中山大學
SUN YAT-SEN UNIVERSITY

《手机平台应用开发（与 Google 共建）实验》

实验六：服务与多线程——简单音乐播放器

实验报告

学院名称：	数据科学与计算机学院
专　　业	软件工程（计应）
学生姓名：	张凯鑫
学　　号：	14331362
班　　级：	周三上午 4-5 节、周五下午 7-8 节

【实验目的】

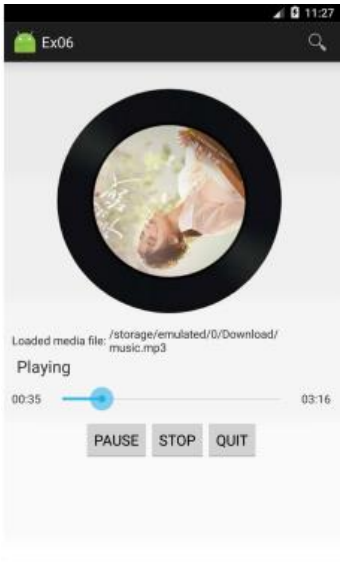
1. 学会使用 MediaPlayer；
2. 学会简单的多线程编程，使用 Handle 更新 UI；
3. 学会使用 Service 进行后台工作；
4. 学会使用 Service 与 Activity 进行通信。

【实验内容】

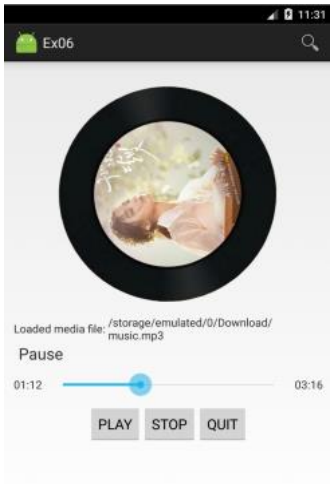
实现一个简单的播放器，要求功能有：

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、手动进度条改变进度功能；
4. 播放时图片旋转，显示当当前播放时间功能。

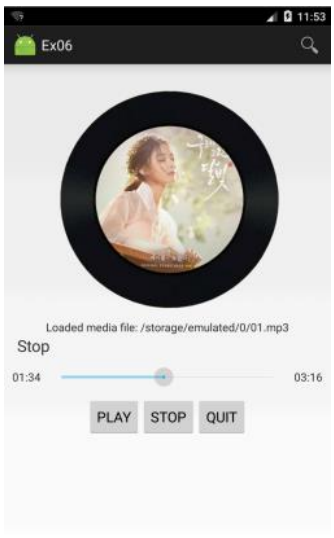
打开程序主页面



开始播放



暂停



停止

【实验过程】

1. 创建新新的 Android Studio 项目，命名为：ExperimentSix。
2. 在主页面的布局文件 activity_main.xml 中，按照界面要求添加控件：ImageView、TextView、SeekBar、Button 等，并按要求设置控件属性。
3. 在主页面的 java 文件 MainActivity.java 中，实现相对简单的控件的逻辑处理（详见实验结果部分）。
4. 创建 Service 文件 MusicService.java，在 AndroidManifest.xml 中注册 Service（创建后自动注册），使用 Service 与 Activity 进行通信。

AndroidManifest.xml：注册 Service

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

MainActivity.java：调用 bindService 保持与 Service 的通信

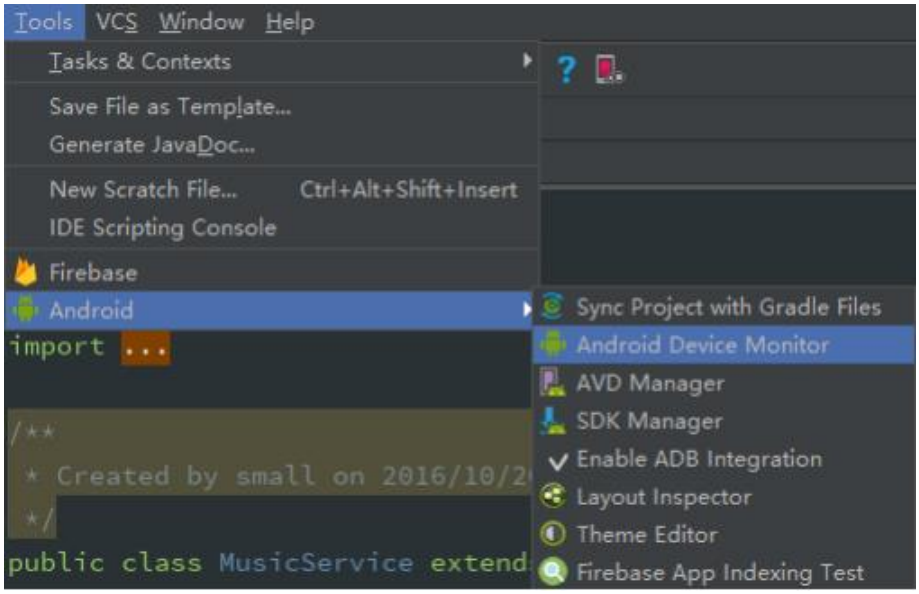
```
private MusicService ms;
private ServiceConnection sc = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        ms = ((MusicService.MyBinder)service).getService();
    }
    @Override
    public void onServiceDisconnected(ComponentName name) { ms = null; }
};
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

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

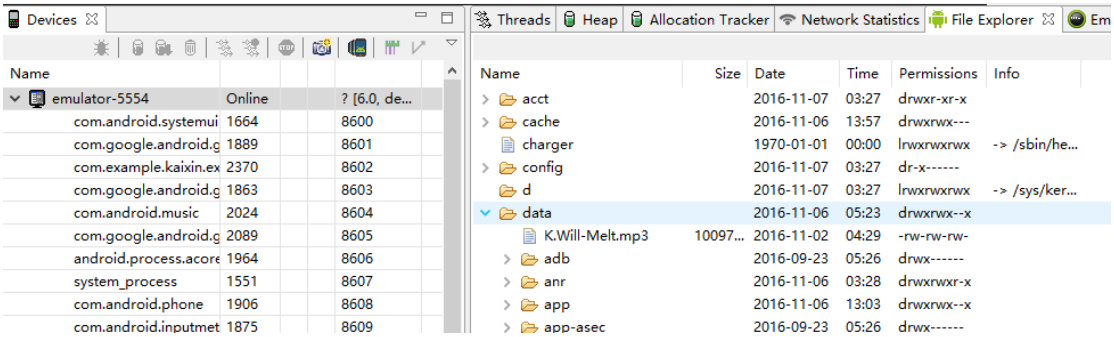
MusicService.java：通过 Binder 来保持 Activity 和 Service 的通信

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

5. 往虚拟机中添加文件，在 MusicService.java 中创建 MediaPlayer 类，进行后台工作。

添加 K.Will-Melt.mp3 到/data/目录下：





创建 MusicService 进行后台工作：

```
public MusicService() {
    try {
        mp.setDataSource("/data/K.Will-Melt.mp3");
        mp.prepare();
        mp.setLooping(true);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
@Override
public void onCreate() {
    super.onCreate();
}
@Override
public void onDestroy() {
    super.onDestroy();
    if (mp != null) {
        mp.stop();
        mp.release();
    }
}
```

6. 实现 ImageView 的旋转、SeekBar 进度条的变化，在 MainActivity.java 中使用简单的多线程编程，使用 Handle 更新 UI。

ImageView 旋转设置：

```
final ObjectAnimator animator = ObjectAnimator.ofFloat(cover, "rotation", 0f, 360, 0f);
animator.setDuration(36000);
animator.setInterpolator(new LinearInterpolator());
animator.setRepeatCount(-1);
animator.setRepeatMode(ValueAnimator.RESTART);
rotate = false;
```

SeekBar 进度条拖动处理：

```
seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {

    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {

    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
        int newPos = seekBar.getProgress();
        ms.mp.seekTo(newPos);
        cTime.setText(time.format(ms.mp.getCurrentPosition()));
    }
});
```

多线程编程及 Handle 更新 UI:

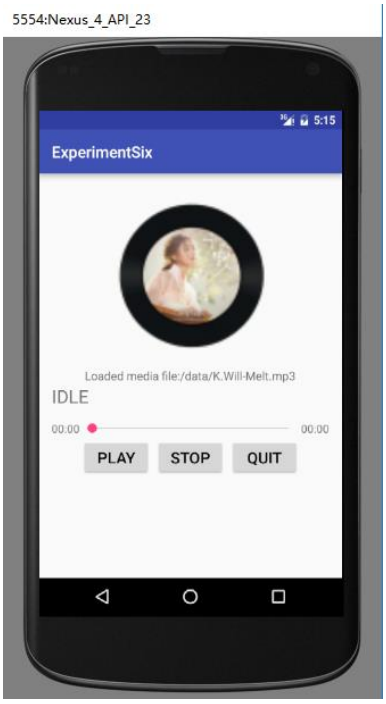
```
Handler mHandler = new Handler();
Runnable mRunnable = new Runnable() {
    @Override
    public void run() {
        while (seekBar.getProgress() < ms.mp.getDuration() - 1) {
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            mHandler.post(() -> {
                    seekBar.setProgress(ms.mp.getCurrentPosition());
                    cTime.setText(time.format(ms.mp.getCurrentPosition()));;
            });
        }
    }
};
```

7. 运行并调整实验代码直到实验完成。

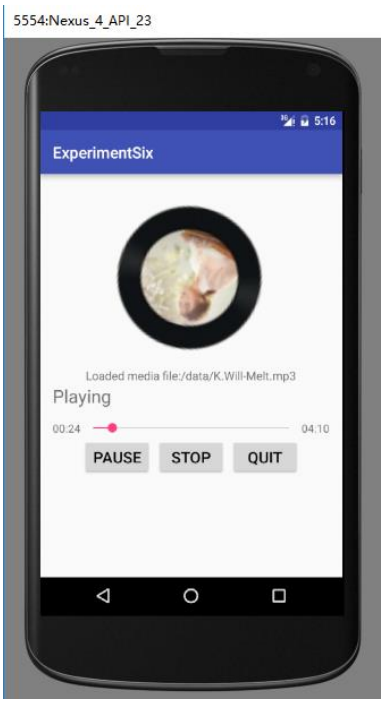
【实验结果】

实验效果图如下（实验代码详见 lab6_code 文件夹）：

程序主页面



开始播放

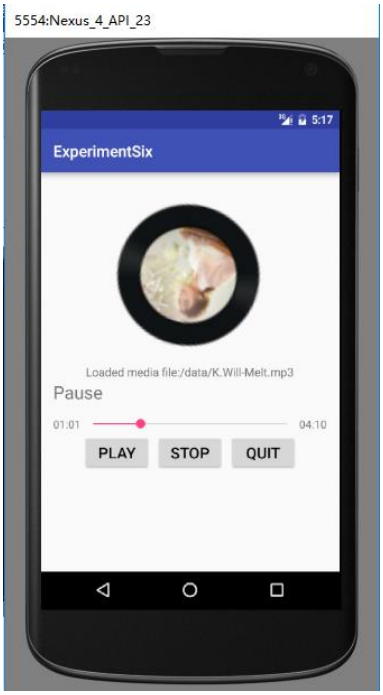


“PLAY/PAUSE”按键的点击事件处理：通过调用 Service 中的方法 play()来实现后台工作

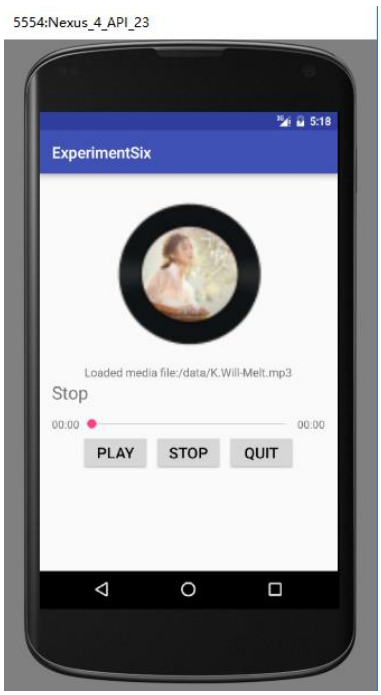
```
btn_play.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (ms != null) {
            ms.play();
            tTime.setText(time.format(ms.mp.getDuration()));
            seekBar.setMax(ms.mp.getDuration());
            if (ms.mp.isPlaying()) {
                btn_play.setText("PAUSE");
                state.setText("Playing");
                if (animator.isPaused()) {
                    animator.resume();
                } else {
                    animator.start();
                }
                mThread = new Thread(mRunnable);
                mThread.start();
            } else {
                btn_play.setText("PLAY");
                state.setText("Pause");
                animator.pause();
                mThread = null;
            }
        }
    }
```

```
public void play() {
    if (mp != null) {
        if (mp.isPlaying()) {
            mp.pause();
        } else {
            mp.start();
        }
    } else {
        try {
            mp.setDataSource("/data/K.Will-Melt.mp3");
            mp.prepare();
            mp.setLooping(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

暂停



停止



“STOP”按键的点击事件处理：通过调用 Service 中的方法 stop()来实现后台停止播放

```
btn_stop.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (ms != null) {
            ms.stop();
            rotate = false;
            state.setText("Stop");
            btn_play.setText("PLAY");
            tTime.setText("00:00");
            cTime.setText("00:00");
            animator.end();
            mThread = null;
        }
    }
});
```

```
public void stop() {
    if (mp != null) {
        mp.stop();
        try {
            mp.reset();
            mp.setDataSource("/data/K.Will-Melt.mp3");
            mp.prepare();
            mp.seekTo(0);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

“QUIT”按键的点击事件处理：停止服务，解除绑定

```
btn_quit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mHandler.removeCallbacks(mRunnable);
        unbindService(sc);
        try {
            MainActivity.this.finish();
            System.exit(0);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});
```

【遇到的问题与解决方法】

1. 图片的旋转问题：起初使用一个单独的 xml 为 ImageView 设置旋转属性，能够实现图片的旋转、停止，但一直无法实现图片的暂停，即保留所停的位置，恢复旋转时从该位置继续旋转。之后在 MainActivity.java 中设置 ImageView 的的旋转属性，通过 ObjectAnimator 进行设置，该类可以使用 start()、pause()、resume()、end()等方法来实现旋转的开始、暂停、恢复、结束等，使用起来更加方便。
2. 向虚拟机添加文件后路径获取问题：实验文档中关于往虚拟机中添加文件后，获取文件路径的相关知识点不够完整，个人在理解时出错，最初要获得路径时，使用实验文档中的：`Environment.getExternalStorageDirectory() + "/data/K.Will-Melt.mp3`，但总是无法获得文件，以为是自己未能将文件放入到模拟器中，但模拟器中的 sdcard 并不能打开，纠结了很久，查了很久资料，最后使用如下路径成功取得所添加的文件：`"/data/K.Will-Melt.mp3”`
3. 进度条问题：MediaPlayer.getDuration()这个方法获得的是音频的总时长，单位为毫秒，实验时以为是秒，使得进度条移动幅度太大。进度条起初不会移动，后来意识到应该是线程中实现，于是转为 Thread、Handle、Runnable 的问题，通过翻看实验文档和理论课课件，成功解决，至此实现基本完成。
4. 停止播放问题：点击 STOP 按键停止播放后，播放器进度条归零，时间归零，图片停止旋转并回到最初的位置，但还会重头播放一两秒的音乐才停止，一直处理不好这个问题，最后不得已在 stop()和 prepare()之间调用了 reset()方法，再重新获取资源。

【实验心得与体会】

1. 本次实验主要是使用 MediaPlayer、Thread、Handle、Service 等知识点来实现，实验中还涉及到了 ImageView 的图片旋转、SeekBar 进度条等陌生知识点，总得来说难度比较大，但只要思路清晰，实现起来还是能够接受的。
2. MediaPlayer 比较简单，只需要清楚各个方法的功能及能使用的时间即可，使用 MediaPlayer 类可以很快的实现简单播放器，主要用到的方法有：setDataSource()、start()、stop()、pause()、release()。
3. 实现后台播放功能考查的是 Service 的知识点，Service 创建后会自动在 AndroidManifest.xml 文件中注册，使用时一般难点在 Service 与 Activity 之间的通信，在 Service 类中通过 Binder 来保持通信，在 Activity 中调用 bindService 保持通信，而且要声明 ServiceConnection。
4. Handle 的使用涉及多线程编程问题，一般重写 Runnable 中的 run()方法来实现多线程，它是一个循环，不断地调用 Handle 的 post(Runnable())来更新 UI。
5. 图片的旋转属性可以单独用一个 xml 附加给 ImageView，通过这种方法实现的旋转属性一般功能比较单一，难以更改属性。个人觉得更好地是直接在 java 文件中通过 ObjectAnimator 类来给图片的控件添加属性，可以即时地更改属性的值和使用类的方法。