中山大学数据科学与计算机学院本科生实验报告

（2017 年秋季学期）

课程名称：手机应用平台开发　　任课教师： 刘宁

年级	15 级	专业（方向）	软件工程计算机应用
学号	15331236	姓名	马朝露
电话	15521122675	Email	1073627941@qq.com
开始日期	2017/11/21	完成日期	2017/11/22

一、 实验题目

服务与多线程--简单音乐播放器

二、 实现内容

【实验目的】

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。

【实验内容】

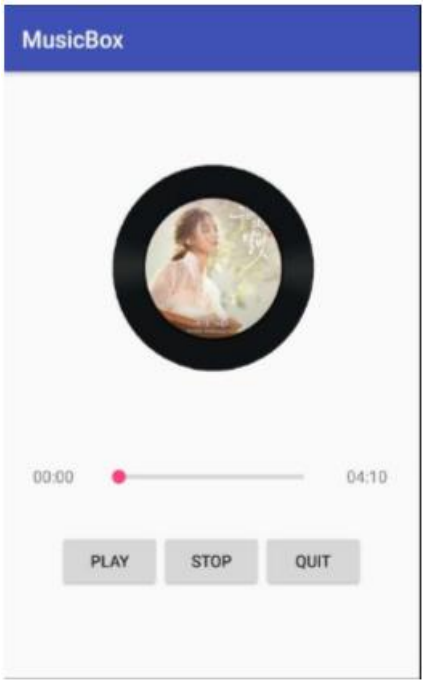
实现一个简单的播放器，要求功能有：

1. 播放、暂停，停止，退出功能；

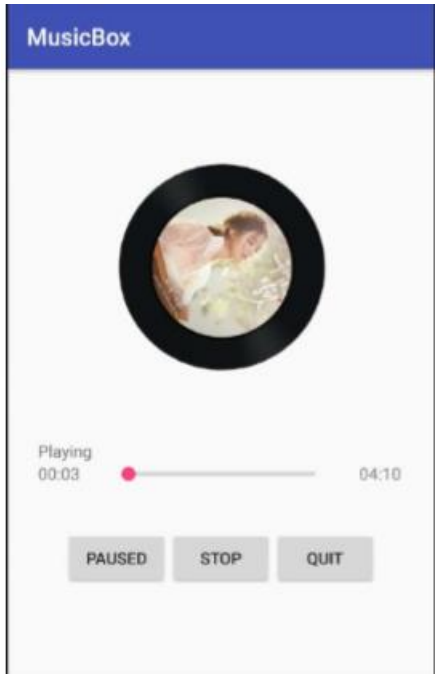
2. 后台播放功能；

3. 进度条显示播放进度、拖动进度条改变进度功能；

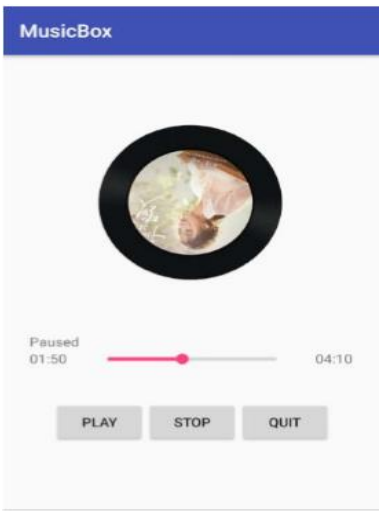
4. 播放时图片旋转，显示当前播放时间功能；



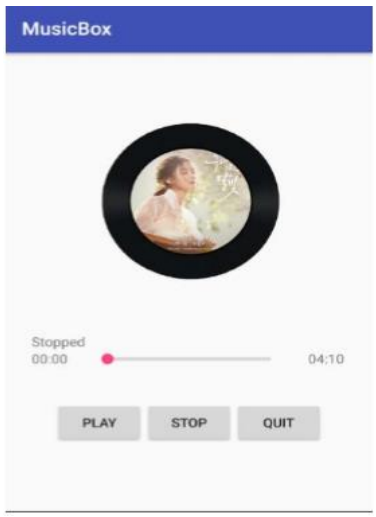
打开程序主页面



开始播放



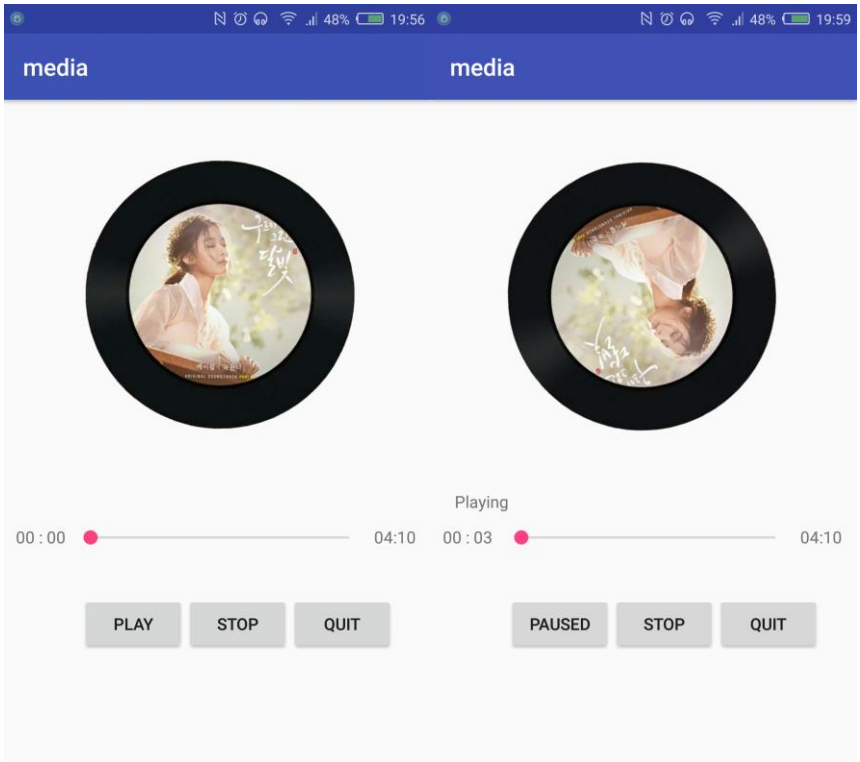
暂停



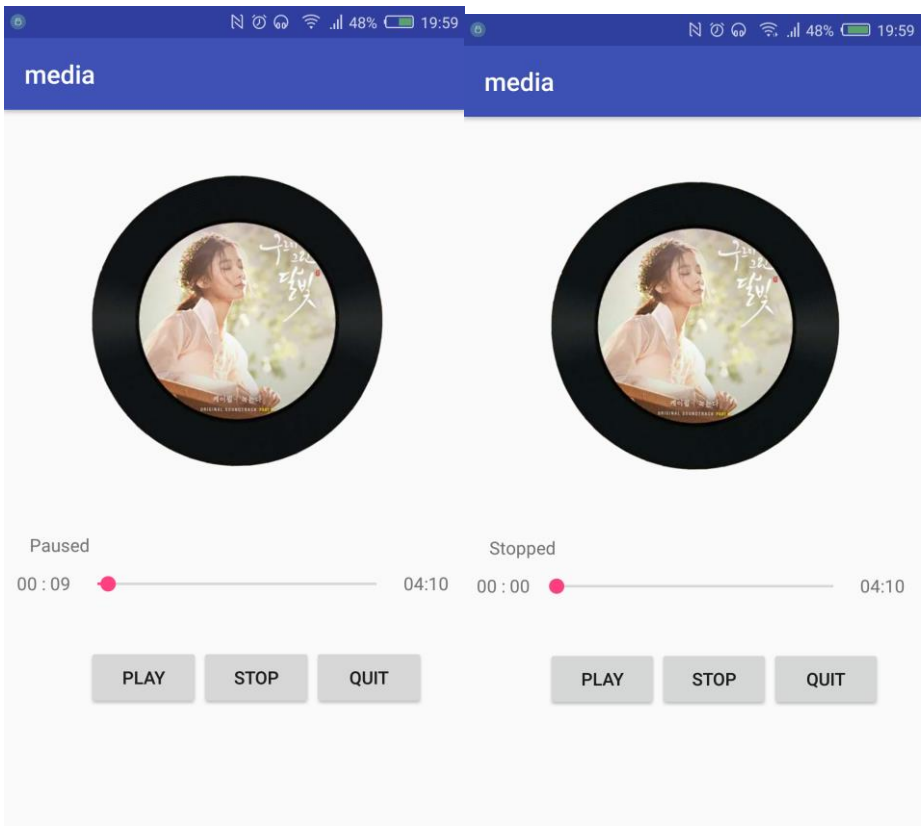
停止

三、 课堂实验结果

（1） 实验截图



打开程序主页面　　开始播放



暂停　　停止

（2） 实验步骤以及关键代码

【1】 界面的设计

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context="com.example.zhaoluma.media.Media">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <ImageView
            android:id="@+id/image"
            android:layout_width="250dp"
            android:layout_height="250dp"
            android:layout_marginTop="40dp"
            android:layout_gravity="center_horizontal"
            android:src="@drawable/image" />
    </LinearLayout>

    <RelativeLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="40dp">
        <TextView
            android:id="@+id/state"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text=""
            android:layout_marginLeft="20dp"
            />
        <TextView
            android:id="@+id/time"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="10dp"
            android:layout_alignTop="@+id/state"
            android:layout_marginTop="30dp"
            android:text="00:00"
        />
        <SeekBar
            android:id="@+id/bar"
            android:layout_width="250dp"
            android:max="100"
            android:layout_height="wrap_content"
            android:layout_toRightOf="@+id/time"
            android:layout_alignBottom="@+id/time"
            android:layout_marginLeft="5dp"/>
        <TextView
            android:id="@+id/stoptime"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
```

```
<RelativeLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="40dp"
    android:layout_marginLeft="65dp">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/button1"
        android:text="PLAY"
        />
    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@+id/button1"
        android:text="STOP"/>
    <Button
        android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@+id/button2"
        android:text="QUIT"/>
</RelativeLayout>
```

```
/LinearLayout>
```

整体使用线性布局，中间嵌套两个相对布局，一个用于实现时间进度条，一个用于实现按钮

【2】 Service 的实现

在 AndroidManifest.xml 文件里注册 Service：

```
<service android:name=".Myservice"
    android:exported="true"/>
```

通过 Binder 来保持 Activity 和 Service 的通信

```
/*
* 绑定服务的实现流程：
* 1.服务 onCreate， onBind， onDestroy 方法
* 2.onBind 方法需要返回一个 IBinder 对象
* 3.如果 Activity 绑定，Activity 就可以取到 IBinder 对象，可以直接调用对象的方法
*/

// 相同应用内部不同组件绑定，可以使用内部类以及Binder对象来返回。
public class MyBinder extends Binder {//定义一个类实现IBinder接口（ Binder实现了IBinder接口）
    Myservice getService() {
        return Myservice.this;
    }
}
@Override
public IBinder onBind(Intent intent) { //在onBind(Intent intent)中返回IBinder对象
    return binder;
}
```

在服务类中 oncreate 中给音乐播放器导入·音乐，设置音乐播放器的·属性，准备音乐播放器

```
@Override
public void onCreate() {
    super.onCreate();

    try{
        //player.setDataSource()
        player.setDataSource(Environment.getExternalStorageDirectory().getAbsolutePath() +
                "/melt.mp3");
        player.prepare();
        player.setLooping(true);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

在 service 类中实现播放，暂停，停止函数，方便与前台进程通信

```
public void play() {

    player.start(); // 开启音乐并启动动画

}
public void paused() {

    player.pause(); // 暂停音乐并停止动画

}
public void stop() {   // 停止音乐
    if (player != null) {
        player.stop();

        try {
            player.prepare();
            player.seekTo(0);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

实现 onDestroy 函数，释放资源

```
@Override
public void onDestroy() {
    if (player.isPlaying()) {
        player.stop();
    }
    player.release();
    super.onDestroy();
}
```

activity 中定义 service 类类型变量：private Myservice myservice; 与服务端连接这样就可以互相通信，通过 connection 获取了服务端类的实例

```
    myservice = new Myservice();

    // 连接绑定服务端
    Intent intent = new Intent(this, Myservice.class);
    startService(intent);
    /*第二参数是一个ServiceConnection对象，该对象用于监听访问者与service之间的连接情况，
    当访问者与Service连接成功时会回调该类的onServiceConnected(ComponentName name, IBinder service)方法，
    然后将服务中创建的Ibinder对象（此时在Service的onBinder方法中需要返回该Ibinder对象）传递给第一个参数intent，
    通过该Ibinder对象就能与Service进行通信。第三个参数flags指定绑定时是否自动创建Service，
    一般我们指定为BIND_AUTO_CREATE（自动创建，如果传入0表示不自动创建）*/
    bindService(intent, connection, BIND_AUTO_CREATE );

}
private ServiceConnection connection = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        //Toast.makeText(Media.this,"test",Toast.LENGTH_SHORT).show();
        myservice = ((Myservice.MyBinder)service).getService();
        rotate();
    }
    @Override
    public void onServiceDisconnected(ComponentName name) { myservice = null; }
};
```

【3】 mainactivity 的实现

play，stop，paused，quit 的实现

```
play.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        // Toast.makeText(Media.this,"test",Toast.LENGTH_SHORT).show();
        if (play.getText().toString().equals("PLAY")) {
            myservice.play();
            play.setText("PAUSED");
            state.setText("Playing");
            if (!objectAnimator.isRunning()) {
                objectAnimator.start();
            } else {
                objectAnimator.resume();
            }
        } else if (play.getText().toString().equals("PAUSED")) {
            myservice.paused();
            play.setText("PLAY");
            state.setText("Paused");
            if (objectAnimator.isRunning()) {
                objectAnimator.pause();
            }
        }
    }
});
```

play 和 paused 公用一个按钮，所以每次点击按钮时要判断按钮的 text 在确定执行什么操作，同时对应相应的图片旋转动画，如果 play 则开始动画开始，否则动画暂停。

```
stop.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        play.setText("PLAY");
        state.setText("Stopped");
        myservice.stop();
        objectAnimator.end();
    }
});
```

stop 按钮的监听函数要注意的是这时要结束动画，同时停止服务端的播放器，并重新加载服务端的音乐播放器，播放进度重设。

```
quit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        handler.removeCallbacks(runnable);
        unbindService(connection);
        try {
            System.exit(0);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});
```

quit 按钮的监听函数要实现的时，与服务端解除绑定，同时 handler 移除回调。

进度条的实现：通过 Handler 发送 Runnable 对象，每隔一段时间都重新实现 Runnable 中的 run 函数，run 相当于 handler 的任务，即 handler 循环处理 run 任务。每次都要判断播放器的状态，以便更新时间和状态 textview。每次 seekbar 都要设置滑块的位置，这里我也同时设置了进度条的最大长度，其实这只要设置一次就可

以了，因为这次实验只有一首歌要播放。设置 seekbar 的进度改变事件。当拖动滑块时，时间随着滑块的位置的改变而改变。音乐设置到拖动结束后滑块的位置。这里我没有使用 thread，使用了相对较简单的 runnable。Runnable 是一个接口，Thread 是 Runnable 的子类

```
public Handler handler = new Handler();
public Runnable runnable = new Runnable() {
    @Override
    public void run() {
        if(myservice.player.isPlaying()) {
            state.setText("Playing");
            play.setText("PAUSED");
        } else {

            //state.setText("Paused");
            play.setText("PLAY");
        }
        time.setText(getPlayingTime(myservice.player.getCurrentPosition()));
        seekbar.setProgress(myservice.player.getCurrentPosition());
        seekbar.setMax(myservice.player.getDuration());
        seekbar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
            @Override
            public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
                time.setText(getPlayingTime(progress));
            }
            @Override
            public void onStartTrackingTouch(SeekBar seekBar) {

            }
            @Override
            public void onStopTrackingTouch(SeekBar seekBar) {
                // 寻找指定的时间位置
                myservice.player.seekTo(seekBar.getProgress());
            }
        });
        handler.postDelayed(runnable, 100);
    }
};
```

时间 Textview 的实现：时间根据进度条的进度而设定。

```
//播放时间变化
private String getPlayingTime(int progress) {
    int second = progress/1000;
    int min = second / 60;
    second %= 60;
    if (second < 10)
        return "0"+min+" : 0"+ second;
    return"0"+min+" : "+ second;
}
```

（3） 实验遇到困难以及解决思路

1. 只要一绑定服务端，就闪退。解决方法：参考老师给的代码

```
<service android:name=".MyService"
         android:process="serviceApplication"/>
```

这 样 就 会 报 错 ， 参 考 TA 给 的 代 码 ：

```
<service android:name=".MusicService" android:exported="true"/>
```

2.动态获取文件阅读权限出错，程序闪退。

解决方法：代码中不报错，很难找到自己错在哪里，后来打包成 apk 时提示报错信息，说是 <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>的位置错误，我把它放到了 application 里，应该与 application 平行，改过之后就好了。

3.后台播放重新转到应用界面时，动画不转了。

解决方法：在 resume 中判断动画和音乐播放器的状态，如果音乐播放器正在播放，且动画停止了，则重新开始旋转

四、 实验思考及感想

1.这次实验总体来不是特别难，因为网上有很多实现音乐播放器的代码可以参考。但是 handler，runnable，thread 的概念有些复杂难懂。通过代码的实践稍微理解了皮毛。通过这次实验稍微理解了安卓的多线程。在我看来这次实现的进度条的 handler 就像一个一直在监听的监听器，每当一段时间过后就执行一些操作。

2.这次实验用到了后台播放，所以了解了安卓的服务类，服务是一种可以在后台执行长时间运行操作而没 有用户界面的应用组件。Service 可以与访问者之间基本不存在太多关联，因此 Service 和访问者之间无法通讯和数据交换。。也可以与访问者之间有信息交换，若 Service 和访问者之间需要进行方法调用或数据交换，则应该 使用：bindService(intent service, ServiceConnection conn, int flags)。

3.因为这次实验，又加深了我对 activity resume 状态的理解，就是当进程没有停止，但是你看不见进程的的界面之后重新打开进程时要执行的操作，与 paused 相对应。

作业要求：

1. 命名要求: 学号_姓名_实验编号，例如 15330000_林 XX_lab1。
2. 实验报告提交格式为 pdf。
3. 实验内容不允许抄袭，我们要进行代码相似度对比。如发现抄袭，按 0 分处理。