项目 1　在线音乐播放器

1. 项目需求

从本项目开始，我们一起开发一个简单的在线音乐播放器，它具有以下功能：

（1）程序启动时显示一个启动画面，如图 1-1 所示。

（2）在程序主页面显示服务器上的歌曲列表，包括歌手名与歌曲名，单击列表条目可实现在线音乐播放，单击“播放”、“暂停”、“上一曲”和“下一曲”按钮可实现对应的功能，进度条可以显示歌曲播放进度，如图 1-2 所示。

图 1-1　程序启动画面

图 1-2　程序主页面

2. 项目设计思路

在线音乐播放器的开发包括客户端的开发与服务端的开发。音乐资源存放在服务器上。客户端向服务端发送 URL 请求，服务端将歌手名、歌曲名、MP3 链接信息封装为 JSON 数组，返回给客户端，如图 1-3 所示。下面对服务端与客户端分别进行设计。

要满足项目需求，服务端需要按以下思路设计：

（1）在 Apache 服务器上合理设计本项目的目录结构，存放音乐资源文件及音乐信息索

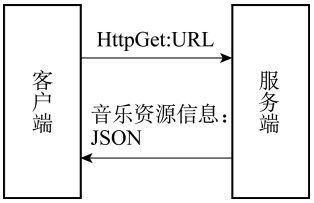

图 1-3　在线音乐播放器开发框架

引文件。

（2）设计服务端与客户端的信息交互接口，包括访问URL、访问参数、返回值。

（3）设计音乐信息索引文件，按照 JSON 格式存放音乐信息，可以方便地将音乐信息返回给客户端。

要满足项目需求，客户端设计结构如图 1-4 所示，相应的设计思路如下：

（1）设计一个程序启动动画视图类 SplashActivity，用于显示一个启动动画，显示 App 名称、版本信息等，动画播放完成后，跳转到主视图。

（2）设计一个主视图类 MainActivity，作为提供给用户的操作界面，控制音乐的播放。

（3）设计一个 HTTP 访问功能类 HttpAsyncRequestTask，负责客户端与服务端信息的交互。

（4）设计一个音乐信息封装类 Music，用于封装音乐信息，如歌手名、音乐名等，方便在不同的类之间传递音乐信息。

（5）设计一个音乐播放服务类 MusicService，用于在后台播放音乐。MainActivity 通过 HttpAsyncRequestTask 从服务器获取到音乐信息，再通过 Music 封装后，传递给 MusicService，从而实现音乐的播放。

（6）设计一个音乐播放工具类 MusicUtil，用于实现音乐播放的具体操作，如播放、暂停、上一曲、下一曲等。MusicService 通过调用 MusicUtil 中的方法，实现音乐播放的具体操作。

（7）设计一个音乐列表显示适配器类 MusicListViewMainAdapter，用于音乐列表显示。MainActivity 通过 HttpAsyncRequestTask 从服务器获取到音乐信息，再通过 Music 封装后，传递给 MusicListViewMainAdapter，从而实现音乐列表的显示。

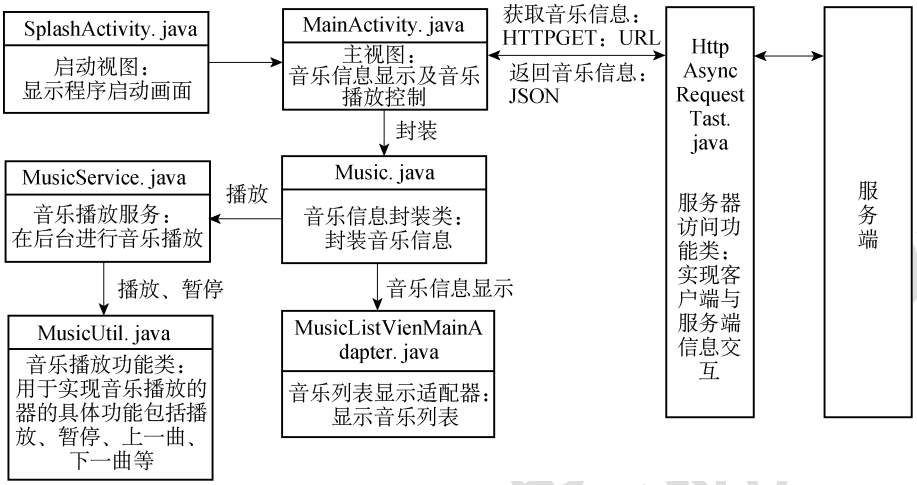

图 1-4　客户端类结构

按照以上设计思路，我们把本项目的开发划分为两个任务：任务 1-1 服务端开发；任务 1-2 客户端开发。下面我们对两个子任务分别进行描述。

任务 1-1　服务端开发

1.1.1　服务端目录结构设置

在 XAMPP 的安装文件夹中找到 htdocs 文件夹，在该文件夹中新建一个 music 文件夹，用于存放音乐资源。在 music 文件夹中新建一个文本文件，命名为 music.json，用于存放歌手名、歌曲名、音乐文件存放路径信息。服务端文件结构，如图 1-5 所示。

图 1-5　服务端文件结构

1.1.2　接口设计

访问 URL：http://localhost/music/music.json。

访问参数：无

返回值（json）：见表 1-1。

表 1-1　音乐查询 JSON 接口服务器响应参数

参　数	类　型	说　明
name	String	歌曲名
singer	String	歌手名
mp3	String	音乐文件存放路径

1.1.3 服务端代码编写

用记事本打开 music.json，建立一个 JSON 数组，存放音乐资源信息，代码如下：

D:\xampp\htdocs\music\music.json

```
[{"name":"时间都去哪了","singer":"王铮亮","mp3":"music/101.mp3"},
{"name":"太多","singer":"陈绮贞","mp3":"music/102.mp3"},
{"name":"你我","singer":"陈妍希","mp3":"music/103.mp3"},
{"name":"十年","singer":"陈奕迅","mp3":"music/104.mp3"},
{"name":"待我长发及腰","singer":"尚雯婕","mp3":"music/105.mp3"}]
```

注意：为保证程序的稳定性，需要将 music.json 保存为 UTF-8 格式。选择“另存为”，然后将编码格式更改为 UTF-8。

1.1.4 测试

在 XAMPP 安装文件夹中双击 xampp-control.exe，打开 XAMPP 控制台。在控制台中单击 Apache 服务器对应的“Start”按钮，开启 Apache 服务器，如图 1-6 所示。

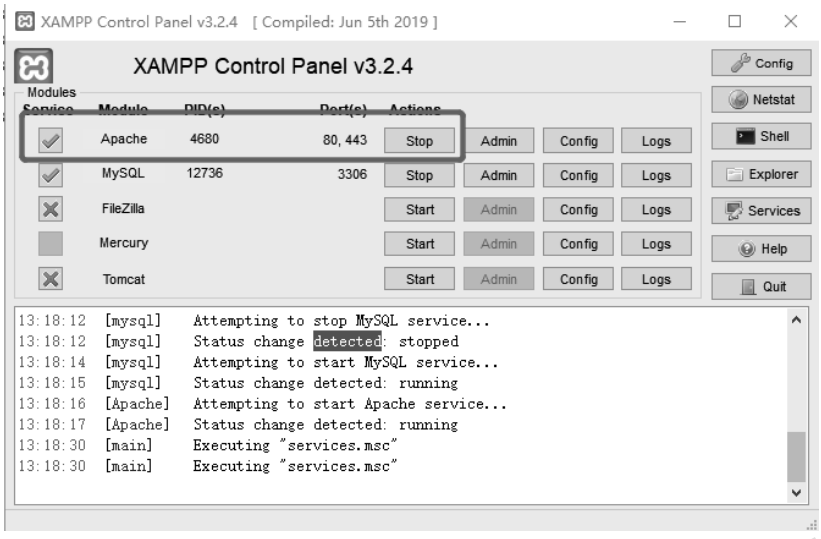

图 1-6 开启 Apache 服务器

打开浏览器，在地址栏中输入“http://localhost/music/music.json”，如果能够正常打开页面，则测试成功，如图 1-7 所示。

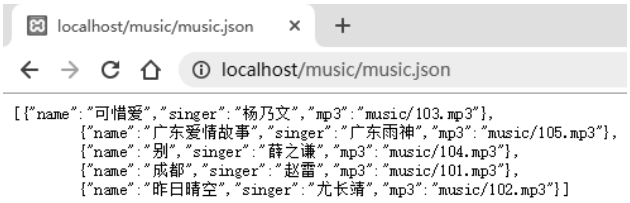

图 1-7　测试成功

注：如果希望使用 Tomcat 服务器，则需将上述开发的服务端（music 文件夹）复制到“D:\xampp\tomcat\webapps\”中，在 XAMPP 控制台中关闭 Apache 服务器，开启 Tomcat 服务器，然后在浏览器中输入“http://localhost:8080/music/music.json”进行测试。

任务 1-2　客户端开发

1.2.1　客户端文件结构

在线音乐播放器客户端为 Android 客户端，主要实现的功能包括显示程序启动画面、显示服务器音乐资源信息、在线播放音乐。客户端主要描述分为启动界面和主界面两个模块。

- 启动界面：显示程序启动动画。
- 主界面：显示服务器音乐资源信息及在线播放音乐。

客户端文件结构如图 1-8 所示，源文件共分 5 个包。

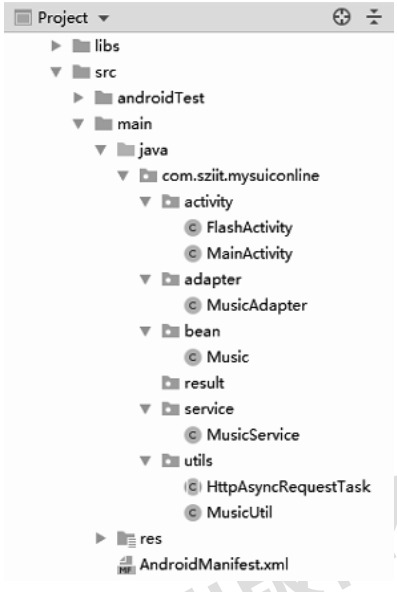

图 1-8　客户端文件结构

1.2.2　客户端开发

1. 启动界面模块开发

（1）新建工程

在 Android Studio 中新建一个 Android 工程，选择“Empty Activity”布局，单击“Next”按钮，应用程序名为 MyMusicOnline，包名为 com.sziit.mymusiconline，如图 1-9 所示。在所有页面中选择默认选项，在最后的页面中单击“Finish”按钮。

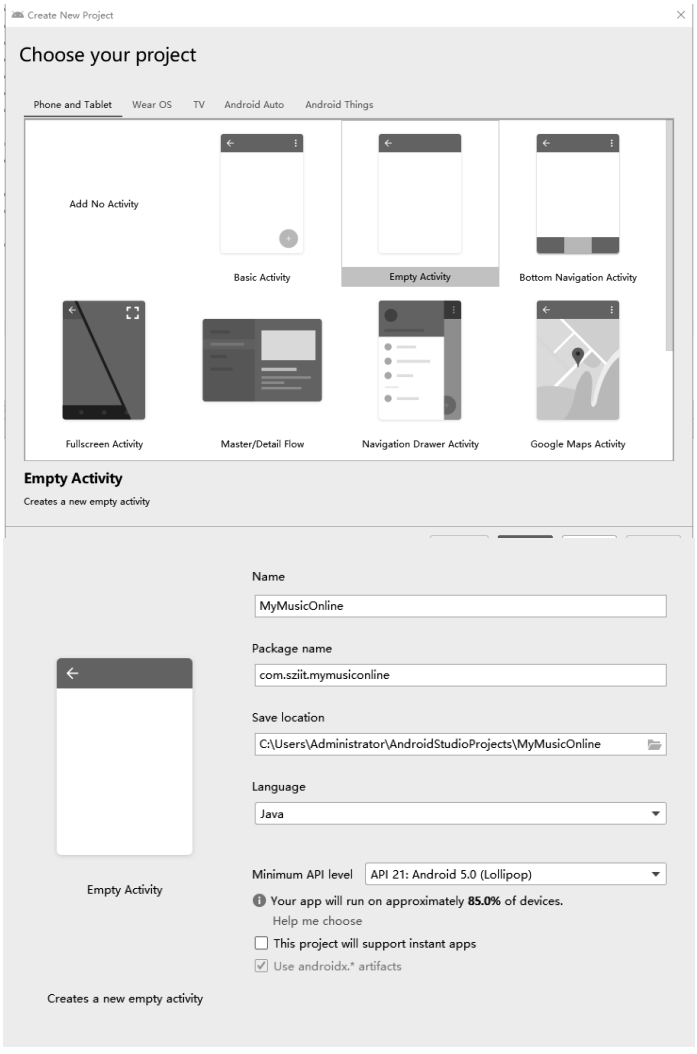

图 1-9　新建一个 Android 工程

工程创建成功后，在包资源管理器中会出现一个 MyMusicOnline 工程，有一个默认的包 com.sziit. mymusiconline，如图 1-10 所示。

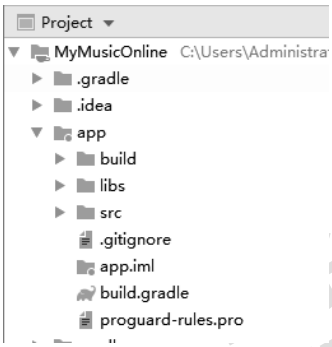

图 1-10　包资源管理器

如图 1-11 所示，展开 src，在 java 类下的包 com.sziit.mymusiconline 上右击，在弹出的快捷菜单中选择“New→Package”命令，建立图 1-12 中的包。

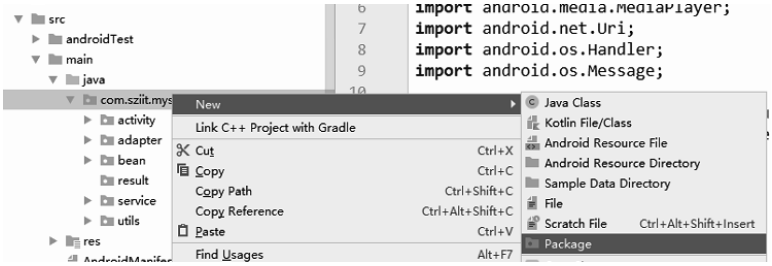

图 1-11　新建包

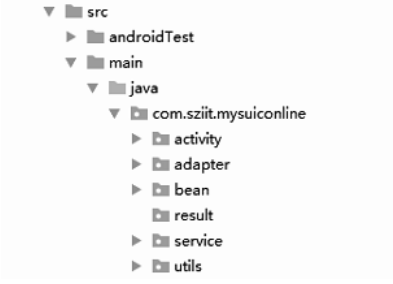

图 1-12　新建的包

再编辑 res\values\strings.xml，方便字符串资源的整理，代码如下。

.\mymusiconline\res\values\strings.xml

```
<resources>
    <string name="app_name">MyMusicOnline</string>
</resources>
```

（2）日志控制类编写

在包 utils 上右击，在弹出的快捷菜单中选择“New→Class”命令，建立一个空的类，命名为 LogUtil，用于控制日志的输出，代码如下。当 isOpen 变量值为 true 时，输出日志。当 isOpen 变量值为 false 时，不输出日志。

.\mymusiconline\src\utils\LogUtil.java

```
package com.sziit.mp3online.utils;
import android.content.Context;
import android.util.Log;
import android.widget.Toast;

public class LogUtil {
    private static final String tag = "myLog";
    private static final boolean isOpen = true;
    /**
     * 打印日志
```

```
     * @param log
     */
    public static void log(String log){
        if(isOpen){
            Log.d(tag, log);
        }
    }

    /**
     * 显示 Toast
     * @param context
     * @param message
     */
    public static void toast(Context context, String message){
        Toast.makeText(context, message, Toast.LENGTH_SHORT).show();
    }

}
```

（3）创建启动画面视图

● 右击包 activity，在弹出的快捷菜单中选择“New→Java Class”命令，打开如图 1-13 所示对话框。

● 在“Name”框中输入“FlashActivity”，单击“OK”按钮。

（4）启动画面布局设计

在 res 下新建一个文件夹 drawable，将本书的相应资源文件复制到该文件夹中，如图 1-14 所示。

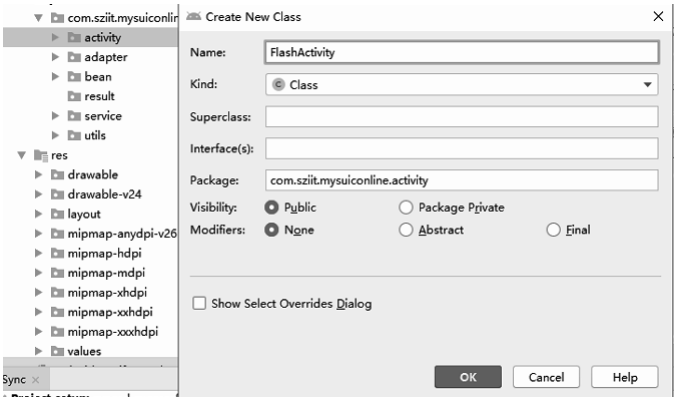

图 1-13 “Create New Class”对话框

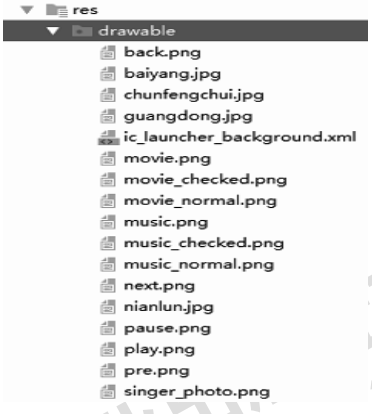

图 1-14 复制资源文件

编辑 activity_flash.xml，代码如下，实现如图 1-15 所示的启动画面。

.\mp3online\res\layout\activity_ flash.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".activity.FlashActivity">
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:orientation="vertical">

        <ImageView
            android:id="@+id/imageView"
            android:layout_width="wrap_content"
            android:layout_height="80dp"
            app:srcCompat="@drawable/music" />

        <TextView
            android:id="@+id/textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:text="在线音乐播放器" />
    </LinearLayout>
</RelativeLayout>
```

图 1-15　播放器启动画面

（5）程序启动动画设计

编辑 FlashActivity.java，代码如下，实现在 2000ms 的屏闪后跳转到播放器主界面。

.\mymusiconline\src\activity\FlashActivity.java

```
package com.sziit.mysuiconline.activity;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.view.Window;
import android.view.WindowManager;
import com.sziit.mysuiconline.R;
public class FlashActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,WindowMana
ger.LayoutParams.FLAG_FULLSCREEN);
        setContentView(R.layout.activity_flash);
        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {
                Intent intent=new Intent(FlashActivity.this,MainActivity.class);
                startActivity(intent);
                FlashActivity.this.finish();
            }
        },2000);
    }
}
```

（6）设置 FlashActivity 为程序启动视图

编辑 AndroidManifest.xml，取消 MainActivity 启动，设置 FlashActivity 为程序启动视图且全屏显示，代码如下。

.\mymusiconline\AndroidManifest.xml

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".activity.FlashActivity" android:theme="@style/
AppThemeA">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <service android:name=".service.MusicService" />
```

```
    <activity android:name=".activity.MainActivity"/>
</application>
```

2. 主界面模块开发

（1）主界面布局设计

编辑 activity_main.xml（代码如下），实现主页面布局如图 1-16 所示，使用 ListView 控件实现歌曲信息的显示。该布局设计较为复杂，采用了线性布局的多层嵌套，如图 1-17 所示。权重设置方法 android:layout_weight="1"在多处出现，目的是使相应控件的宽（对应水平线性布局）或高（对应垂直线性布局）尽可能地充满整个布局空间。

图 1-16　播放器主页面布局

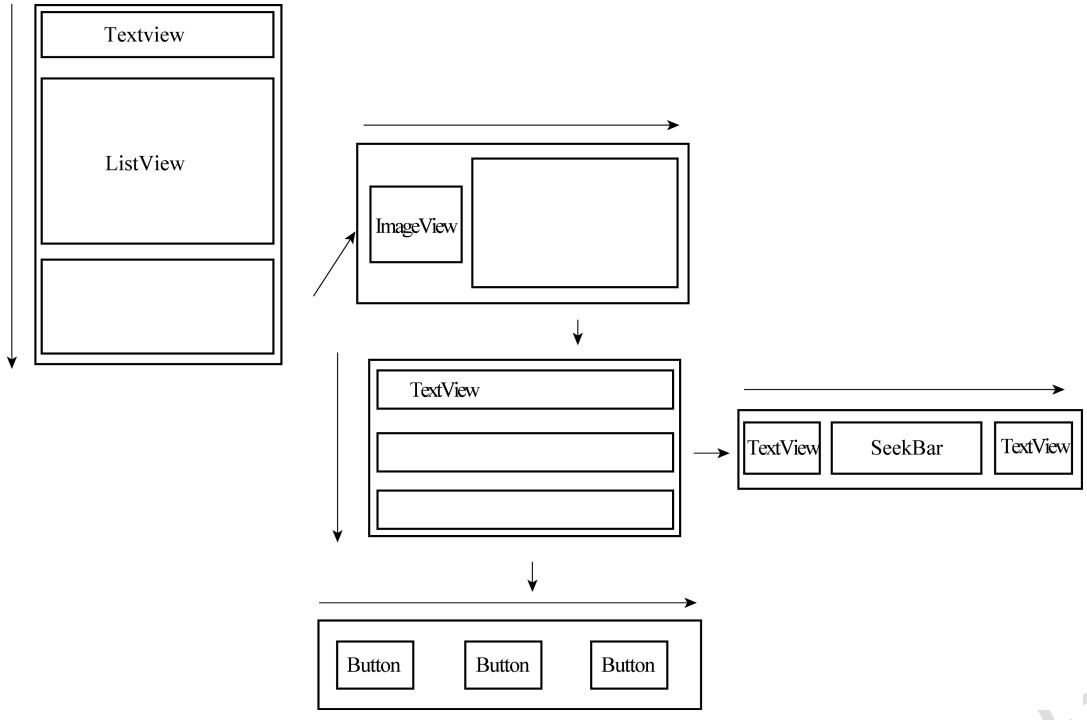

图 1-17　播放器主页面布局设计

.\mymusiconline\res\layout\activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

```
    android:orientation="vertical">
    <ListView
        android:id="@+id/lv"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="8"
        tools:listitem="@layout/list_item" >
    </ListView>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="3"
        android:background="#e9e9e9"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="0.7"
            android:orientation="vertical">
            <TextView
                android:id="@+id/tv"
                android:layout_width="match_parent"
                android:layout_height="0dp"
                android:layout_weight="0.8"
                android:gravity="center"
                android:text="在线音乐播放器"
                android:textSize="17sp" />
        </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_marginBottom="5dp"
            android:layout_weight="2"
            android:orientation="horizontal">
            <ImageView
                android:id="@+id/img"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="0.5"
                android:src="@drawable/singer_photo" />
            <TextView
                android:id="@+id/tv_currentTime"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="0.4"
                android:gravity="center"
                android:text="00:00"
                android:textSize="18sp" />
             <SeekBar
                android:id="@+id/progressBar"
                style="?android:attr/progressBarStyleHorizontal"
```

```
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_weight="1" />
        <TextView
            android:id="@+id/tv_total_time"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="0.4"
            android:gravity="center"
            android:text="00:00"
            android:textSize="18sp" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_marginBottom="5dp"
        android:layout_weight="1">
        <Space
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1.5" />
        <ImageButton
            android:id="@+id/btn_pre"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="0.5"
            android:background="@drawable/pre" />
        <ImageButton
            android:id="@+id/btn_play"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="0.5"
            android:background="@drawable/play" />
        <ImageButton
            android:id="@+id/btn_next"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="0.5"
            android:background="@drawable/next" />
        <Space
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1.5" />
    </LinearLayout>
</LinearLayout>
</LinearLayout>
```

新建布局文件 list_item.xml，为 activity_main.xml 中的 ListView 控件编写布局，使用 TextView 控件实现歌手名与歌曲名的显示，使用 weight 参数调整 TextView 控件宽度，歌

手名占每行的 3/10，歌曲名占每行的 7/10（代码如下），实现效果如图 1-18 所示。

.\mymusiconline\res\layout\list_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginBottom="10dp">
    <TextView
        android:id="@+id/tv_singer_name"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="3"
        android:text="歌手名称"
        android:textSize="24sp" />
    <TextView
        android:id="@+id/tv_music_name"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="7"
        android:text="音乐名称"
        android:textSize="24sp" />
</LinearLayout>
```

图 1-18 ListView 布局

（2）设计 HTTP 访问功能类

在包 utils 中建一个 HttpAsyncRequestTast 类，用于 HTTP 访问。该类继承自异步任务类 AsyncTask，实现 HTTP 访问的后台异步执行。通过参数 HttpUriRequest 获取客户端请求的 URL，并返回服务器相应参数。代码如下。

.\mymusiconline\src\utils\HttpAsyncRequestTast.java

```
package com.sziit.mysuiconline.utils;
import android.os.AsyncTask;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpUriRequest;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.util.EntityUtils;
import java.io.IOException;

public abstract class HttpAsyncRequestTask extends AsyncTask<HttpUriRequest,
```

```
Integer,String> {
    @Override
    protected String doInBackground(HttpUriRequest... httpUriRequests) {
        //HttpGet、HttpPost 都继承自 HttpUriRequest
        HttpUriRequest request=httpUriRequests[0];
        String result=null;
        //实例化一个 http 客户端 client
        HttpClient client=new DefaultHttpClient();
        try {
            //http 客户端 client 执行请求，它可以执行 get 和 post 请求
            HttpResponse response=client.execute(request);
            //把服务器返回的信息转换成 String,并用 utf-8 编码
            result=EntityUtils.toString(response.getEntity(),"UTF-8");
        } catch (ClientProtocolException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return result;
    }
    @Override
    protected void onPostExecute(String s) {
        super.onPostExecute(s);
        //调用 onComplete，传入结果
        onComplete(s);
    }
    protected abstract void onComplete(String s);
    @Override
    protected void onProgressUpdate(Integer... values) {
        super.onProgressUpdate(values);
    }
    @Override
    protected void onCancelled() {
        super.onCancelled();
    }
}
```

（3）设计音乐信息封装类

在 com.sziit.mp3online.bean 中创建一个类 Music，用于封装音乐数据信息。为该类定义属性（title、path、singer），分别存放音乐的歌曲名、音乐文件在服务器的存放路径、歌手名。定义属性后，在代码空白处右击，在弹出的快捷菜单中选择“Source→Generate Getters and Setters”命令，为所有的属性添加 get 与 set 方法。代码如下。

.\mymusiconline\src\bean\Music.java

```
package com.sziit.mysuiconline.bean;
```

```
public class Music {
    private String title;//歌曲名称
    private String name;//歌手名称
    private String path;//歌曲存放的路径
    public String getTitle() {
        return title;
    }
    public void setTitle(String title) {
        this.title = title;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getPath() {
        return path;
    }
    public void setPath(String path) {
        this.path = path;
    }
}
```

（4）设计音乐列表显示适配器

在包 adapter 中新建一个 MusicAdapter 类，作为主界面 ListView 适配器。主界面视图将封装好的音乐信息发送给该适配器，该适配器实现音乐列表的显示。代码如下。

.\mymusiconline\src\adapter\ MusicAdapter.java

```
package com.sziit.mysuiconline.adapter;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.TextView;
import com.sziit.mysuiconline.R;
import com.sziit.mysuiconline.bean.Music;
import java.util.List;

public class MusicAdapter extends BaseAdapter {
    //ctrl+I 实现需要实现的方法
    //Music 对象的数组
    private List<Music> mData;
    //定义上下文
    private Context mCtx;
    //布局动态生成器
    private LayoutInflater layoutInflater;
```

```
    //构造函数
    public MusicAdapter(List<Music> mData, Context mCtx) {
        this.mData = mData;
        this.mCtx = mCtx;
        layoutInflater = LayoutInflater.from(mCtx);
    }
    @Override
    public int getCount() {
        //返回数组长度
        return mData == null ? 0 : mData.size();
    }
    @Override
    public Object getItem(int position) {
        //获取 Music 对象
        return mData == null ? null : mData.get(position);
    }
    @Override
    public long getItemId(int position) {
        //获取子项的位置
        return position;
    }
    //获取子视图
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        ViewHolder mViewHolder;
        if (convertView == null) {
            convertView = layoutInflater.inflate(R.layout.list_item, null);
            mViewHolder=new ViewHolder(convertView);
            convertView.setTag(mViewHolder);
        }else {
            //alt+enter
            mViewHolder= (ViewHolder) convertView.getTag();
        }
        mViewHolder.mTvMusicName.setText(mData.get(position).getTitle());
        mViewHolder.mTvSingerName.setText(mData.get(position).getName());
        return convertView;
    }
    public static class ViewHolder {
        public View rootView;
        public TextView mTvSingerName;
        public TextView mTvMusicName;
        public ViewHolder(View rootView) {
            this.rootView = rootView;
            this.mTvSingerName = (TextView) rootView.findViewById(R.id.tv_
singer_name);
            this.mTvMusicName = (TextView) rootView.findViewById(R.id.tv_
music_name);
        }
```

```
    }
}
```

（5）添加 Internet 访问权限

编辑 AndroidManifest.xml，为在线音乐播放器添加互联网访问权限，代码如下。

.\mymusiconline\AndroidManifest.xml

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

（6）设计音乐播放服务类

在包 service 里添加一个 MusicService 类，用于实现在后台播放音乐的服务。该类主要实现的功能包括：

- 定义一个全局静态变量 KEY_COMMAND，用于接收主视图 MainActivity 发送的不同指令，包括播放、暂停、上一曲、下一曲等。
- 定义一组全局静态变量，用于记录音乐播放信息，包括音乐名、歌手名等。
- 定义一个全局静态变量 CAST_ACTION_UPDATE，用于为播放进度条广播命名。
- 根据主视图 MainActivity 发送的不同指令，调用 MusicUtil 类中的不同方法，实现音乐播放器的控制。

代码如下：

.\mp3online\src\com.sziit.mp3online.services\MusicService.java

```
package com.sziit.mysuiconline.service;
import android.app.Service;
import android.content.Context;
import android.content.Intent;
import android.os.IBinder;
import com.sziit.mysuiconline.utils.MusicUtil;
import java.util.List;

public class MusicService extends Service {
    //定义一个全局静态变量，来接收主视图MainActivity发送的不同指令
    public static final String KEY_COMMAND = "k_command";
    //存放音乐列表索引的键名
    public static final String KEY_MUSIC_INDEX = "k_music_index";
    public static final String KEY_MUSIC_LIST = "k_music_list";
    public static final String MUSIC_TIME_TOTAL = "music_time_total";
    public static final String MUSIC_TIME_CURR = "music_time_curr";
    public static final String CAST_ACTION_UPDATE = "com.sziit.mymusiconline.
MUSIC_TIME_UPDATE";
    //初始化
```

```
    public static final int CMD_INIT = 1000;
    //音乐播放
    public static final int CMD_PLAY = 1001;
    //暂停
    public static final int CMD_PAUSE = 1002;
    //下一曲
    public static final int CMD_NEXT = 1003;
    //上一曲
    public static final int CMD_PRE = 1004;
    //停止
    public static final int CMD_STOP = 1005;
    //从暂停状态恢复
    public static final int CMD_RESUME = 1006;

    private MusicUtil musicUtil;
    private Context mCtx;
    @Override
    public void onCreate() {
        super.onCreate();
        mCtx = this;
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        int command = intent.getIntExtra(KEY_COMMAND, -1);
        switch (command) {
            case CMD_INIT:
                //获取音乐路径列表
                List<String>
musicList=intent.getStringArrayListExtra(KEY_MUSIC_LIST);
                musicUtil=new MusicUtil(musicList,mCtx);
                break;
            case CMD_PLAY:
                int index=intent.getIntExtra(KEY_MUSIC_INDEX,0);
                musicUtil.play(index);
                break;
            case CMD_PAUSE:
                musicUtil.pause();
                break;
            case CMD_NEXT:
                musicUtil.next();
                break;
            case CMD_PRE:
                musicUtil.pre();
                break;
            case CMD_STOP:
                musicUtil.stop();
                break;
```

```
            case CMD_RESUME:
                musicUtil.play();
                break;
            default:
                break;
        }
        return super.onStartCommand(intent, flags, startId);
    }
    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }
}
```

（7）注册音乐播放服务

编辑 AndroidManifest.xml，注册音乐播放服务 MusicService，代码如下：

.\mymusiconline\AndroidManifest.xml

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

（8）设计音乐播放功能类

在包 utils 里添加一个 MusicUtil 类，用于实现对音乐播放器的具体操作。该类主要实现的功能包括：

- 根据 MusicService 传递的不同指令，调用 Android 系统提供的 MediaPlayer 类，实现音乐的播放、暂停、上一曲、下一曲等操作。
- 在音乐开始播放后，监听音乐播放的进度，将进度信息作为广播 CAST_ACTION_UPDATE 发送，从而实现主界面中的进度条功能。

代码如下：

.\mymusiconline\src\utils\MusicUtil.java

```
package com.sziit.mysuiconline.utils;
import android.annotation.SuppressLint;
import android.content.Context;
import android.content.Intent;
import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Handler;
import android.os.Message;
import com.sziit.mysuiconline.bean.Music;
import com.sziit.mysuiconline.service.MusicService;
import java.io.IOException;
import java.util.List;

public class MusicUtil {
```

```
    private List<String> musics = null;
    //媒体播放器
    private MediaPlayer mediaPlayer;
    private Context mCtx;
    //歌曲路径索引
    public static int index = 0;
    public boolean isPlaying = false;
    @SuppressLint("HandlerLeak")
    Handler handler = new Handler() {
        @Override
        public void handleMessage(Message msg) {
            super.handleMessage(msg);
            Intent intent = new Intent(MusicService.CAST_ACTION_UPDATE);
            intent.putExtra(MusicService.KEY_MUSIC_INDEX, index);
            intent.putExtra(MusicService.MUSIC_TIME_CURR,
mediaPlayer.getCurrentPosition());
            intent.putExtra(MusicService.MUSIC_TIME_TOTAL,
mediaPlayer.getDuration());
            mCtx.sendBroadcast(intent);
            handler.sendEmptyMessageDelayed(1, 500);
        }
    };
    public MusicUtil(List<String> musics, Context mCtx) {
        this.musics = musics;
        this.mCtx = mCtx;
        mediaPlayer = new MediaPlayer();
        mediaPlayer.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
            @Override
            public void onCompletion(MediaPlayer mediaPlayer) {
                next();
            }
        });
    }
    public void stop() {
        if (mediaPlayer != null) {
            mediaPlayer.stop();
            isPlaying = false;
        }
    }
    public void pause() {
        if (mediaPlayer != null) {
            mediaPlayer.pause();
            isPlaying = false;
        }
    }
    public void next() {
        if (musics != null) {
            if (index == musics.size() - 1) {
                index = 0;
```

```
            } else {
                ++index;
            }
            play(index);
        }
    }
    public void pre() {
        if (musics != null) {
            if (index == 0) {
                index = musics.size() - 1;
            } else {
                --index;
            }
            play(index);
        }
    }
    public void play() {
        mediaPlayer.start();
        isPlaying = true;
        handler.sendEmptyMessage(1);
    }
    //播放指定位置音乐
    @SuppressLint("SdCardPath")
    public void play(int index) {
        MusicUtil.index = index;
        if (musics != null) {
            String music = musics.get(MusicUtil.index);
            try{
                //重用MediaPlayer对象
                mediaPlayer.reset();
                //从网络读取文件
                mediaPlayer.setDataSource(mCtx, Uri.parse(music));
                mediaPlayer.prepare();
                mediaPlayer.start();
                isPlaying = true;
                handler.sendEmptyMessage(0);
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}
```

（9）设计主界面视图类

编辑 MainActivity.java，最终实现在线音乐播放器。该类主要实现的功能包括：

● 定义一个全局静态变量 public static final String HTTP_ROOT = " http://10.254.72.35/"，用于记录服务器根目录。注意，请根据自己的计算机查看 IP 地址。查看方法为按 WIN+R

组合键，在弹出的的窗口中输入“cmd”，如图 1-19 所示，然后在 DOS 界面中输入“ipconfig”，字符串“IPv4 地址”后面的就是本机的 IP，如图 1-20 所示。

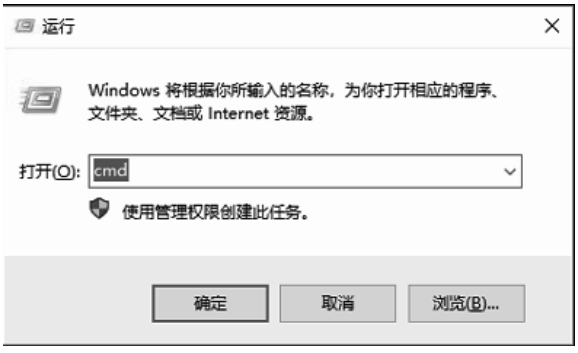

图 1-19　运行 cmd 命令

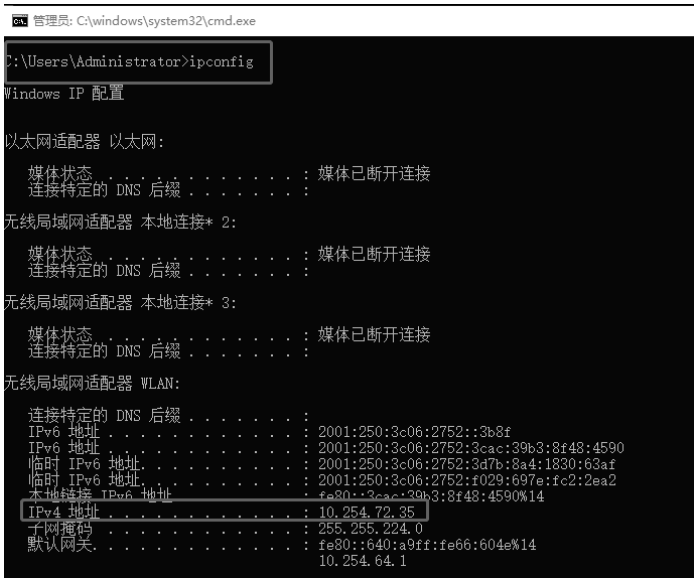

图 1-20　显示本机 IP 地址

- 将音乐信息使用 Music 类封装后，发送给 MusicAdapter，实现音乐列表显示。
- 根据用户单击的不同按钮，向 MusicService 发送不同指令，根据用户选择的音乐列表位置，向 MusicService 发送位置信息，实现音乐播放功能。
- 设计一个广播接收内部类 MusicReceiver，以 CAST_ACTION_UPDATE 注册广播接收器，接收 MusciUtil 发送的进度条状态广播，实现进度条更新。
- 定义一个格式化函数，将毫秒转换为常用的时间格式，实现进度条时间信息的格式化显示。

代码如下：

.\mymusiconline\src\activitys\MainActivity.java

```
package com.sziit.mysuiconline.activity;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.SeekBar;
import android.widget.TextView;
import com.sziit.mysuiconline.R;
import com.sziit.mysuiconline.adapter.MusicAdapter;
import com.sziit.mysuiconline.bean.Music;
import com.sziit.mysuiconline.service.MusicService;
import com.sziit.mysuiconline.utils.HttpAsyncRequestTask;
import org.apache.http.client.methods.HttpGet;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity implements
View.OnClickListener {
    public static final String HTTP_ROOT = "http://10.254.72.35/";
    private MusicAdapter musicAdapter;
    public static boolean isPlaying = false;
    public static List<Music> mMusics;
    private int musicIndex = -1;
    private Context mCtx;
    private MusicReceiver mMusicReceiver;
    private int[] singerPic = {R.drawable.guangdong, R.drawable.baiyang,
R.drawable.chunfengchui, R.drawable.nianlun};
    private ListView mLv;
    private ImageView mImageView;
    private TextView mTv;//歌曲名称
    private ImageView mImg;
    private TextView mTvCurrentTime;
    private SeekBar mProgressBar;
    private TextView mTvTotalTime;
    private ImageButton mBtnPre;
    private ImageButton mBtnPlay;
    private ImageButton mBtnNext;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mCtx = this;
        initView();
        HttpAsyncRequestTask task = new HttpAsyncRequestTask() {
            @Override
            protected void onComplete(String s) {
                try {
                    JSONArray array = new JSONArray(s);
                    mMusics = new ArrayList<Music>();
                    Music music;
                    for (int i = 0; i < array.length(); i++) {
                        music = new Music();
                        JSONObject jsonObject = array.getJSONObject(i);
                        music.setTitle(jsonObject.getString("name"));
                        music.setPath(jsonObject.getString("mp3"));
                        music.setName(jsonObject.getString("singer"));
                        mMusics.add(music);
                    }
                    musicAdapter = new MusicAdapter(mMusics, mCtx);
                    mLv.setAdapter(musicAdapter);
                    Intent intent = new Intent(mCtx, MusicService.class);
                    //音乐路径列表
                    ArrayList<String> musicList = new ArrayList<String>();
                    for (Music music1 : mMusics) {
                        musicList.add(HTTP_ROOT + music1.getPath());
                    }
                    //初始化指令
                    intent.putExtra(MusicService.KEY_COMMAND, MusicService.
CMD_INIT);
                    //传递音乐路径列表

intent.putStringArrayListExtra(MusicService.KEY_MUSIC_LIST, musicList);
                    //启动服务
                    startService(intent);
                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        };
        //执行异步线程
        task.execute(new HttpGet(HTTP_ROOT + "music/music.json"));

        //ListView 子项监听器
        mLv.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> adapterView, View view, int
i, long l) {
```

```
                Intent intent = new Intent(mCtx, MusicService.class);
                //设置初始化指令
                intent.putExtra(MusicService.KEY_COMMAND, MusicService. CMD_PLAY);
                //传递播放位置
                intent.putExtra(MusicService.KEY_MUSIC_INDEX, i);
                startService(intent);
                isPlaying = true;
                mTv.setText(mMusics.get(i).getTitle());
                mImageView.setImageResource(singerPic[i]);
            }
        });
    }

    private void initView() {
        mLv = (ListView) findViewById(R.id.lv);
        mImageView = (ImageView) findViewById(R.id.img);
        mTv = (TextView) findViewById(R.id.tv);

        //注册广播
        mMusicReceiver = new MusicReceiver();
        IntentFilter filter = new IntentFilter(MusicService.CAST_ACTION_UPDATE);
        registerReceiver(mMusicReceiver, filter);

        mImg = (ImageView) findViewById(R.id.img);
        mImg.setOnClickListener(this);
        mTvCurrentTime = (TextView) findViewById(R.id.tv_currentTime);
        mTvCurrentTime.setOnClickListener(this);
        mProgressBar = (SeekBar) findViewById(R.id.progressBar);
        mProgressBar.setOnClickListener(this);
        mProgressBar.setMax(-999);
        mTvTotalTime = (TextView) findViewById(R.id.tv_total_time);
        mTvTotalTime.setOnClickListener(this);
        mBtnPre = (ImageButton) findViewById(R.id.btn_pre);
        mBtnPre.setOnClickListener(this);
        mBtnPlay = (ImageButton) findViewById(R.id.btn_play);
        mBtnPlay.setOnClickListener(this);
        mBtnNext = (ImageButton) findViewById(R.id.btn_next);
        mBtnNext.setOnClickListener(this);
    }

    @Override
    public void onClick(View v) {
        Intent intent=new Intent(mCtx,MusicService.class);
        switch (v.getId()) {
            case R.id.btn_pre:

intent.putExtra(MusicService.KEY_COMMAND,MusicService.CMD_PRE);

                break;
            case R.id.btn_play:
```

```
                if (isPlaying){

intent.putExtra(MusicService.KEY_COMMAND,MusicService.CMD_PAUSE);
                    isPlaying=false;
                    mBtnPlay.setBackgroundResource(R.drawable.play);
                }else {

intent.putExtra(MusicService.KEY_COMMAND,MusicService.CMD_RESUME);
                    isPlaying=true;
                    mBtnPlay.setBackgroundResource(R.drawable.pause);
                }

                break;
            case R.id.btn_next:

intent.putExtra(MusicService.KEY_COMMAND,MusicService.CMD_NEXT);

                break;
        }
        startService(intent);
    }

    @Override
    protected void onRestart() {
        super.onRestart();
        if (isPlaying=false){
            mBtnPlay.setBackgroundResource(R.drawable.play);
        }else {
            mBtnPlay.setBackgroundResource(R.drawable.pause);
        }
    }

    private class MusicReceiver extends BroadcastReceiver {
        @Override
        public void onReceive(Context context, Intent intent) {
            int index = intent.getIntExtra(MusicService.KEY_MUSIC_INDEX, 0);
            int curr = intent.getIntExtra(MusicService.MUSIC_TIME_CURR, 0);
            int total = intent.getIntExtra(MusicService.MUSIC_TIME_TOTAL, 0);
            //-1
            if (musicIndex != index) {
                musicIndex = index;
                mProgressBar.setMax(total);
                mTvTotalTime.setText(format(total));
                mTv.setText(mMusics.get(musicIndex).getTitle());
                mBtnPlay.setBackgroundResource(R.drawable.pause);
            }
            mProgressBar.setProgress(curr);
            mTvCurrentTime.setText(format(curr));
        }
    }
```

```
    private static String format(long time) {
        time=time/1000;
        if (time<60){
            return "00:"+formatLong("00",time);
        }else {
            return formatLong("00",time/60)+":"+formatLong("00",time%60);
        }
    }

    private static String formatLong(String s, long l) {
        return new DecimalFormat(s).format(l);
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
        unregisterReceiver(mMusicReceiver);
    }
}
```

1.2.3 测试

代码编写完成后，需用真机在局域网环境中进行测试，以确保项目在真实环境中是可行的。

（1）将手机与计算机接入同一个局域网子网。

图 1-21 运行效果

（2）开启计算机上的 XAMPP 套件，打开 Apache 服务器。

（3）查看计算机 IP 地址，将 MainAcivity 中的服务器地址设置成该 IP 地址。

（4）使用数据线连接手机与计算机，在 MyMusicOnline 项目上右击，在弹出的快捷菜单中选择“Run→Run'app'”命令，或者直接单击标题栏中的▶图标，在弹出的窗口中选择测试用手机，即可将安卓项目发布到手机上。

（5）如果项目在手机上可正常开启，并在主视图中正常显示了服务器上所有歌曲的信息，则表示客户端与服务端通信成功。主界面最终显示效果如图 1-21 所示，服务器端 JSON 文件中的文本信息显示在客户端的 ListView 控件中。

相关知识点与课后题

1. 知识点

（1）JSON 对象的解析

JSON 对象的格式通常为：

```
{"name":"成都","singer":"赵雷","mp3":"music/101.mp3"},。
```

以大括号“}”为标志的 JSON 参数是无序的对象，客户端的相应解析较为简单，设服务端返回的数据为封装好的 JSON 数据 jsonData。下面代码可以获取服务端返回的键名为“singer”的参数的相应键值。

```
JSONObject jsonObject = new JSONObject(jsonData);
singer = jsonObject.getString("singer");
```

（2）JSON 数组解析

多个 JSON 对象构成一个 JSON 数组。 JSON 数组以方括号“[ ]”为标志，如在线音乐播放器服务端的 JSON 数组。

D:\xampp\htdocs\music\music.json

```
[{"name":"成都","singer":"赵雷","mp3":"music/101.mp3"},
    {"name":"昨日晴空","singer":"尤长靖","mp3":"music/102.mp3"},
    {"name":"可惜爱","singer":"杨乃文","mp3":"music/103.mp3"},
    {"name":"别","singer":"薛之谦","mp3":"music/104.mp3"},
    {"name":"广东爱情故事","singer":"广东雨神","mp3":"music/105.mp3"}]
```

上面的数据为一个数组形式，可以用 Android 提供的框架 JSONArray 读取 JSON 数据，再转换成 Array，代码如下。

.\mymusiconline\src\activity\MainActivity.java

```
JSONArray array = new JSONArray(s);
mMusics = new ArrayList<Music>();
Music music;
for (int i = 0; i < array.length(); i++) {
    music = new Music();
    JSONObject jsonObject = array.getJSONObject(i);
    music.setTitle(jsonObject.getString("name"));
    music.setPath(jsonObject.getString("mp3"));
    music.setName(jsonObject.getString("singer"));
```

```
    mMusics.add(music);
}
```

2. 课后题

（1）尝试更新服务器上的音乐资源，使在线音乐播放器播放不同的音乐。

（2）尝试使用 MySQL 数据库存放音乐信息。思考通过 JSON 文件存放音乐信息与通过 MySQL 数据库存放音乐信息的异同。