1. **需求分析**

功能要求：

1）能够下载并存储mp3音乐文件到Android设备的内部存储空间，下载功能需要以服务的形式实现（可以使用IntentService实现），作为参考，下列网站可以找到部分可供下载的音乐文件：

<https://freemusicarchive.org/genre/Classical/>

2）以ListView的形式显示播放列表，请设置至少包含5首曲目的播放列表用于测试；点击播放列表中的表项后，若文件不存在则下载文件，下载完成后开始播放对应歌曲；若文件存在则直接开始播放。

3）实现一个歌曲播放界面，包括上一首、下一首、播放/暂停、播放进度等按钮并能够显示曲目名称。

非功能要求：界面按钮交互友好，意图清晰，使用户能够一目了然界面如何使用以及软件的功能。程序具有一定健壮性，要能够处理各种潜在的异常。

设计约束：界面约束，ConstraintLayout, LinearLayout结合使用，以此来适配不同型号和屏幕大小的设备。

1. **软件设计**

说明：请在这里根据需求分析的内容，进行软件设计。需要阐明App采用的架构和实现方法，如所使用的主要安卓组件，功能模块划分，数据库设计等。

1. 用户进入后能看到本地音乐组成的播放列表，用listView实现列表呈现，File类和Manifest.xml中注册READ权限来获得音乐文件名。点击列表上某一首歌可以跳转到一个新的Activity来播放音乐，用listView上单独item的clickListener实现。
2. 播放界面，要能显示当前歌曲名，一个音乐图案。播放▶️、暂停⏸️、快进⏩、快退⏪、下一首、上一首，用ImageButton实现。并且切换歌曲时音乐图案有360度旋转动画，用ObjectAnimator实现。歌曲进度条用seekBar实现，seekBar的更新用另一个线程实现。同时在进度条两边显示时间。在获得歌曲时长信息后，播放完的歌曲要能够自动切换下一首，直到整个列表播放结束。
3. 主界面有一个🔍搜索按钮SearchView，用来搜索网络上的音乐，搜索内容提交后，利用OkHttp类的对象提交并且接收报文。利用PostMan软件观察Request和Response的结构特征，将Response生成Document对象进行解析，来获得歌曲信息{歌曲名、歌手、专辑、下载链接}，把歌曲名和歌手组成新的列表刷新listView。此时用户点击歌曲则开始下载，下载完成后有提示，点击返回按钮或者关闭搜索框即可回到本地音乐listView，并且能看到刚下载好的音乐。
4. 由于第一次获得的链接是重定向链接，所以需要解析出最终的mp3音乐文件链接，所以利用递归的方式解析。
5. 下载的时候，把最终的url传到intentService下载。IntentService是一个队列式的单线程，所以可以依次点击多次，依次下载。
6. **核心程序代码及运行结果（30分）**

本次实验的Github项目地址：<https://github.com/Qervas/Android/tree/MusicPlayer>

部分代码如下(省略布局和控件):

播放音乐时，页面最底部加入了动画，是来源于github开源库的。

Import com.gauravk.audiovisualizer.visualizer.BarVisualizer;

visualizer.setAudioSessionId(audiosessionId);

在读到音乐文件的时间信息后，要转化为人类能看懂的时间格式。

public String createTime(int duration){  
 String time = "";  
 int min = duration/1000/60;  
 int sec = duration/1000%60;  
 time += min + ":";  
  
 if(sec < 10){  
 time += "0";  
 }  
 time +=sec;  
 return time;  
}

递归解析重定向链接：

public String getRedirectUrl(String downloadUrl) throws IOException {  
 Log.*e*("redirecting",downloadUrl);  
 HttpURLConnection con = (HttpURLConnection) new URL(downloadUrl).openConnection();  
 con.setInstanceFollowRedirects(false);  
 con.connect();  
 con.getInputStream();  
  
 if (con.getResponseCode() == HttpURLConnection.*HTTP\_MOVED\_PERM* || con.getResponseCode() == HttpURLConnection.*HTTP\_MOVED\_TEMP*) {  
 String redirectUrl = con.getHeaderField("Location");  
 return getRedirectUrl(redirectUrl);  
 }  
 return downloadUrl;  
}

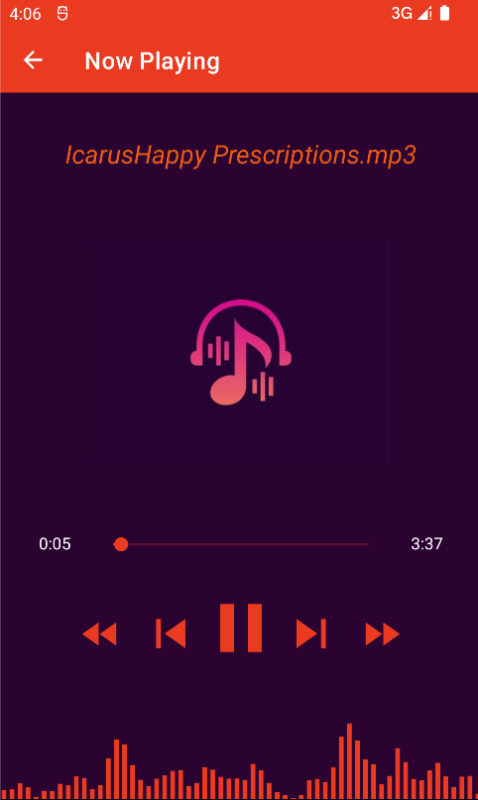
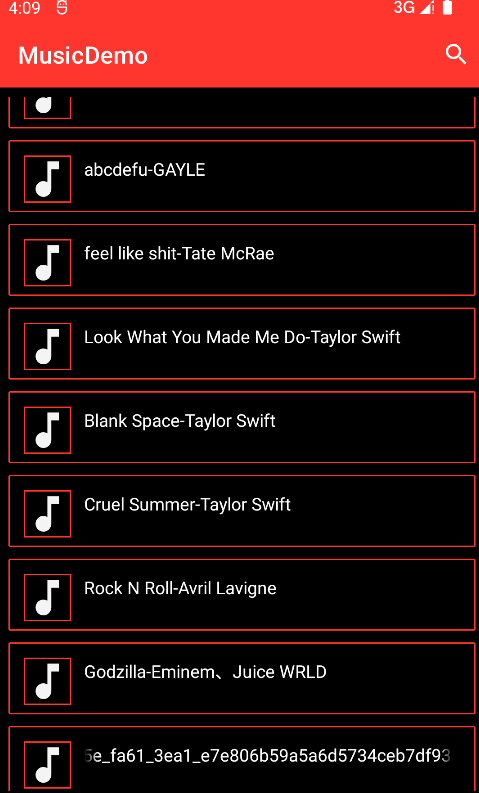
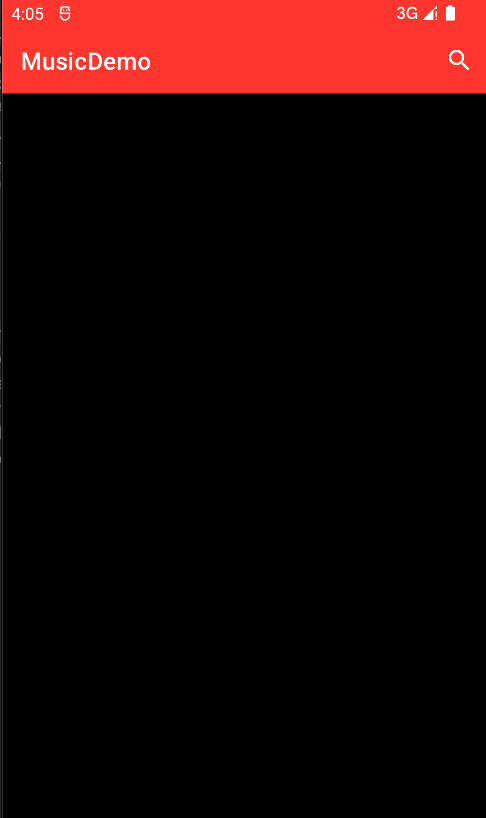
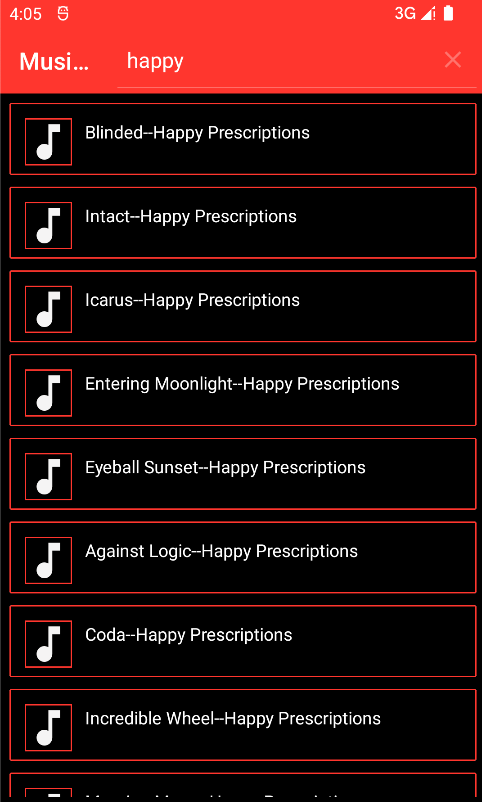
display函数有两个版本，没有参数的是显示本地歌曲，有参数的是显示网络歌曲：

void displaySongs(){  
  
 final ArrayList<File> mySongs = findSong(new File(Environment.*getExternalStorageDirectory*().getPath()+"/Download"));  
 items = new String[mySongs.size()];  
 for(int i = 0; i < mySongs.size(); i++){  
 items[i] = mySongs.get(i).getName().toString().replace(".mp3","").replace(".wav","");  
 }  
 adapter = new customAdapter();  
 listView.setAdapter(adapter);  
  
 listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {  
 @Override  
 public void onItemClick(AdapterView<?> parent, View view, int position, long id) {  
 String songName = (String) listView.getItemAtPosition(position);  
 startActivity(new Intent(getApplicationContext(),PlayerActivity.class)  
 .putExtra("songs",mySongs)  
 .putExtra("songname",songName)  
 .putExtra("pos",position));  
 }  
 });  
 }  
 void displaySongs(Elements elements){  
 String songName, artistName,downloadUrl;  
 String temp="";  
 JSONObject jsonObject;  
 ArrayList<songInfo> mySongs = new ArrayList<>();  
 for (Element element:elements){  
 temp = element.attr("data-track-info");  
 if(temp.contains("title")&&temp.contains("artistName")&&temp.contains("downloadUrl")){//保证一一对应  
 try {  
 jsonObject= new JSONObject(temp);  
 songName = jsonObject.getString("title");  
 artistName = jsonObject.getString("artistName");  
 downloadUrl = jsonObject.getString("downloadUrl");  
 mySongs.add(new songInfo(songName,artistName,downloadUrl));  
// Log.e("歌",songName+"--"+artistName+" "+downloadUrl);  
 } catch (JSONException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 items = new String[mySongs.size()];  
 for (int i = 0; i < mySongs.size(); i++){  
 items[i] = mySongs.get(i).getItemInfo();  
 }  
 adapter = new customAdapter();  
 listView.setAdapter(adapter);  
 listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {  
 @Override  
 public void onItemClick(AdapterView<?> parent, View view, int position, long id) {//position starts from 0 as from top to bottom  
 String info = (String) listView.getItemAtPosition(position);  
 String downloadUrl = mySongs.get(position).getDownloadUrl();  
 //solve Redirection  
  
 Intent downloadIntent = new Intent(MainActivity.this,DownloadService.class)  
 .putExtra("songName\_artistName",mySongs.get(position).getTitle()+mySongs.get(position).getArtistName())  
 .putExtra("downloadUrl", downloadUrl)  
 ;  
 startService(downloadIntent);  
 }  
 });  
  
 }

在搜索框提交文字以后，向音乐网站发出http请求，将获得的报文根据已经提前在postman观察好的标签字段进行解析，获得所需信息。并把有用信息作为参数传给display(Elements elements)来显示网络歌曲：

public boolean onQueryTextSubmit(String query) {  
 OkHttpClient client = new OkHttpClient();  
 String baseUrl = "https://freemusicarchive.org/search/?quicksearch="+ searchView.getQuery();  
 Request request = new Request.Builder()  
 .url(baseUrl)  
 .build();  
 client.newCall(request).enqueue(new Callback() {  
 @Override  
 public void onFailure(@NonNull Call call, @NonNull IOException e) {  
 e.printStackTrace();  
 }  
 @Override  
 public void onResponse(@NonNull Call call, @NonNull Response response) throws IOException {  
  
 if(response.isSuccessful()) {  
 final String myResponse = response.body().string();  
 MainActivity.this.runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 Document doc = Jsoup.*parse*(myResponse);  
 Elements elements = doc.select("div[data-track-info]");  
 displaySongs(elements);  
 }  
 });  
  
 }  
 }  
 });  
 return true;  
}

执行结果如下：



**参考文献（10分）**

[1] 安卓官网文档Documentation for app developers <https://developer.android.com/docs>

[2] 课本《第一行代码》郭霖

[3] Dexter <https://github.com/Karumi/Dexter>

[4] audio-visualizer-android <https://github.com/gauravk95/audio-visualizer-android>

[5] jsoup: Java HTML Parser <https://jsoup.org/apidocs/>

[6] OkHttp <https://square.github.io/okhttp/>