

EE317 - Application Design: Music Player

Name: 张策 (Zhang Ce)

SID: 11910803

Part 1 - Introduction

In the era of mobile big data, mobile applications have been integrated with our lives, providing us with a lot of convenience. For example, people can directly buy goods on many online shopping apps, saving the trouble of offline purchases; people can also communicate with friends and classmates in real time on chat apps, and the accuracy of message transmission guarantees people's communication needs .

On the other hand, people also have a strong demand for a platform that can store large amount of music sources which can entertain themselves. Nowadays, music player applications like NetEase and QQ music are making a great impact on people by now. Therefore, it is a good problem for us to practice.

In this paragraph, I will cover the latter parts of the report. In the second part, I will introduce how I designed the user interface of this music player app, and give the corresponding display; in the third part, I will introduce how the music player app implements logical operations, and demonstrate some real-scene tests. In the fourth part, I will focus on demonstration on real mobile phones.

Part 2 - User Interface

In Part 2, I will guide you through how I designed the user interface and present you the final version.

For the user interface design, *Linear layout* is used for the general interface, containing the playing interface and the song list interface. Specifically, in the song list, we use *Recycler View* to display the songs that can be played. The related codes are presented as the following.

activity_main.xml:

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".MainActivity"
8      android:orientation="vertical">
9      <TextView
10         android:layout_width="match_parent"
11         android:layout_height="wrap_content"
12         android:text="Song List"
13         android:textAllCaps="false"
14         android:fontFamily="@font/gothicb"
```

```

15         android:textColor="#FF8C94"
16         android:textSize="25sp"
17         android:gravity="center"
18         android:background="#FFD3B6"/>
19     <androidx.recyclerview.widget.RecyclerView
20         android:id="@+id/rv"
21         android:layout_width="match_parent"
22         android:layout_height="wrap_content"
23         android:background="#FFECDA"/>
24 </LinearLayout>

```

As for the playing interface, there exist some buttons, image view boxes and text view boxes, so we adopt *linear layout* with *relative layout* inside. Besides, a seek bar is adopted to display the current playing progress. The full-edition code will be released at Appendix. The related codes are presented as the following.

activity_song1.xml:

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:background="#7CEBFFFA"
8      tools:context=".SongActivity1"
9      android:orientation="vertical">
10     <TextView
11         android:layout_width="match_parent"
12         android:layout_height="wrap_content"
13         android:fontFamily="@font/gothicb"
14         android:id="@+id/name"
15         android:text="Song of PIAO"
16         android:gravity="center"
17         android:layout_marginTop="35dp"
18         android:textColor="#07689f"
19         android:textSize="30sp"/>
20
21     <TextView
22         android:layout_width="match_parent"
23         android:layout_height="wrap_content"
24         android:text="PIAO"
25         android:id="@+id/singer"
26         android:fontFamily="@font/gothici"
27         android:gravity="center"
28         android:layout_marginTop="3dp"
29         android:textColor="#07689f"
30         android:textSize="20sp"/>
31
32     <ImageView
33         android:id="@+id/iv_cover"
34         android:layout_width="260dp"
35         android:layout_height="260dp"
36         android:layout_gravity="center_horizontal"
37         android:layout_marginTop="40dp"

```

```

38         android:src="@drawable/cover" />
39
40     <SeekBar
41         android:id="@+id/sb"
42         android:layout_width="match_parent"
43         android:layout_height="wrap_content"
44         android:maxHeight="10.0dp"
45         android:minHeight="10.0dp"
46         style="@style/CustomSeekBarStyle"
47         android:thumb="@drawable/sb_thumb"
48         android:layout_marginTop="40dp"/>
49
50     <RelativeLayout
51         android:layout_width="match_parent"
52         android:layout_height="wrap_content"
53         android:layout_margin="10dp">
54
55         <TextView
56             android:id="@+id/tv_progress"
57             android:layout_width="wrap_content"
58             android:layout_height="wrap_content"
59             android:fontFamily="@font/gothici"
60             android:text="00:00"
61             android:textColor="@color/black"
62             android:textSize="16sp" />
63         <TextView
64             android:id="@+id/tv_total"
65             android:layout_width="wrap_content"
66             android:layout_height="wrap_content"
67             android:fontFamily="@font/gothici"
68             android:text="04:50"
69             android:layout_alignParentRight="true"
70             android:textSize="16sp"
71             android:textColor="@color/black"/>
72
73     </RelativeLayout>
74     <LinearLayout
75         android:layout_width="match_parent"
76         android:layout_height="wrap_content"
77         android:orientation="horizontal"
78         android:layout_margin="5dp">
79         <Button
80             android:id="@+id/btn_play"
81             android:layout_width="wrap_content"
82             android:layout_height="wrap_content"
83             android:background="@drawable/shape"
84             android:fontFamily="@font/gothicb"
85             android:textAllCaps="false"
86             android:text="Start"
87             android:textColor="#07689f"
88             android:textSize="15sp"
89             android:layout_weight="1"
90             android:layout_margin="5dp"/>
91         <Button
92             android:id="@+id/btn_pause"

```

```

93         android:layout_width="wrap_content"
94         android:layout_height="wrap_content"
95         android:background="@drawable/shape"
96         android:fontFamily="@font/gothicb"
97         android:textAllCaps="false"
98         android:text="Pause"
99         android:textColor="#07689f"
100        android:textSize="15sp"
101        android:layout_weight="1"
102        android:layout_margin="5dp"/>
103    <Button
104        android:id="@+id/btn_continue"
105        android:layout_width="wrap_content"
106        android:layout_height="wrap_content"
107        android:textAllCaps="false"
108        android:background="@drawable/shape"
109        android:fontFamily="@font/gothicb"
110        android:text="Continue"
111        android:textSize="15sp"
112        android:textColor="#07689f"
113        android:layout_weight="1"
114        android:layout_margin="5dp"/>
115    <Button
116        android:id="@+id/btn_exit"
117        android:layout_width="wrap_content"
118        android:layout_height="wrap_content"
119        android:background="@drawable/shape"
120        android:textAllCaps="false"
121        android:fontFamily="@font/gothicb"
122        android:text="Exit"
123        android:textColor="#07689f"
124        android:textSize="15sp"
125        android:layout_margin="5dp"
126        android:layout_weight="1"/>
127
128    </LinearLayout>
129
130 </LinearLayout>

```

Since the original seek bar is not beautiful, I choose to use a more elegant design of the seek bar.

seekbar_progress_drawable.xml (to change the background color of the seek bar):

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <layer-list xmlns:android="http://schemas.android.com/apk/res/android">
3      <!--定义seekbar滑动条的底色-->
4      <item android:id="@android:id/background">
5          <shape>
6              <corners android:radius="5dp" />
7              <gradient
8                  android:angle="270"
9                  android:centerColor="#eeeff3"
10                 android:centerY="0.75"
11                 android:endColor="#eeeff3"
12                 android:startColor="#eeeff3" />

```

```

13         </shape>
14     </item>
15     <!--定义seekbar滑动条进度颜色-->
16     <item android:id="@android:id/progress">
17         <clip>
18             <shape>
19                 <corners android:radius="5dp"/>
20                 <solid android:color="#FFD3B6"/>
21             </shape>
22         </clip>
23     </item>
24 </layer-list>

```

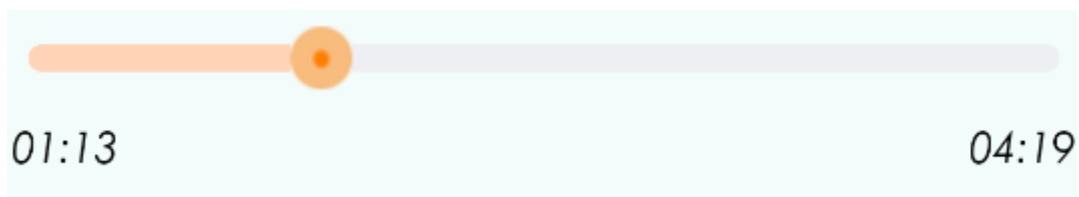
sb_thumb.xml (to change the thumb):

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <selector xmlns:android="http://schemas.android.com/apk/res/android">
3     <item android:state_pressed="true"
4         android:drawable="@drawable/seekbar_thumb_pressed"/>
5     <item android:state_pressed="false"
6         android:drawable="@drawable/seekbar_thumb_normal"/>
7 </selector>

```

Finally, the seek bar design is shown as the following.



As for the color choices, I designed a button style and be called by the buttons using `android:background="@drawable/shape"`.

```

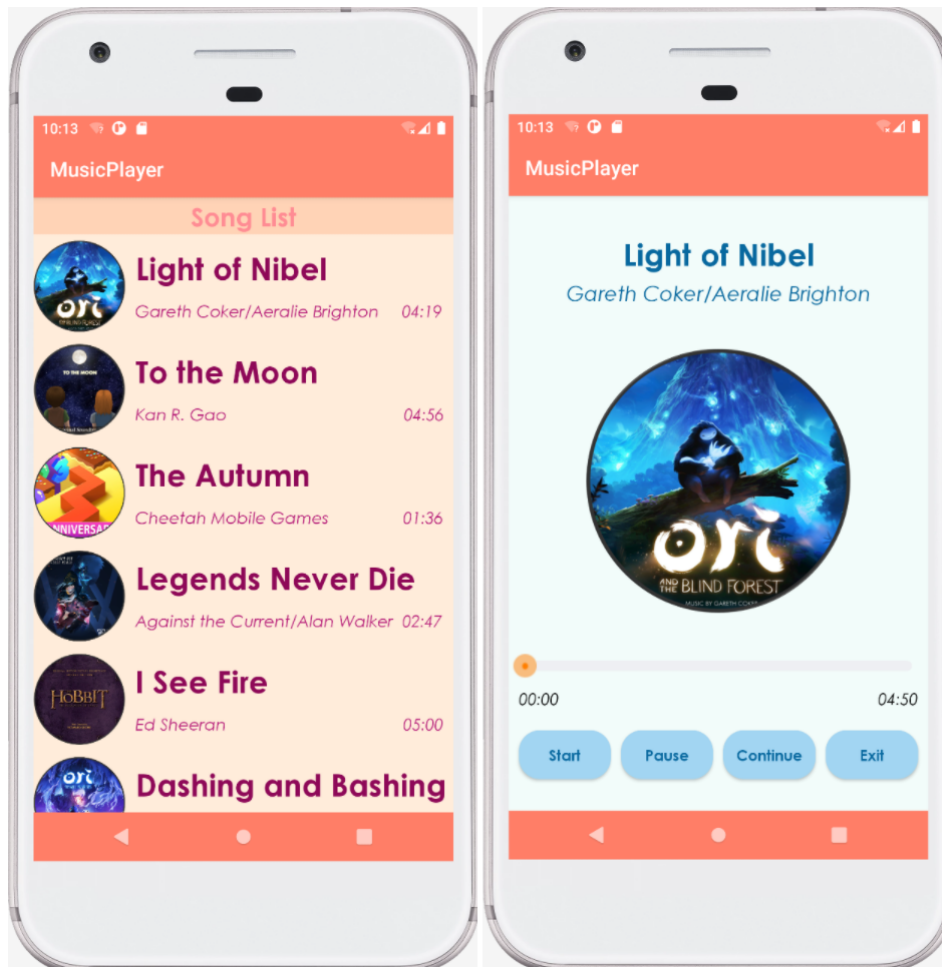
1 <?xml version="1.0" encoding="utf-8"?>
2 <shape xmlns:android="http://schemas.android.com/apk/res/android"
3     android:padding="16dp"
4     android:shape="rectangle">
5     <solid android:color="#a2d5f2" />
6     <stroke
7         android:width="1dp"
8         android:color="#a2d5f2" />
9     <corners android:radius="20dp" />
10 </shape>

```

The button design is shown as the following.



Finally, I adapted the size of the buttons to display perfectly on my virtual device *Pixel API 30*. The presentation of the user interface is given below.



Part 3 - Implementation of Music Player Function

In Part 3, I will show you how I implemented the operations and functions for the music player. This part will be divided into 2 subsections: Button Functions and Page Jumping.

3.1 - Button Functions

The codes related to button functions:

```
1  @Override
2  public void onClick(View view) {
3      switch (view.getId()){
4          case R.id.btn_play:
5              //播放音乐
6              control.play(song_index);
7              //光盘开始转
8              animator.start();
9              break;
10         case R.id.btn_pause:
11             //停止播放音乐
12             control.pausePlay();
13             //光盘停止转
```

```

14         animator.pause();
15         break;
16     case R.id.btn_continue:
17         //继续播放音乐
18         control.continuePlay();
19         //光盘继续转
20         animator.resume();
21         break;
22     case R.id.btn_exit:
23         finish();
24         break;
25     }
26 }

```

3.2 - Page Jumping

The related codes:

```

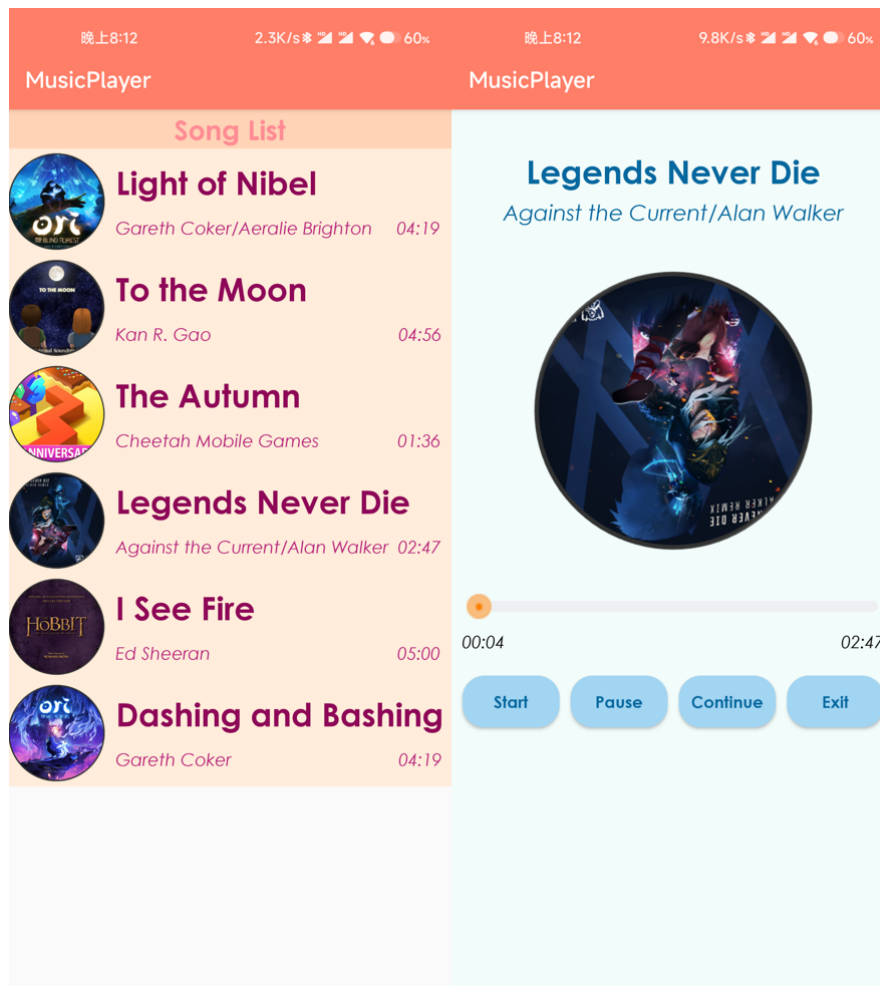
1  @Override
2  //给每个itemview赋具体的内容
3  public void onBindViewHolder(@NonNull MyHolder holder,
4  @SuppressWarnings("Recyclerview") int position) {
5      holder.miv_image.setBackgroundResource(images[position]);
6      holder.mtv_name.setText(names[position]);
7      holder.mtv_singer.setText(singers[position]);
8
9      holder.itemView.setOnClickListener(new View.OnClickListener() {
10         @Override
11         public void onClick(View view) {
12             Intent intent = null;
13             song_index = position;
14             intent=new Intent(MainActivity.this,SongActivity1.class);
15             startActivity(intent);
16         }
17     });
18 }

```

To be mentioned, the real device function presentation will be recorded in a video. It will be attached in the assignment submission file.

Part 4 - Real Device Presentation

The following tests are based on Mi 11, MIUI 13 system. Due to resolution problems, there may exists a blank area on the bottom. But all the buttons and functions can be executed correctly.



For more detailed contents, please refer to the attached presentation video.

Appendix - All Codes

Java program for song list MainActivity.java:

```

1  package com.example.musicplayer;
2
3  import androidx.annotation.NonNull;
4  import androidx.appcompat.app.AppCompatActivity;
5  import androidx.recyclerview.widget.LinearLayoutManager;
6  import androidx.recyclerview.widget.RecyclerView;
7  import androidx.recyclerview.widget.StaggeredGridLayoutManager;
8
9  import android.annotation.SuppressLint;
10 import android.content.Intent;
11 import android.graphics.Rect;
12 import android.os.Bundle;
13 import android.os.Handler;
14 import android.provider.ContactsContract;
15 import android.view.View;
16 import android.view.ViewGroup;
17 import android.widget.ImageView;
18 import android.widget.TextView;
19 import android.widget.Toast;
20
21 import java.util.zip.Inflater;

```



```

22
23 public class MainActivity extends AppCompatActivity {
24     private RecyclerView mrv;
25     public static int song_index = 0;
26
27     private int[] images=
    {R.drawable.cover,R.drawable.cover2,R.drawable.cover3,R.drawable.cover4,R.d
    rawable.cover5,R.drawable.cover6};
28     private String[] names={"Light of Nibel","To the Moon","The
    Autumn","Legends Never Die","I See Fire","Dashing and Bashing"};
29     private String[] singers={"Gareth Coker/Aeralie Brighton    04:19",
30                               "Kan R. Gao
    04:56",
31                               "Cheetah Mobile Games                01:36",
32                               "Against the Current/Alan Walker  02:47",
33                               "Ed Sheeran
    05:00",
34                               "Gareth Coker
    04:19"};
35
36     @Override
37     protected void onCreate(Bundle savedInstanceState) {
38         super.onCreate(savedInstanceState);
39         setContentView(R.layout.activity_main);
40         mrv = findViewById(R.id.rv);
41         mrv.setLayoutManager(new
    LinearLayoutManager(MainActivity.this,LinearLayoutManager.VERTICAL,false));
42         mrv.setAdapter(new MyAdapter());
43         mrv.addItemDecoration(new MyDecoration());
44
45     }
46
47
48     private class MyAdapter extends RecyclerView.Adapter<MyHolder> {
49         @NonNull
50         @Override
51         //把item_layout布局转成视图
52         public MyHolder onCreateViewHolder(@NonNull ViewGroup parent, int
    viewType) {
53             View v =
    View.inflate(MainActivity.this,R.layout.item_layout,null);
54             MyHolder myHolder = new MyHolder(v);
55             return myHolder;
56         }
57
58         @Override
59         //给每个itemView赋具体的内容
60         public void onBindViewHolder(@NonNull MyHolder holder,
    @SuppressWarnings("RecyclerView") int position) {
61             holder.miv_image.setBackgroundResource(images[position]);
62             holder.mtv_name.setText(names[position]);
63             holder.mtv_singer.setText(singers[position]);
64
65             holder.itemView.setOnClickListener(new View.OnClickListener() {
66                 @Override

```

```

67         public void onClick(View view) {
68             Intent intent = null;
69             song_index = position;
70             intent=new
Intent(MainActivity.this,SongActivity1.class);
71             startActivity(intent);
72         }
73     });
74
75     }
76
77     @Override
78     public int getItemCount() {
79         return images.length;
80     }
81 }
82
83 private class MyHolder extends RecyclerView.ViewHolder {
84     private ImageView miv_image;
85     private TextView mtv_name,mtv_info,mtv_singer;
86     public MyHolder(@NonNull View itemView) {
87         super(itemView);
88         miv_image = itemView.findViewById(R.id.iv_image);
89         mtv_singer = itemView.findViewById(R.id.tv_singer);
90         mtv_name = itemView.findViewById(R.id.tv_name);
91     }
92 }
93 //自定义了一个ItemDecoration类
94 private class MyDecoration extends RecyclerView.ItemDecoration {
95     @Override
96     public void getItemOffsets(@NonNull Rect outRect, @NonNull View
view, @NonNull RecyclerView parent, @NonNull RecyclerView.State state) {
97         super.getItemOffsets(outRect, view, parent, state);
98         outRect.set(0,15,0,15);
99     }
100 }
101 }

```

Java program for song playing interface `SongActivity1.java`:

```

1  package com.example.musicplayer;
2
3  import static com.example.musicplayer.MainActivity.song_index;
4
5  import androidx.annotation.NonNull;
6  import androidx.appcompat.app.AppCompatActivity;
7
8  import android.animation.ObjectAnimator;
9  import android.content.ComponentName;
10 import android.content.Intent;
11 import android.content.ServiceConnection;
12 import android.media.MediaPlayer;
13 import android.os.Bundle;

```

```

14 import android.os.Handler;
15 import android.os.HandlerThread;
16 import android.os.IBinder;
17 import android.os.Looper;
18 import android.os.Message;
19 import android.view.View;
20 import android.view.animation.LinearInterpolator;
21 import android.widget.Button;
22 import android.widget.ImageView;
23 import android.widget.SeekBar;
24 import android.widget.TextView;
25
26 import com.example.musicplayer.MusicService;
27
28 import java.util.Timer;
29
30 public class SongActivity1 extends AppCompatActivity{
31     private ImageView iv_cover;
32     private static SeekBar sb;
33     private static TextView tv_progress,tv_total, tv_name, tv_singer;
34     private Button btn_play,btn_pause,btn_continue,btn_exit;
35
36     public String[] name = {"Light of Nibel","To the Moon","The
Autumn","Legends Never Die","I See Fire","Dashing and Bashing"};
37     public String[] singer = {"Gareth Coker/Aeralie Brighton","Kan R.
Gao","Cheetah Mobile Games","Against the Current/Alan Walker","Ed
Sheeran","Gareth Coker"};
38
39     private ObjectAnimator animator; //声明一个动画组件ObjectAnimator
40
41     private MusicService.MusicControl control;//声明MusicService中的音乐控制器
42
43     private ServiceConnection connection = new ServiceConnection() { //声明
服务连接
44         @Override
45         public void onServiceConnected(ComponentName componentName, IBinder
iBinder) {
46             control = (MusicService.MusicControl) iBinder;//实例化control。
47         }
48         @Override
49         public void onServiceDisconnected(ComponentName componentName) {
50
51         }
52     };
53
54
55     @Override
56     protected void onCreate(Bundle savedInstanceState) {
57         super.onCreate(savedInstanceState);
58         setContentView(R.layout.activity_song1);
59         init();
60     }
61     public void init(){
62         iv_cover = findViewById(R.id.iv_cover);
63         sb = findViewById(R.id.sb);

```

```

64         tv_progress = findViewById(R.id.tv_progress);
65         tv_total = findViewById(R.id.tv_total);
66         tv_name = findViewById(R.id.name);
67         tv_singer = findViewById(R.id.singer);
68
69         btn_play = findViewById(R.id.btn_play);
70         btn_pause = findViewById(R.id.btn_pause);
71         btn_continue = findViewById(R.id.btn_continue);
72         btn_exit = findViewById(R.id.btn_exit);
73
74         OnClickListener monclick = new OnClickListener();
75         btn_play.setOnClickListener(monclick);
76         btn_pause.setOnClickListener(monclick);
77         btn_continue.setOnClickListener(monclick);
78         btn_exit.setOnClickListener(monclick);
79
80         //执行动画的对象是iv_cover, // 动画效果是0-360°旋转（用的是浮点数，所以加个
81         f）。
82         animator = ObjectAnimator.ofFloat(iv_cover, "rotation", 0.0f, 360.0f);
83         animator.setDuration(10000); //旋转一周的时长，单位是毫秒，此处设置了10s
84         animator.setInterpolator(new LinearInterpolator()); //设置匀速转动
85         animator.setRepeatCount(-1); //设置循环，此处设置的是无限循环。如果是正值，
86         意味着转动多少圈。
87
88         //声明一个意图，该意图进行服务的启动，意思是将MusicService里面的服务要传到主程
89         序这里来。
90         Intent mintent = new Intent(SongActivity1.this, MusicService.class);
91         bindService(mintent, connection, BIND_AUTO_CREATE); //建立意图中
92         MainActivity与MusicService两对象的服务连接
93
94         switch (song_index){
95             case 0:
96                 iv_cover.setImageResource(R.drawable.cover);
97                 break;
98             case 1:
99                 iv_cover.setImageResource(R.drawable.cover2);
100                break;
101             case 2:
102                 iv_cover.setImageResource(R.drawable.cover3);
103                 break;
104             case 3:
105                 iv_cover.setImageResource(R.drawable.cover4);
106                 break;
107             case 4:
108                 iv_cover.setImageResource(R.drawable.cover5);
109                 break;
110         }
111
112         tv_name.setText(name[song_index]);
113         tv_singer.setText(singer[song_index]);
114         seekBarListener msbListener = new seekBarListener();
115         sb.setOnSeekBarChangeListener(msbListener);
116
117     }
118
119     // 设置播放、暂停、继续和退出按钮的监听（或点击）事件

```

```

115     class OnClickListener implements View.OnClickListener{
116
117         @Override
118         public void onClick(View view) {
119             switch (view.getId()){
120                 case R.id.btn_play:
121                     //播放音乐
122                     control.play(song_index);
123                     //光盘开始转
124                     animator.start();
125                     break;
126                 case R.id.btn_pause:
127                     //停止播放音乐
128                     control.pausePlay();
129                     //光盘停止转
130                     animator.pause();
131                     break;
132                 case R.id.btn_continue:
133                     //继续播放音乐
134                     control.continuePlay();
135                     //光盘继续转
136                     animator.resume();
137                     break;
138                 case R.id.btn_exit:
139                     finish();
140                     break;
141             }
142         }
143     }
144 }
145
146 @Override
147 protected void onDestroy() {
148     control.stopPlay();
149     unbindService(connection);
150     super.onDestroy();
151 }
152
153 //Handler主要用于异步消息的处理，在这里是处理子线程MusicService传来的消息
154 public static Handler handler = new Handler(Looper.getMainLooper()){
155
156
157     @Override
158     public void handleMessage(@NonNull Message msg) {
159         //super.handleMessage(msg);
160         Bundle bundle = msg.getData();
161         int duration = bundle.getInt("duration");//把音乐时长放在bundle里
162         int currentDuration = bundle.getInt("currentDuration");//把音乐
        当前播放时长放在bundle里
163
164         sb.setMax(duration);
165         sb.setProgress(currentDuration);
166
167         //显示总时长
168         int minite = duration / 1000 /60;

```

```

169         int second = duration / 1000 % 60;
170         String strMinite = "";
171         String strSecond = "";
172         if (minite < 10){
173             strMinite = "0" +minite;
174         }else {
175             strMinite = minite + "";
176         }
177         if (second < 10){
178             strSecond = "0" + second;
179         }else {
180             strSecond = second + "";
181         }
182         tv_total.setText(strMinite + ":" + strSecond);
183
184
185         //显示播放时长
186         minite = currentDuration / 1000 /60;
187         second = currentDuration / 1000 % 60;
188
189         if (minite < 10){
190             strMinite = "0" +minite;
191         }else {
192             strMinite = minite + "";
193         }
194         if (second < 10){
195             strSecond = "0" + second;
196         }else {
197             strSecond = second + "";
198         }
199         tv_progress.setText(strMinite + ":" + strSecond);
200     }
201 };
202
203 //给进度条设置监听
204 class seekBarListener implements SeekBar.OnSeekBarChangeListener {
205     @Override
206     //进度条行进过程的监听
207     public void onProgressChanged(SeekBar seekBar, int i, boolean b) {
208         if (i == seekBar.getMax()){
209             animator.pause();
210         }
211         if (b){//判断是否来自用户
212             control.seekTo(i);
213         }
214     }
215
216     @Override
217     //用户开始滑动进度条的监听
218     public void onStartTrackingTouch(SeekBar seekBar) {
219         control.pausePlay();
220         animator.pause();
221     }
222
223     @Override

```

```

224         //用户停止滑动进度条的监听
225         public void onStopTrackingTouch(SeekBar seekBar) {
226             control.continuePlay();
227             animator.resume();
228         }
229     }
230 }

```

Music Service `MusicService.java`:

```

1  package com.example.musicplayer;
2
3  import android.app.Service;
4  import android.content.Intent;
5  import android.media.MediaPlayer;
6  import android.os.Binder;
7  import android.os.Build;
8  import android.os.Bundle;
9  import android.os.IBinder;
10 import android.os.Message;
11
12 import androidx.annotation.Nullable;
13
14 import java.util.Timer;
15 import java.util.TimerTask;
16
17 /**
18  * 在这里设置音乐播放功能的服务
19  *
20  */
21
22 public class MusicService extends Service {
23
24     // 设置两个成员变量
25     private MediaPlayer player;//多媒体对象
26     private Timer timer;//时钟
27     private int[] songs = {R.raw.song1, R.raw.song2, R.raw.song3,
28         R.raw.song4, R.raw.song5, R.raw.song6};
29
30     public MusicService(){};//一个空的构造函数（为什么放？）
31
32     @Nullable
33     @Override
34     public IBinder onBind(Intent intent) {
35         return new MusicControl();//这样的话，绑定服务的时候，可以把音乐控制器实例
36         化。
37     }
38
39     @Override
40     public void onCreate() {
41         super.onCreate();
42         player = new MediaPlayer();//实例化多媒体
43     }
44 }

```

```

42
43     @Override
44     public int onStartCommand(Intent intent, int flags, int startId) {
45         return super.onStartCommand(intent, flags, startId);
46     }
47
48     @Override
49     public void onDestroy() {
50         super.onDestroy();
51     }
52     //创建一个内部类MusicControl，功能是让主程序控制sevice里面的多媒体对象。IBinder
    是Binder的子类，因此要返回MusicControl给IBinder。
53     class MusicControl extends Binder{
54         public void play(int index) {
55             try{
56                 player.reset();//重置音乐播放器
57                 player = MediaPlayer.create(getApplicationContext(),
songs[index]); //加载多媒体文件
58                 player.start(); //开始播放音乐
59                 addTimer();//添加计时器
60             }catch (Exception exception) { //catch用来处理播放时产生的异常
61                 exception.printStackTrace();
62             }
63         }
64         public void pausePlay(){
65             player.pause();    //暂停播放
66         }
67         public void continuePlay(){
68             player.start();    //继续播放
69         }
70         public void stopPlay(){
71             player.stop();
72             player.release();
73             try {
74                 timer.cancel();
75             }catch (Exception e){
76                 e.printStackTrace();
77             }
78         }
79         public void seekTo(int progress){
80             player.seekTo(progress);//设置播放位置播放
81         }
82     }
83
84     //添加计时器，计时器是一个多线程的东西，用于设置音乐播放器中的进度条信息
85     public void addTimer(){
86         if (timer == null){
87             timer = new Timer();
88             TimerTask task = new TimerTask() {
89                 @Override
90                 public void run() { //run就是多线程的一个东西
91                     if (player == null) return; //如果player没有实例化，就退
    出。
92                     int duration = player.getDuration();//获取歌曲总长度
93                     int currentDuration = player.getCurrentPosition();

```



```
94         //将音乐的总时长、播放时长封装到消息对象中去；
95         Message message =
SongActivity1.handler.obtainMessage();
96         Bundle bundle = new Bundle();
97         bundle.putInt("duration",duration);
98         bundle.putInt("currentDuration",currentDuration);
99         message.setData(bundle);//使用bundle给主线程发消息
100
101         //将消息添加到主线程中
102         SongActivity1.handler.sendMessage(message);
103
104     }
105 };
106 //开始计时任务后5ms，执行第一次任务，以后每500ms执行一次任务
107 timer.schedule(task, 5,500);
108 }
109 }
110 }
```