中山大学移动信息工程学院本科生实验报告

（2016 年秋季学期）

课程名称：移动应用开发　　任课教师：郑贵锋

年级	14	专业（方向）	物联网
学号	14353259	姓名	商家煜
电话	13211143626	Email	shangjy3@mail2.sysu.edu.cn
开始日期	2016.9.22	完成日期	2016.9.22

一、 实验题目

服务与多线程--简单音乐播放器

二、 实现内容

1. 学会使用MediaPlayer；

2. 学会简单的多线程编程，使用Handle 更新UI；

3. 学会使用Service 进行后台工作；

4. 学会使用Service 与Activity 进行通信

实现一个简单的播放器，要求功能有：

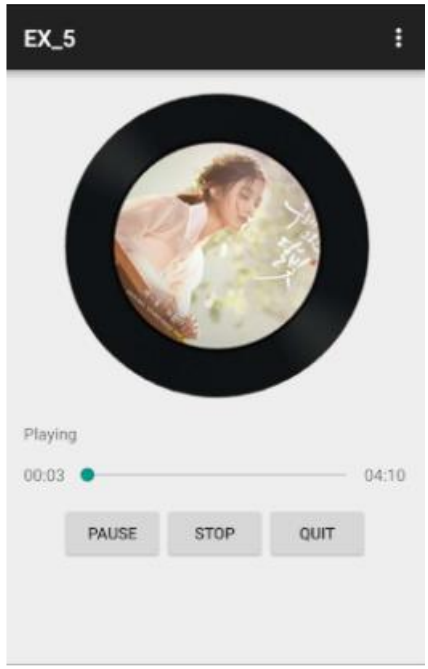
1. 播放、暂停，停止，退出功能；

2. 后台播放功能；

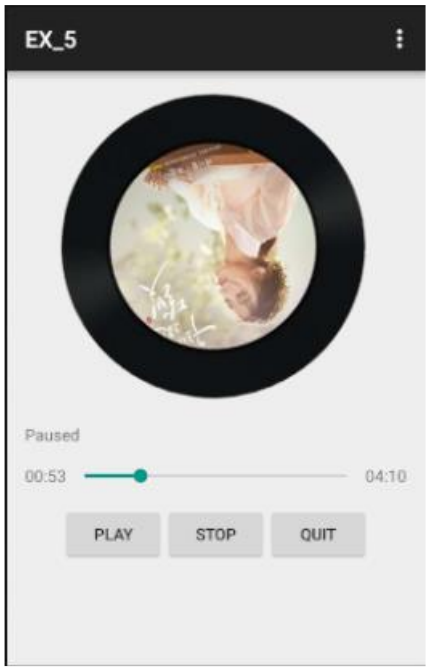
3. 进度条显示播放进度、拖动进度条改变进度功能；

4. 播放时图片旋转，显示当前播放时间功能；

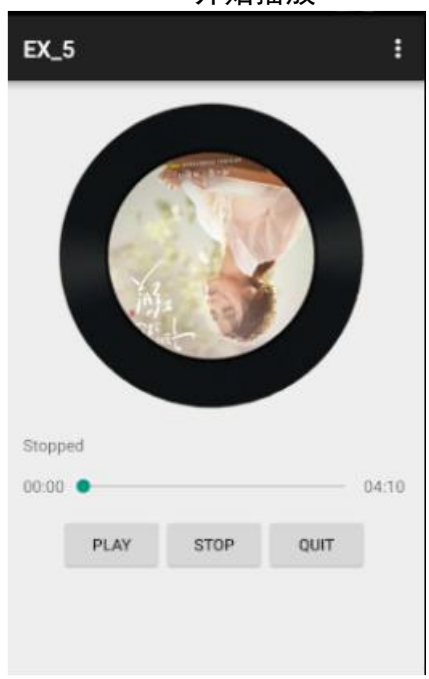
打开程序主页面



开始播放



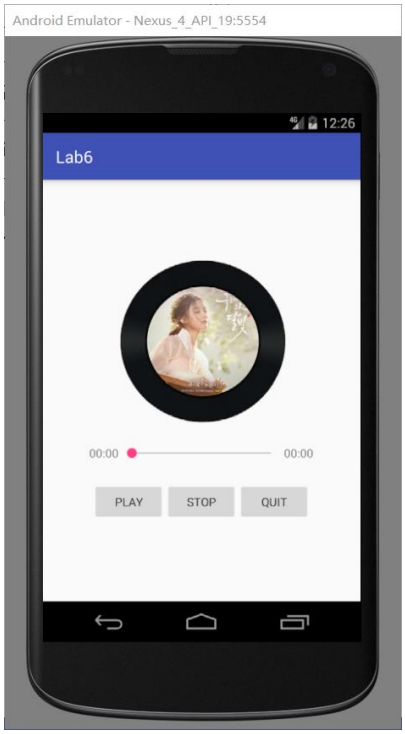
暂停

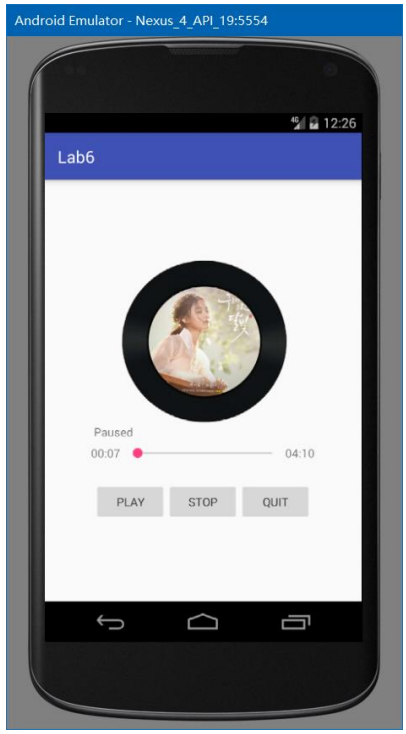


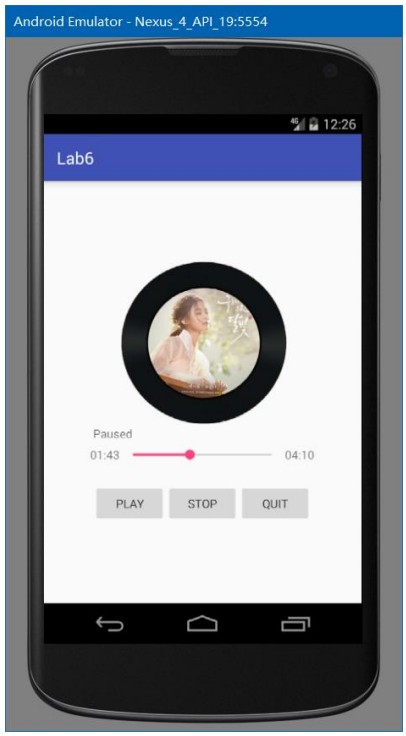
停止

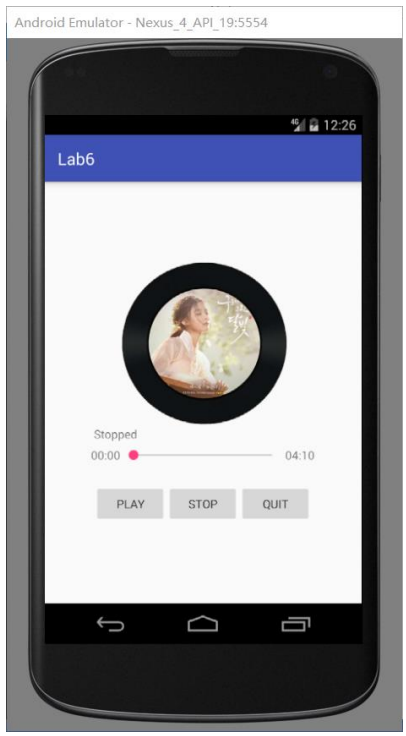
三、 课堂实验结果

（1） 实验截图









（2） 实验步骤以及关键代码

1. MainActivity

- 定义一个 findView 函数用来将所有的组件和 id 进行绑定

```
private void findView(){
    playBtn = (Button) findViewById(R.id.PLAY);
    stopBtn = (Button) findViewById(R.id.STOP);
    exitBtn = (Button) findViewById(R.id.QUIT);
    infoOperatingIV = (ImageView) findViewById(R.id.Image);
    seekBar = (SeekBar) findViewById(R.id.Seekbar_music);
    text1 = (TextView) findViewById(R.id.current_time);
    text2 = (TextView) findViewById(R.id.hole_time);
    text3 = (TextView) findViewById(R.id.playing_statue);
```

- 创建一个 Serviceconnection 实例以为了后续部分与 Service 进行链接并调用 Service 提供的方法，创建的过程中需要判断该 Service 是否被已经被链接。作用是监听访问者和 Service 之间的链接情况

```
private ServiceConnection sc = new ServiceConnection(){
    public void onServiceDisconnected(ComponentName name) { musicService = null; }
    public void onServiceConnected(ComponentName name, IBinder service){
        musicService = ((MusicService.MyBinder)(service)).getService();
    }
};
```

- 对 Service 进行链接，其中通过调用 bindService 方法，让 Service 和访问者之间能够进行方法的调用以及数据的交换。。其中 intent 包含了要启用的 Service，sc 为上一步中提到的用于监听访问者和 Service 之间的链接情况，最后一个参数表示绑定时自动创建 Service

```
private void connection(){
    Intent intent = new Intent(this, MusicService.class);
    bindService(intent, sc, Context.BIND_AUTO_CREATE);
}
```

- 设置一个 bindButton 类来进行对 button 的监听操作，以第一个按钮为例子，第一个按钮的功能为点击后为播放和暂停的转换。因此需要在进行操作之前，当监听到按钮被按下时判断当前文本文字是播放或暂停来进行对应的操作。播放操作中，把文字修改为暂停，并调用 Service 中的方法使音乐播放。并判断是否为第一次按下该按钮，如果是则需要读取 Service 中的媒体长度来设置 UI 界面中的滑动条和时间显示，并且将 mRunnable 丢入线程队列中。同时启动图片旋转功能 onAnimator.start()。

```
playBtn.setOnClickListener(new View.OnClickListener(){
    int flag = 1;
    @Override
    public void onClick(View v){
        if(playBtn.getText().toString().equals("PLAY")){
            text3.setText("Playing");
            musicService.paly();

            playBtn.setText("PAUSE");
            if(flag == 1){
                seekBar.setMax(musicService.getMusicDuration());
                text2.setText(time.format(musicService.mp.getDuration()))
                oaAnimator.start();
                mHandler.post(mRunnable);
                flag = 0;
            }
            else
                oaAnimator.resume();
```

如果不是第一次按下该按钮，则图片旋转应该为重新从当前位置开始，而非重新启动。其余的按钮功能基本与之一致，因此不再赘叙。

- seekbar 监听器需要进行判断当前 seekBar 是否是在被用户操作而不是被 Service 进行更改，否则将会产生卡顿的现象，这里，我们有一个布尔类型进行判断，如果为真，则对 Service 中的 seekto 方法进行调用

```
seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
        if(fromUser)
            musicService.seekTo(progress);
    }
    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {
    }
    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
    }
});
```

- Handler 与 UI 是同一线程，这里可以通过 Handler 更新 UI 上的组件状态，Handler 有很多方法，这里使用比较简便的 post 和 postDelayed 方法。在 Runnable 中，我们需要进行对 seekbar 进度的更改和 text 文本显示时间的更改。然后通过 postDelay 的方法，延迟 1 秒将自身再重新放入队列中

```
mHandler = new Handler();
mRunnable = (Runnable) () -> {
        seekBar.setProgress(musicService.getMusicCurrentPosition());
        text1.setText(time.format(musicService.getMusicCurrentPosition()));
        mHandler.postDelayed(mRunnable,1000);
};
```

2. Service

- 首先创建一个 MediaPlayer 实例，然后再初始化函数中对其进行初始化，得到音乐的路径并调用 prepare 方法和设置无限循环来等待进一步操作。其中因为调用的是 io 接口，所以必须通过 try，catch 的方法来进行异常的捕获

```
public static MediaPlayer mp = new MediaPlayer();
public MusicService() {
    try{
        mp.setDataSource("/data/K.Will-Melt.mp3");
        mp.prepare();
        mp.setLooping(true);
    }
    catch (Exception e){
        e.printStackTrace();
    }
```

- 通过Binder 来保持Activity 和Service 的通信。onBind()只有采用Context.-bindService()方法启动服务时才会回调该方法。该方法自调用者和服务绑定中被调用。当调用者与服务已经绑定多次调用Context.bindService方法并不会导致该方法被多次调用。采用。

```
@Override
public IBinder onBind(Intent intent) { return binder; }

public final IBinder binder = new MyBinder();
public class MyBinder extends Binder {
    MusicService getService() { return MusicService.this; }
}
```

- 封装函数中，将 Service 中提供的修改 mediaplayer 当前状态的方法进行封装，以便使用者进行调用。其中需要注意的是需要判断当前的 media 是否为空后再进行对应的操作。并同样用到 IO 接口的需要通过异常捕获来进行

```
public void paly() { mp.start(); }
public void stop(){
    if(mp != null){
        mp.stop();
        try{
            mp.prepare();
            mp.seekTo(0);
        }catch (Exception e){
            e.printStackTrace();
        }
    }
}
```

- 同样的原理设置拖动条所需要用到的函数即可

```
public void seekTo(int position){
    if (mp != null){
        mp.seekTo(position);
    }
}

public int getMusicDuration(){
    int rtn = 0;
    if (mp != null){
        rtn = mp.getDuration();
    }
    return rtn;
}
```

四、 实验思考及感想

本次实验课基本过程根据老师的 ppt 即可实现，实现的过程中碰到了音乐卡顿的问题，后来 debug 后发信，是没有判断拖动条是否为被用户操作还是后台进行的修改，对滑动条进行条件判断之后程序即可完美运行。