



中山大學
SUN YAT-SEN UNIVERSITY

《手机平台应用开发》 服务与多线程 ——简单音乐播放器

学 院 名 称 : 数据科学与计算机学院

成 员 : 陈伟宸

时 间 : 2016 年 11 月 3 日

一、实验环境

操作系统：Windows 10

IDE：Android Studio 2.1.2

二、实验过程

1. 新建项目，添加 MainActivity.java，MusicService.java 和 activity_main.xml
2. 在 activity_main.xml 中设计音乐播放器的样式

```
<ImageView
    android:id="@+id/photo"
    android:layout_width="280dp"
    android:layout_height="280dp"
    android:layout_gravity="center"
    android:src="@mipmap/img" />

<TextView
    android:layout_marginLeft="4dp"
    android:id="@+id/state"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:text="↑"
    android:textSize="20dp" />
```

<LinearLayout

```
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_gravity="center"  
    android:layout_marginTop="20dp"  
    android:orientation="horizontal">
```

<TextView

```
    android:id="@+id/begin"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginRight="5dp"  
    android:text="00:00"  
    android:textSize="15dp" />
```

<SeekBar

```
    android:id="@+id/progress"  
    android:layout_width="260dp"  
    android:layout_height="wrap_content"  
    android:max="100"  
    android:progress="50" />
```

<TextView

```
    android:id="@+id/end"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginLeft="5dp"  
    android:text="04:00"  
    android:textSize="15dp" />
```

</LinearLayout>

<LinearLayout

```
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_gravity="center"  
    android:layout_marginTop="20dp"  
    android:orientation="horizontal">
```

<Button

```
    android:id="@+id/play_pause"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="PLAY" />
```

```

<Button
    android:id="@+id/stop"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="STOP" />

<Button
    android:id="@+id/quit"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="QUIT" />
</LinearLayout>

```

3. 在 Android Device Monitor 中向模拟器导入音乐文件

▼ data		2016-11-02	16:48	drwxrwx--x
K.Will-Melt.mp3	10097247	2016-11-02	04:29	-rw-rw-rw-
Nightwish - She Is My Sin.mp3	4663465	2016-06-14	13:06	-rw-rw-rw-
> adb		2016-09-07	09:16	drwx-----
> anr		2016-11-03	02:17	drwxrwxr-x
> app		2016-11-03	02:55	drwxrwxr-x

4. 实现 MusicService

实例化一个 MediaPlayer

```
public static MediaPlayer mp = new MediaPlayer();
```

在构造函数中载入音乐文件，并设置为循环播放

```

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

```

实现播放与暂停功能

```

public void play_pause() {
    if (mp.isPlaying())
        mp.pause();
    else
        mp.start();
}

```

实现停止功能

```

public void stop() {
    mp.stop();
    try {
        mp.prepare();
        mp.seekTo(0);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

重写 onCreate 函数和 onDestroy 函数

```

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

@Override
public void onDestroy() {
    super.onDestroy();
    if (mp != null) {
        mp.stop();
        mp.reset();
    }
}

```

新建一个继承 Binder 的 MyBinder 类，并实例化一个对象

```

public IBinder binder = new MyBinder();

public class MyBinder extends Binder {
    MusicService getService() {
        return MusicService.this;
    }
}

```

重写 onBind 函数

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

..

5. 实现 MainActivity

绑定 MusicService 服务

```
Intent intent = new Intent(this, MusicService.class);
bindService(intent, sc, BIND_AUTO_CREATE);
```

```
private ServiceConnection sc = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName componentName, IBinder iBinder) {
        musicService = ((MusicService.MyBinder) iBinder).getService();
    }

    @Override
    public void onServiceDisconnected(ComponentName componentName) {
        musicService = null;
    }
};
```

实现点击 PLAY/PAUSE 按钮播放或暂停音乐，并修改状态提示信息

```
play_pause.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if (play_pause.getText().equals("PLAY")) {
            musicService.play_pause();
            play_pause.setText("PAUSE");
            state.setText("PLAYING");
        } else {
            musicService.play_pause();
            play_pause.setText("PLAY");
            state.setText("PAUSED");
        }
    }
});
```

实现点击 STOP 按钮停止播放音乐

```
stop.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        musicService.stop();  
        play_pause.setText("PLAY");  
        state.setText("STOPPED");  
    }  
});
```

实现点击 QUIT 按钮停止播放音乐并退出程序

```
quit.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        mHandler.removeCallbacks(mRunnable);  
        unbindService(sc);  
        finish();  
    }  
});
```

使用 Handler 和 Runnable 实现图片旋转，进度条和播放时间的变化

```
private Handler mHandler = new Handler();  
private Runnable mRunnable = new Runnable() {  
    @Override  
    public void run() {  
        try {  
            if (MusicService.mp.isPlaying())  
                photo.setRotation(photo.getRotation() + 1);  
            progress.setProgress(MusicService.mp.getCurrentPosition());  
            progress.setMax(MusicService.mp.getDuration());  
            begin.setText(time.format(MusicService.mp.getCurrentPosition()));  
            end.setText(time.format(MusicService.mp.getDuration()));  
            mHandler.postDelayed(mRunnable, 50);  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
};
```

并在 onCreate 函数中启用该线程

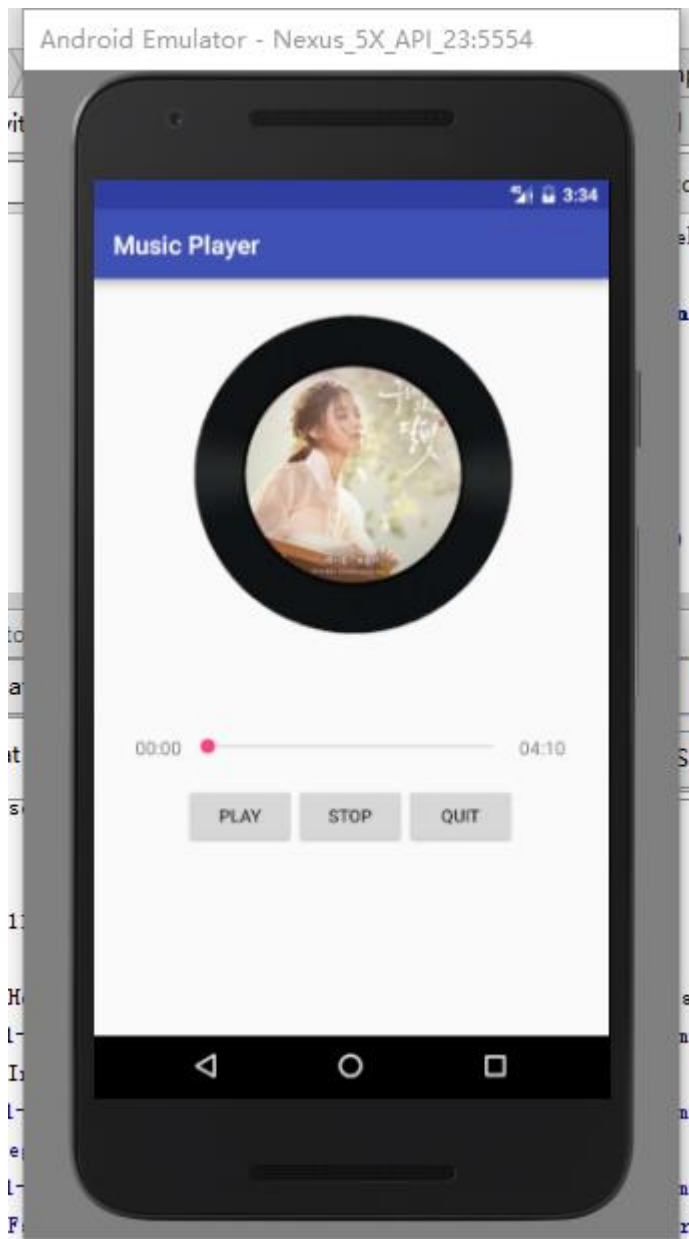
```
mHandle.post(mRunnable);
```

实现拖动进度条

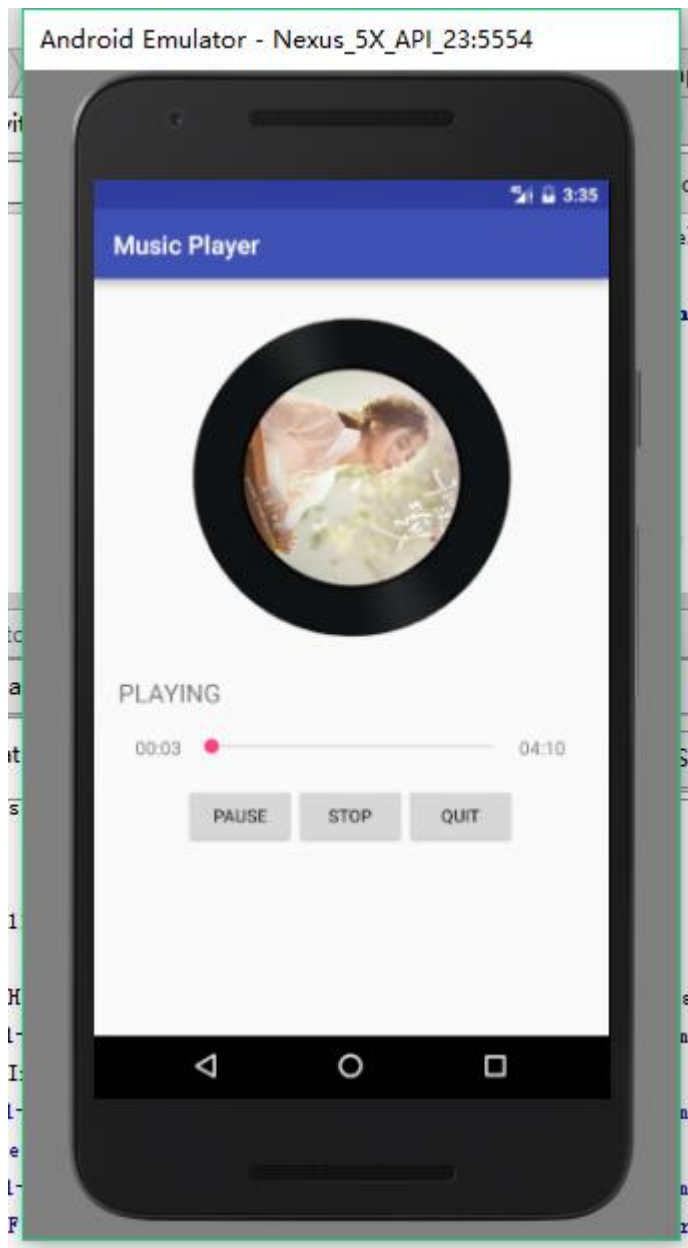
```
progress.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {  
    @Override  
    public void onProgressChanged(SeekBar seekBar, int i, boolean b) {  
        if (flag == true)  
            MusicService.mp.seekTo(i);  
    }  
  
    @Override  
    public void onStartTrackingTouch(SeekBar seekBar) {  
        flag = true;  
    }  
  
    @Override  
    public void onStopTrackingTouch(SeekBar seekBar) {  
        flag = false;  
    }  
});
```

三、实验结果

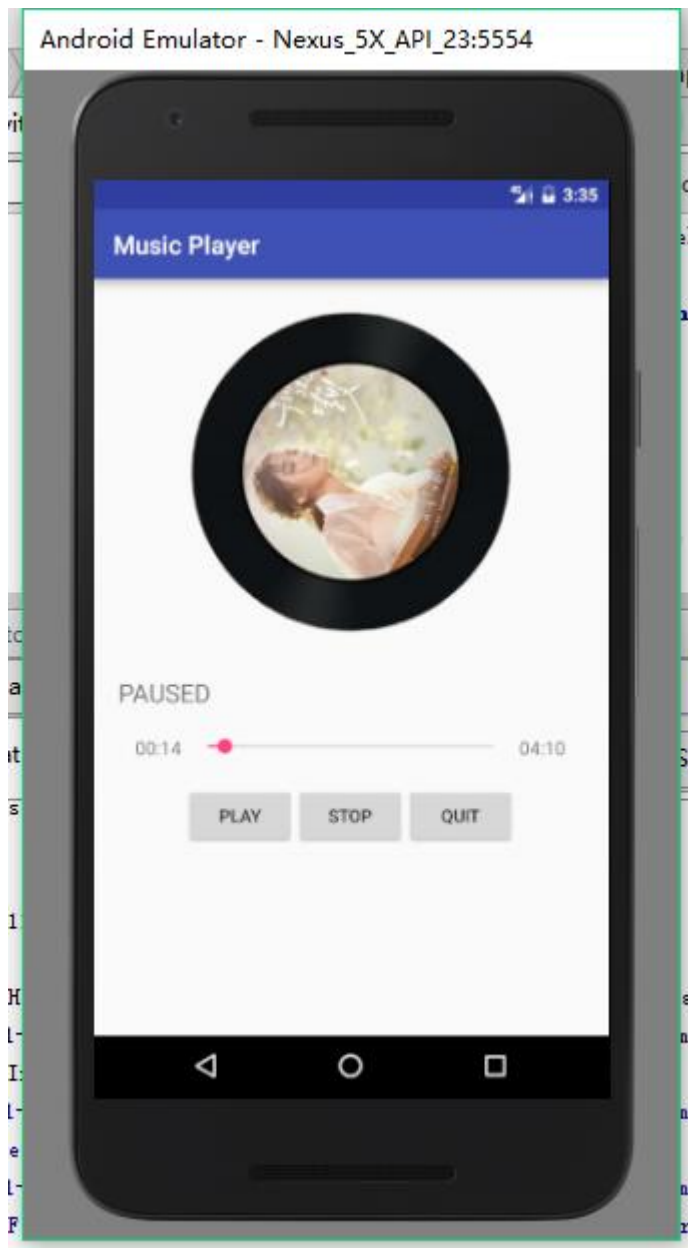
1. 运行程序



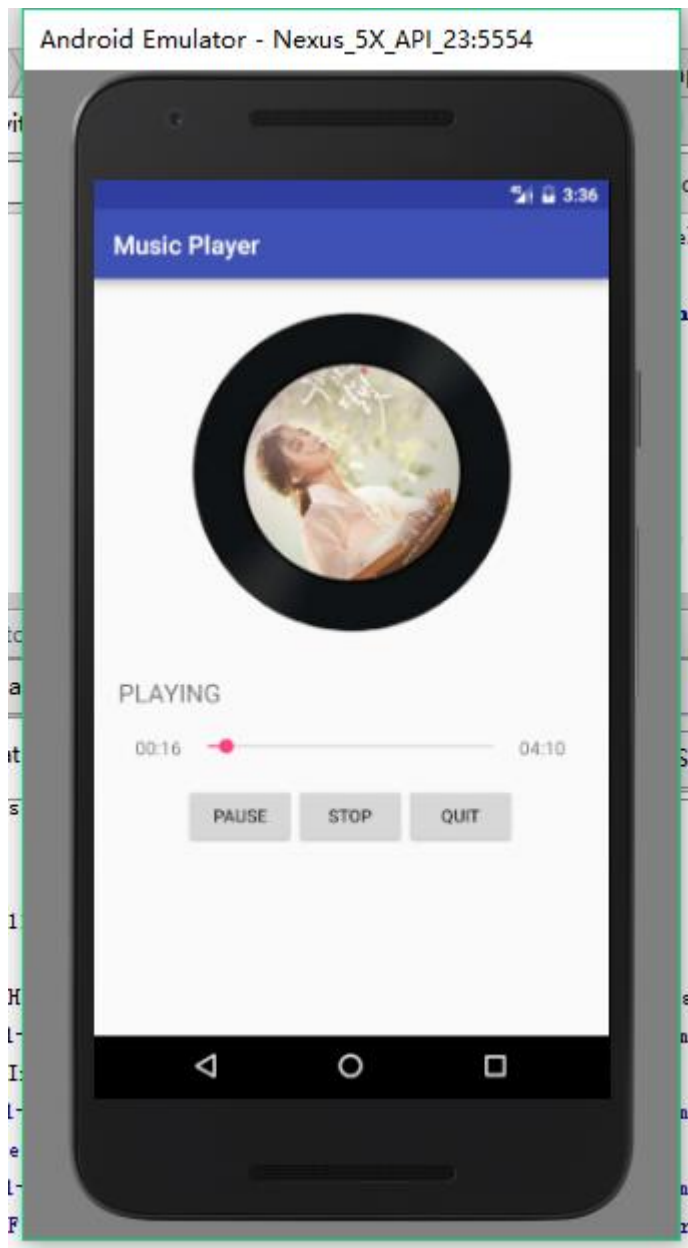
2. 点击 PLAY 按钮



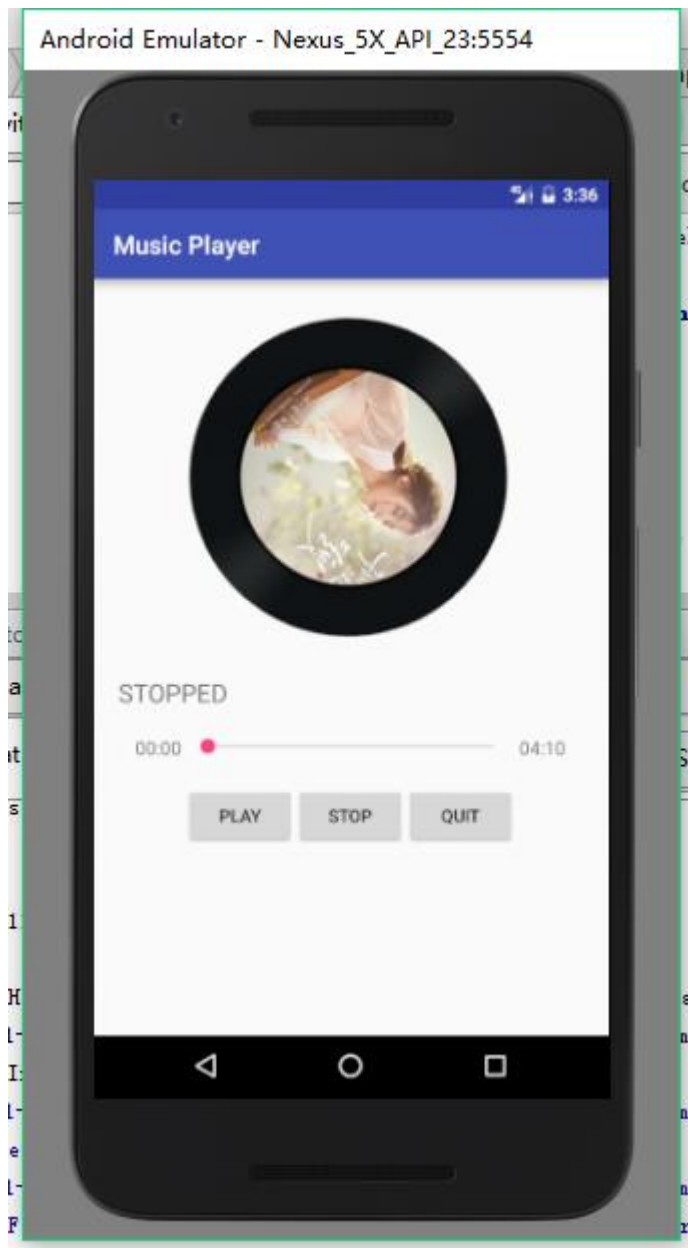
3. 点击 PAUSE 按钮



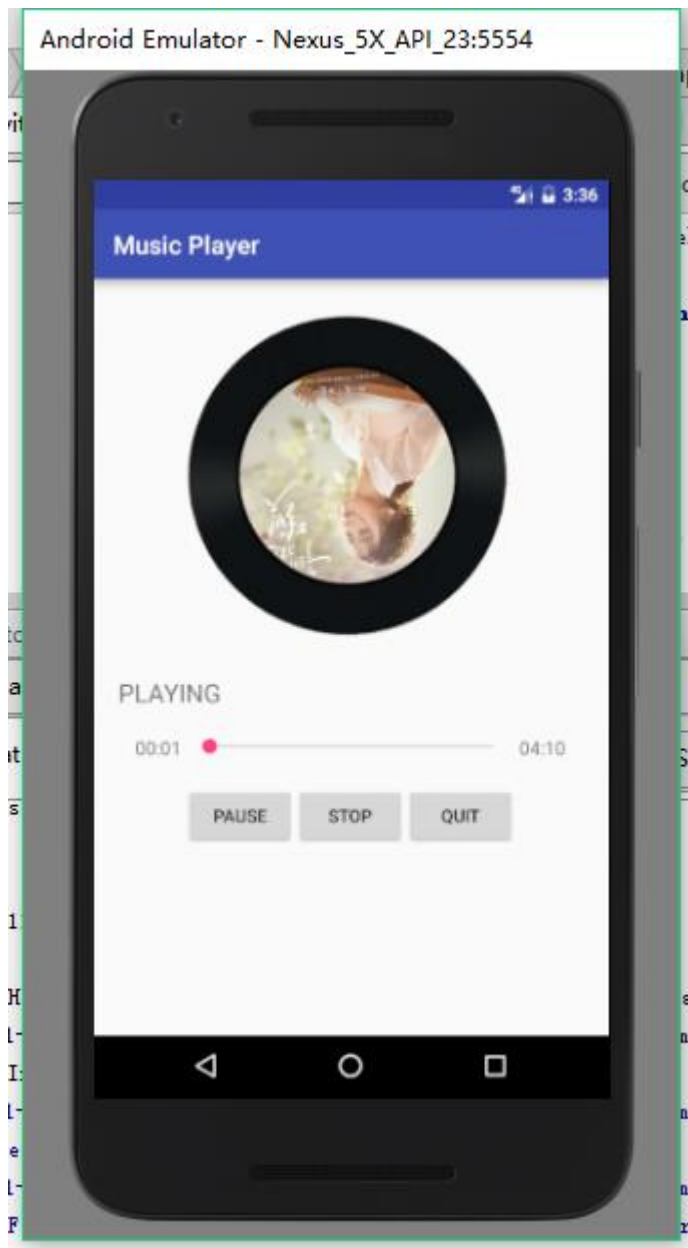
4. 再次点击 PLAY 按钮



5. 点击 STOP 按钮



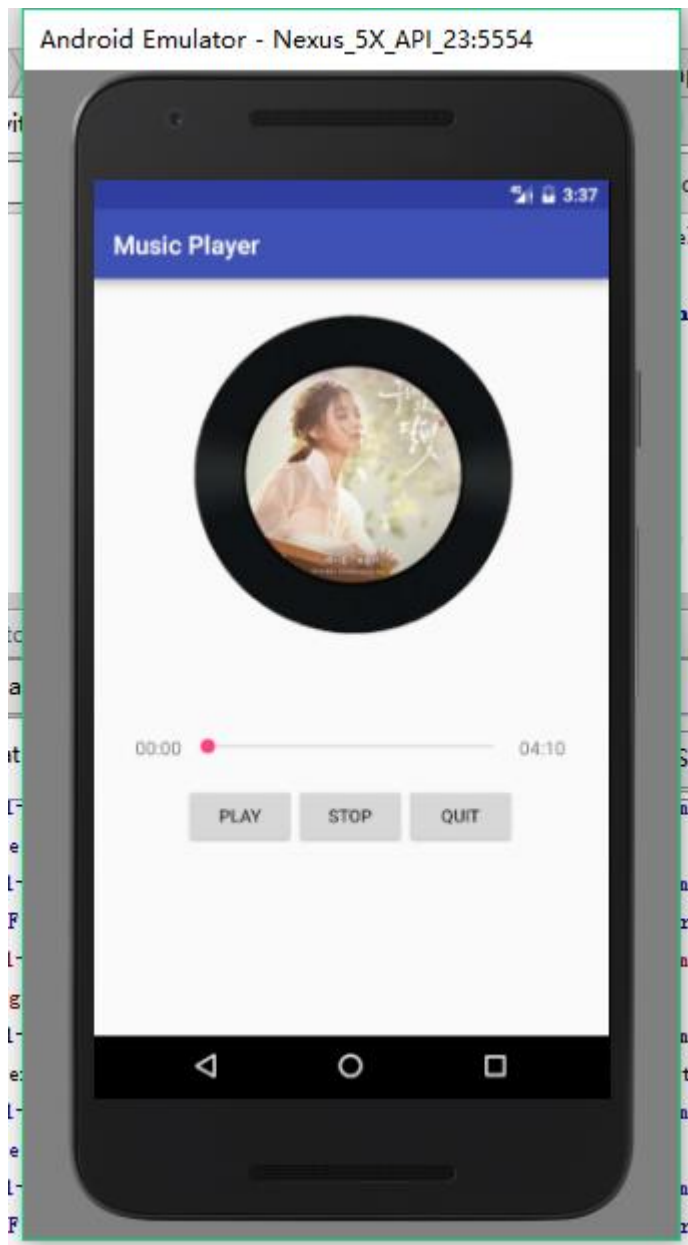
6. 再次点击 PLAY 按钮



7. 点击 QUIT 按钮



8. 在应用列表中找到 MusicPlayer 并打开



四、实验心得

1. Service 作为一种没有界面的后台活动，需要通过 binder 实现对其的操作，同样，Service 需要使用 Handler 来通知并修改 UI
2. 在 SeekBar 改变的事件监听器中，由于播放音乐时 SeekBar 也会自己变化，如果仅仅通过 SeekBar 的变化来定位此时音乐播放的时间点的话，播放的音乐会很不流畅。因此需要引入一个标识来标志当


前 SeekBar 的变化是人为的还是自己变化的，该标识在人为开始拖动时设置为 true，在结束拖动时设为 false，并在拖动时根据标识响应不同的操作

```
public void onProgressChanged(SeekBar seekBar, int i, boolean b) {  
    if (flag == true)  
        MusicService.mp.seekTo(i);  
}  
  
@Override  
public void onStartTrackingTouch(SeekBar seekBar) {  
    flag = true;  
}  
  
@Override  
public void onStopTrackingTouch(SeekBar seekBar) {  
    flag = false;  
}
```

3. 如果按照实验文档中使用 release()来实现 MusicService 的 onDestroy 函数，在 QUIT 之后通过点击应用列表中的图标再次打开 APP 的话会在判断播放状态时抛出 java.lang.IllegalStateException 异常，将 release()改成 reset()之后就可以了

4. 存在一个暂时无法解决的已知问题：在播放过程中点击 STOP，播放器会从音乐的开始部分再次播放 1s 左右才停止播放，DEMO 也存在该问题

使用的模拟器版本为：

	Nexus 5X API 23	1080 × 1920: 420dpi	23	Android 6.0 (Google APIs)
---	-----------------	---------------------	----	---------------------------