用的好处：

通过软件重用的方法，我借鉴了一些非播放器视频软件中的直接调度系统播放器的方法，一方面此类构件的应用方便了我对软件的维护与修改，另一方面，极大地提高了我开发软件的效率。

几大设计原理的应用：

抽象原理，封装原理，信息隐藏原理，模块化原理，耦合和内聚原理，接口实现和分离原理，分而治之原理，注意点分离原理等在整个开发过程种都有体现，相辅相成。

例如：对触屏功能实现的封装，分别实现不同功能在进行整合，

每个功能作为一个控件，它们内部高度内聚代码不便修改，但可通过一定的约束关系进行耦合等等。在软件开发过程中，我们可以考虑不同地方对设计原理的应用，加以理解改进，便于我们构建SA，使得我们的代码最为美观简洁，我们的程序更加具有可维护性和灵活性。

SA相关知识的巩固和掌握：

在开发前夕经常性用到SA相关知识，在反思时又再一次地回顾，一个简单的软件开发过程，却饱含了软件开发规范性的相关知识，也加强了我对所学知识的理解。

体系结构设计的好处：

实际开发时发现，通过软件体系结构设计，能够便于我们理解所需要开发软件的结构，能够极大地提高效率并保证开发软件的规范性，最终软件成品的可维护性与实用性，也便于我对其进行修改与优化。

对Android的熟悉:

掌握Android相关知识，有了Android开发经验，受益匪浅。

对开发一个软件系统过程的规范化：

从曾经的“走一步看一步”的软件开发过程相比，其好处不言而喻，整个开发过程中思路清晰，对以后参与其他软件开发有着宝贵启示。