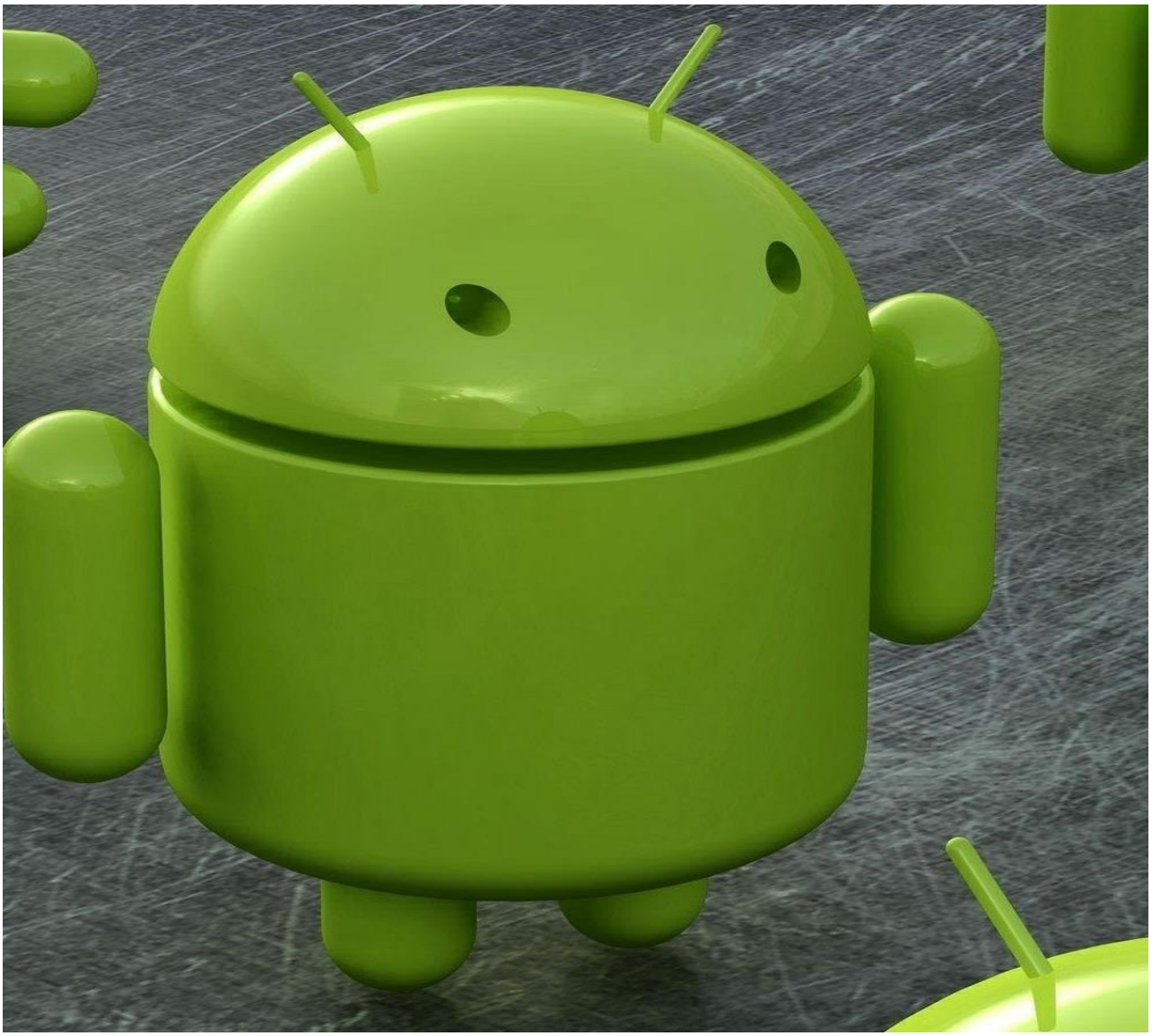


实验六

服务与多线程--简单音乐播放器

【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】

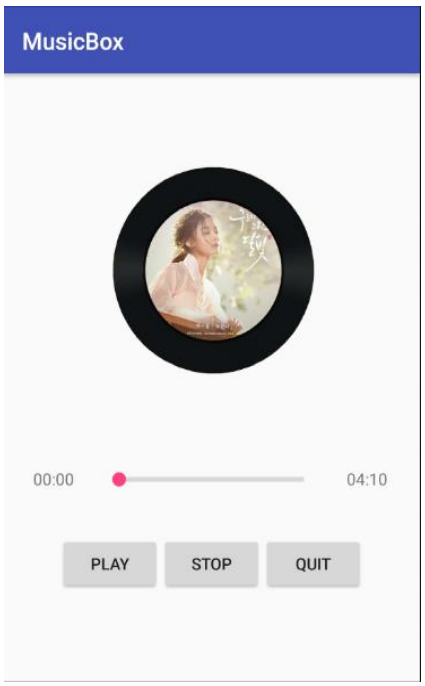
实现一个简单的播放器，要求功能有：

1. 播放、暂停，停止，退出功能；

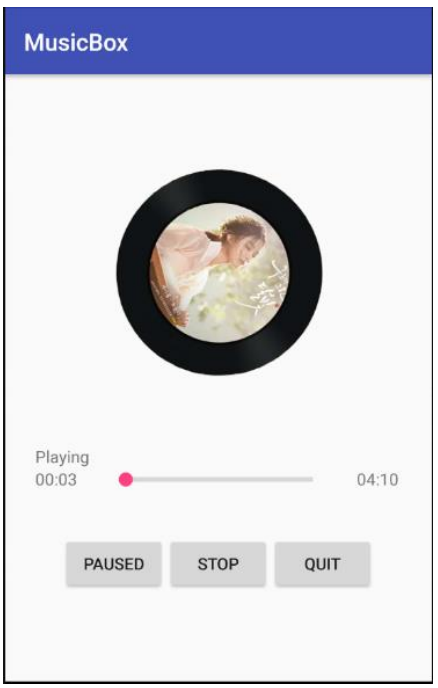
2. 后台播放功能；

3. 进度条显示播放进度、拖动进度条改变进度功能；

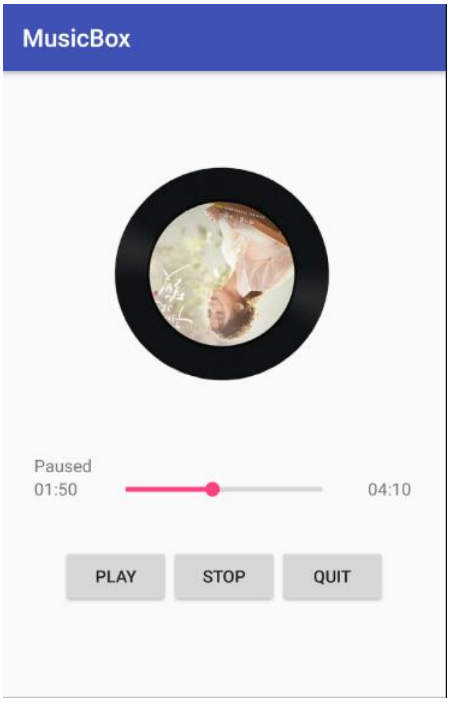
4. 播放时图片旋转，显示当前播放时间功能；



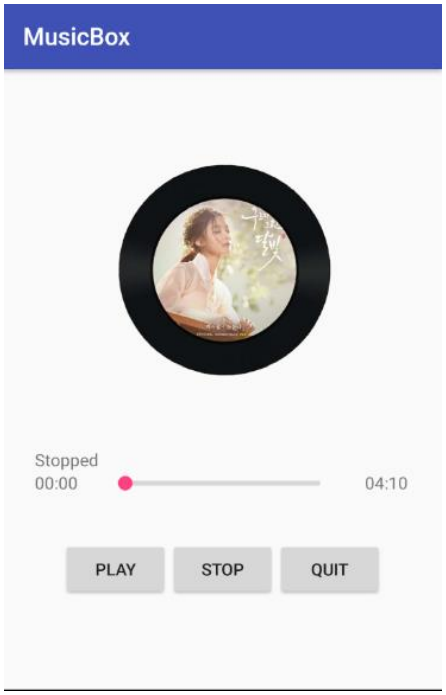
打开程序主页面



开始播放



暂停

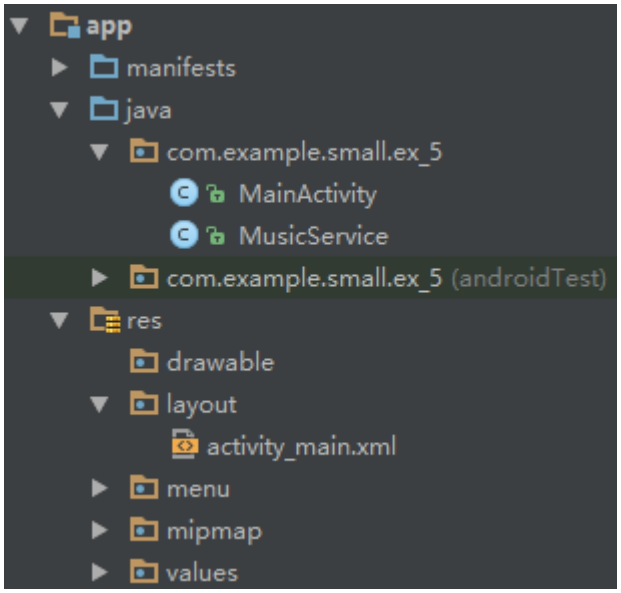


停止

【参考内容】

注：以下所有代码仅供参考，请根据实际需要适当修改或者添加。

1. 参考文件目录如下：



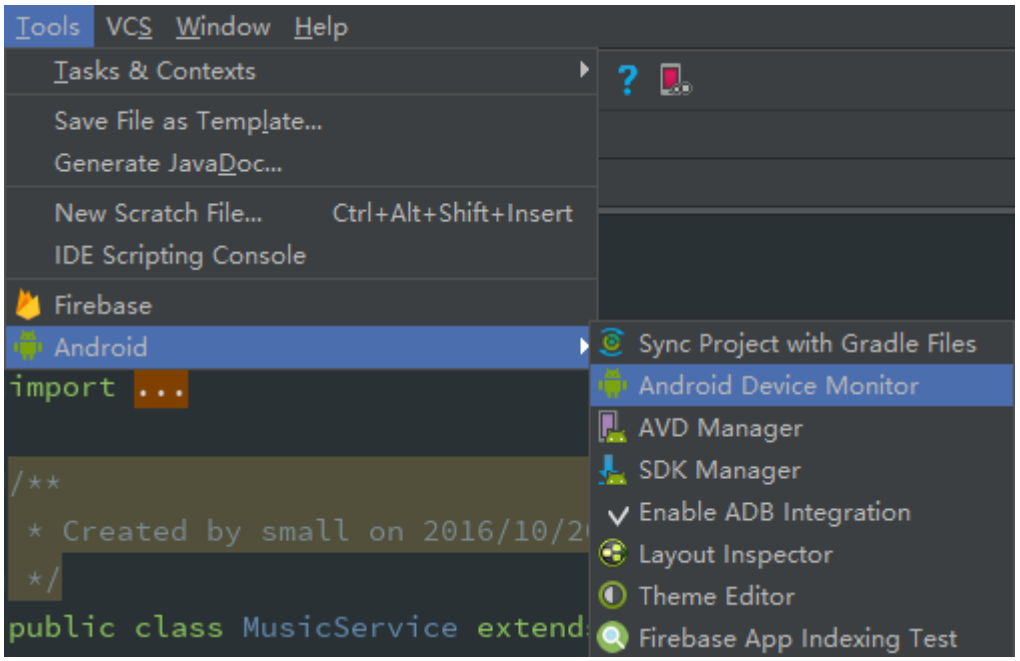
2. MediaPlayer 介绍

常用方法：

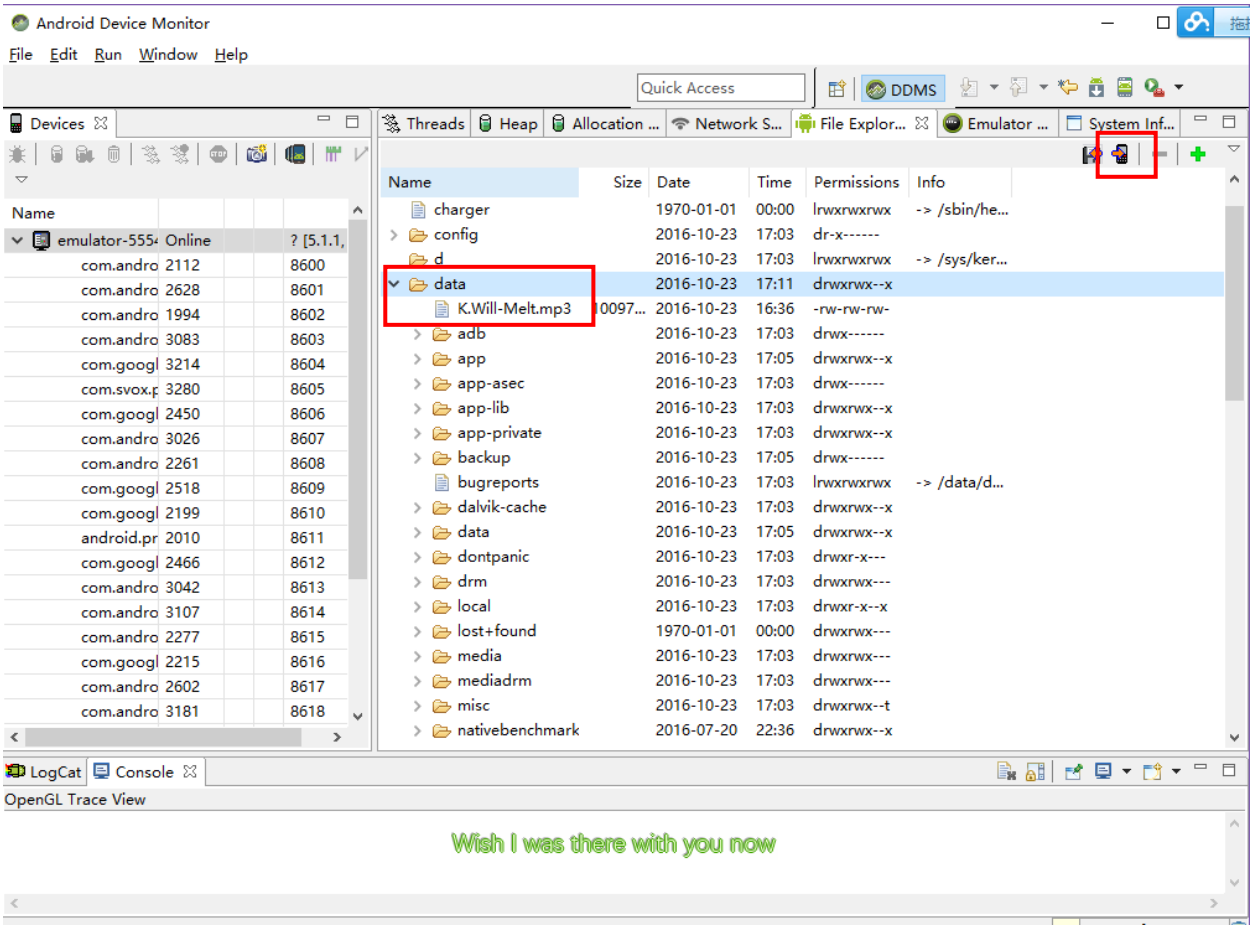
函数	功能	使用时机
setDataSource(String)	设置音频文件路径 进入初始化状态	MediaPlayer 对象已创建
prepare()	进入就绪状态	已初始化或停止
start()	进入播放状态	已就绪
pause()	进入暂停状态	正在播放
stop()	进入停止状态	正在播放或暂停
isPlaying()	检查是否正在播放	任意正常状态
getCurrentPosition()	获取当前已播放的毫秒数	已就绪
getDuration()	获取文件的时间长度(毫秒)	已就绪
release()	停止播放并释放资源	任何时候

3. 向虚拟机添加文件

首先打开 Android Device Monitor，如下图：



然后打开 file explorer 选择 data 文件夹点击右上角的导入文件，将音乐文件导入进去



注意不要导入有中文名的文件，要导入有中文名的可以使用 UltraISO 试试。

Ps：安卓6.0以上系统可能会有系统文件夹权限保护，所以如果有同学碰到无法加入虚拟机的问题，可使用自己手机进行调试，注意下把文件拷到内置 SD 卡而不是外置 SD 卡会比较方便。
文件拷贝到存储系统的方法也很简单，可以使用电脑端手机管家软件进行文件管理，也可直接在手机端进行文件管理，把文件放在一个SD卡上的目录内，只需要根据目录地址进行索引即可。
要使用外置的 SD 卡时，注意下文件路径的获取。这是相关的路径获取方法:http://blog.sina.com.cn/s/blog_5da93c8f0102vcam.html

4. 使用 MediaPlayer

创建对象：初始化：
注意下获取的文件路径，若是使用模拟器的如下，若是使用自己手机的内置 SD 卡则使用：`Environment.getExternalStorageDirectory() + "/K.Will-Melt.mp3"`，斜杠之后是内置SD卡的相对地址。

```
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();
    }
}
```

播放/暂停：

```
if (mp.isPlaying()) {
    mp.pause();
} else {
    mp.start();
}
```

停止：

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

5. Service 的使用

创建 service 类，实现 MediaPlayer 的功能。

注意在 AndroidManifest.xml 文件里注册 Service：

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

通过 Binder 来保持 Activity 和 Service 的通信(写在 service 类)：

```
public class MyBinder extends Binder {
    @Override
    protected boolean onTransact(int code, Parcel data, Parcel reply, int flags) throws RemoteException {
        switch (code)
        {
            case 101:
                // 播放按钮，服务处理函数
                break;
            case 102:
                // 停止按钮，服务处理函数
                break;
            case 103:
                // 退出按钮，服务处理函数
                break;
            case 104:
                // 界面刷新，服务返回数据函数
                break;
            case 105:
                // 拖动进度条，服务处理函数
                break;
        }
        return super.onTransact(code,data,reply,flags);
    }
}
```

举个例子：在onTransact函数中，如果想要回传一个数据给activity，处理函数可以这样写：

```
int i = getPosition();
reply.writeInt(i);
```

在Activity中通过IBinder与service通信（写在 activity 类）：

```
try {
    int code = 101;
    Parcel data = Parcel.obtain();
    Parcel reply = Parcel.obtain();
    mBinder.transact(code, data, reply, 0);
} catch (RemoteException e) {
    e.printStackTrace();
}
```

在 Activity 中调用 bindService 保持与 Service 的通信(写在 activity 类)：
Activity 启动时绑定 Service：

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

bindService 成功后回调 onServiceConnected 函数，通过 IBinder 获取 Service 对象，实现 Activity 与 Service 的绑定(写在 activity 类)：

```
private ServiceConnection sc;
```

```
sc = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        Log.d("service", "connected");
        mBinder = service;
    }
    @Override
    public void onServiceDisconnected(ComponentName name) { sc = null; }
};
```

停止服务时，必须解除绑定，写入退出按钮中(写在 activity 类)：

```
unbindService(sc);
sc = null;
try {
    MainActivity.this.finish();
    System.exit(0);
} catch (Exception e) {
    e.printStackTrace();
}
```

此时，在 Activity 的 onCreate 方法中执行上述与 Service 通信的方法后，即可实现后台播放。点击退出按钮，程序会退出，音乐停止；返回桌面，音乐继续播放。

6. Handler 的使用

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。使用 Seekbar 显示播放进度，设置当前值与最大值：

```
seekBar.setProgress(ms.mp.getCurrentPosition());
seekBar.setMax(ms.mp.getDuration());
```

定义 Handler：run 函数中进行更新 seekbar 的进度在类中定义简单日期格式，用来显示播放的时间，用 time.format 来格式所需要的数据，用来监听进度条的滑动变化：

```
private SimpleDateFormat time = new SimpleDateFormat("mm:ss");
```

```
seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
```

定义一个新线程：

```
Thread mThread = new Thread() {
    @Override
    public void run() {
        while (true) {
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            if ( sc != null && hasPermission == true)
                mHandler.obtainMessage(123).sendToTarget();
        }
    }
};
mThread.start();
```

Hander处理：

```
final Handler mHandler = new Handler() {
    @Override
    public void handleMessage(Message msg) {
        super.handleMessage(msg);
        switch (msg.what) {
            case 123:
                // UI 更新相关内容
                break;
        }
    }
};
```

7. 动态文件读取权限申请

对于安卓6.0以上机型，需要动态获取文件阅读权限，征得用户权限认可，这里直接给同学们提供代码，对于使用内置SD卡进行测试的同学可能会有用。

首先在 AndroidManifest.xml 文件里注册权限：

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
```

然后在代码中添加权限确认函数：

```
public static void verifyStoragePermissions(Activity activity) {
    try {
        //检测是否有读取的权限
        int permission = ActivityCompat.checkSelfPermission(activity,
                "android.permission.READ_EXTERNAL_STORAGE");
        if (permission != PackageManager.PERMISSION_GRANTED) {
            // 没有读取的权限，去申请读取的权限，会弹出对话框
            ActivityCompat.requestPermissions(activity, PERMISSIONS_STORAGE, REQUEST_EXTERNAL_STORAGE);
        }
        else {
            hasPermission = true;
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

请求权限会弹出询问框，用户选择后，系统会调用如下回调函数：

```
@Override
public void onRequestPermissionsResult(int requestCode, String permissions[], int[] grantResults) {
    if (grantResults.length > 0
            && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
        // 权限被用户同意，可以做你要做的事情了。
    } else {
        // 权限被用户拒绝了，可以提示用户,关闭界面等等。
        System.exit(0);
    }
    return;
}
```

【检查内容】

1. 布局显示是否正常
2. 播放，暂停，停止功能是否可用，界面显示是否正常
3. 本次实验要求使用Ibinder方式实现服务与Activity之间通信
4. 是否可以后台播放
5. 播放时是否显示当前播放时间，位置，以及图片是否旋转(可用ObjectAnimator类)
6. 是否可以拖动滑动条进行音乐进度调节
7. 加分项：通过申请动态权限读取内置sd卡中音乐文件

【友情提示】

由于本次实验进行两周，可能会用到这两周理论课上的知识，如果服务部分还没有教，现在给大家两种实现方案供大家选择：

1. 第一周可以先实现【实验目的】的第1、2点，将MediaPlayer的相关操作放在Activity里面，然后实现多线程对UI的更新。第二周教完服务之后，再把MediaPlayer搬到服务里面，然后实现Activity和Service之间的通信，完成本次实验。
2. 大家可根据课程ppt先自学，然后按照本文档的过程进行相关的实现，直接一步到位完成本次实验。

为给大家减少压力，本次实验不设置递进式实验过程，最终只需完成一份代码和一份实验报告即可，上述两种方案仅供大家根据理论教学进度和自学水平自行选择。

【提交说明】

1、deadline：本次实验进行两周，下两次实验课前一天晚上 12 点

2、提交作业地址：ftp://edin.sysu.edu.cn

3、文件命名及格式要求：学号_姓名_labX.zip（姓名中文拼音均可）

4、目录结构：

```
14331111_huashen_lab1 --
                       |
                       --  lab1实验报告.pdf
                       |
                       --  lab1_code（包含项目代码文件）
```

其中项目代码文件为项目文件夹，*提交之前先 clean