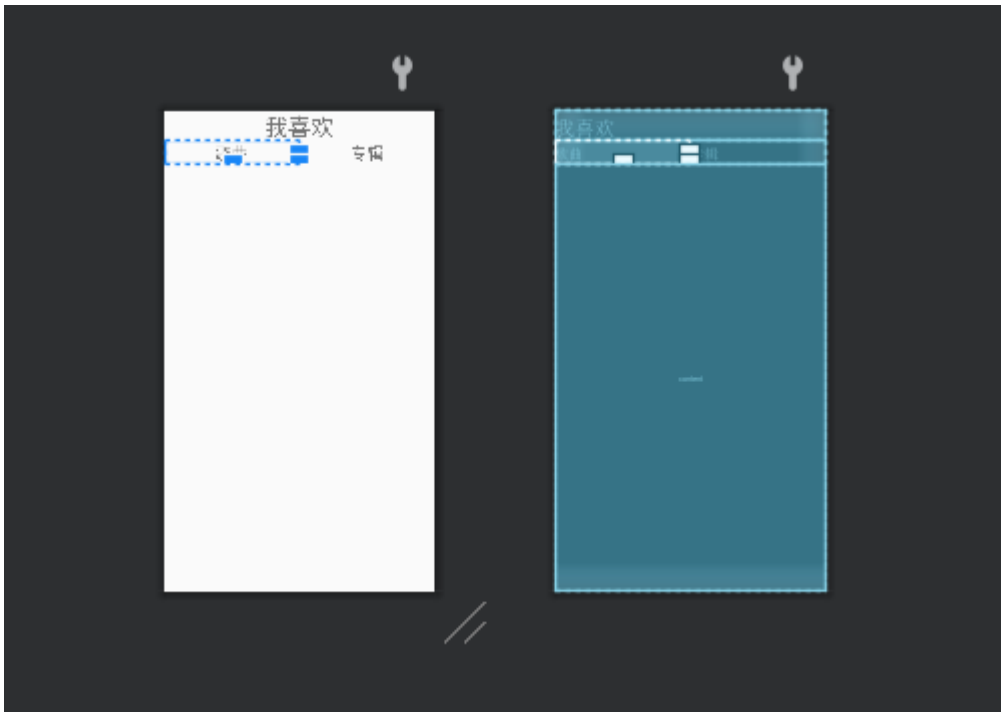
1、首先我们先进行程序运行后的设计，即主程序（MainActivity.java）和主程序页面(activity_main.xml)

Activity_main.xml 的代码如下：

```
<?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"
    tools:context=".MainActivity"
    android:orientation="vertical">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:gravity="center"
            android:text="我喜欢"
            android:textSize="35dp"/>
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
        <TextView
            android:id="@+id/menu1"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:gravity="center"
            android:text="歌曲"
            android:textSize="25dp"
```

```
            />
        <TextView
            android:id="@+id/menu2"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:gravity="center"
            android:text="专辑"
            android:textSize="25dp"
            />
    </LinearLayout>
    <FrameLayout
        android:id="@+id/content"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="9">
    </FrameLayout>

</LinearLayout>
```

页面如下



其中我喜欢和专辑代表的是两个fragment，分别对应着frag1和frag2，点击“我喜欢”显示音乐列表，点击“专辑”显示专辑列表，程序运

行后默认显示音乐列表

MainActivity.java 的代码如下：

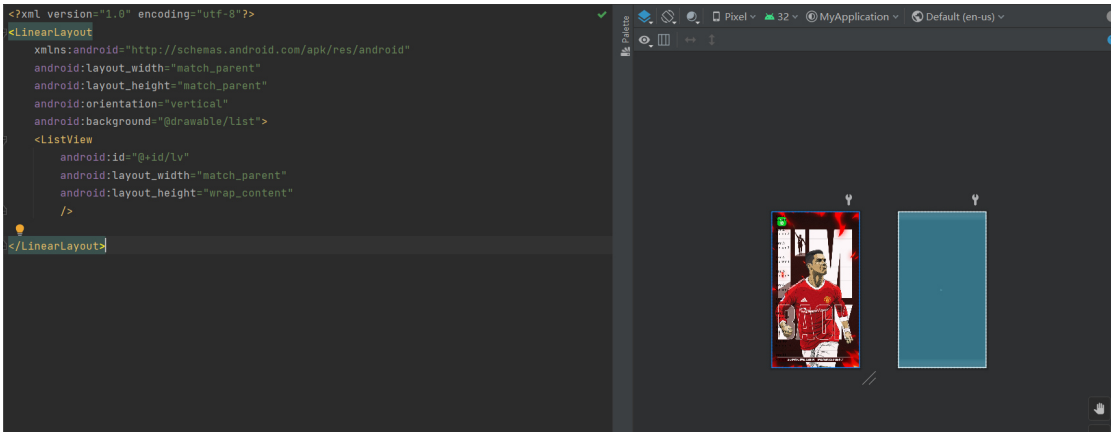
```
package com.example.myapplication;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentTransaction;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import androidx.recyclerview.widget.StaggeredGridLayoutManager;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.graphics.Rect;
import android.nfc.NfcAdapter;
import android.os.Bundle;
import android.os.Handler;
import android.provider.ContactsContract;
import android.provider.Settings;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.zip.Inflater;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {
    private TextView tv1,tv2;
    private FragmentManager fm;
    private FragmentTransaction ft;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //绑定控件
        tv1=findViewById(R.id.menu1);
        tv2=findViewById(R.id.menu2);
```

```
        //设置监听器
        tv1.setOnClickListener(this);
        tv2.setOnClickListener(this);
        //fm 为 Fragment 的管理者，ft 为其改变者
        fm=getSupportFragmentManager();
        //默认显示 frag1
        ft= fm.beginTransaction();
        ft.replace(R.id.content,new frag1());
        ft.commit();
    }

    @Override
    public void onClick(View view) {
        ft= fm.beginTransaction();
        //切换
        switch(view.getId()){
            case R.id.menu1:
                ft.replace(R.id.content,new frag1());
                break;
            case R.id.menu2:
                ft.replace(R.id.content,new frag2());
                break;
            default:
                break;
        }
        ft.commit();
    }
}
```

2、接下来我们进行frag1的设计,frag1显示音乐列表，并设置点击事件，点击歌曲后跳转到MusicActivity中播放该歌曲，与frag1对应的布局文件有music_list和item_layout,music_list获取音乐列表并让音乐垂直排列,item_layout让每首歌曲以图片、

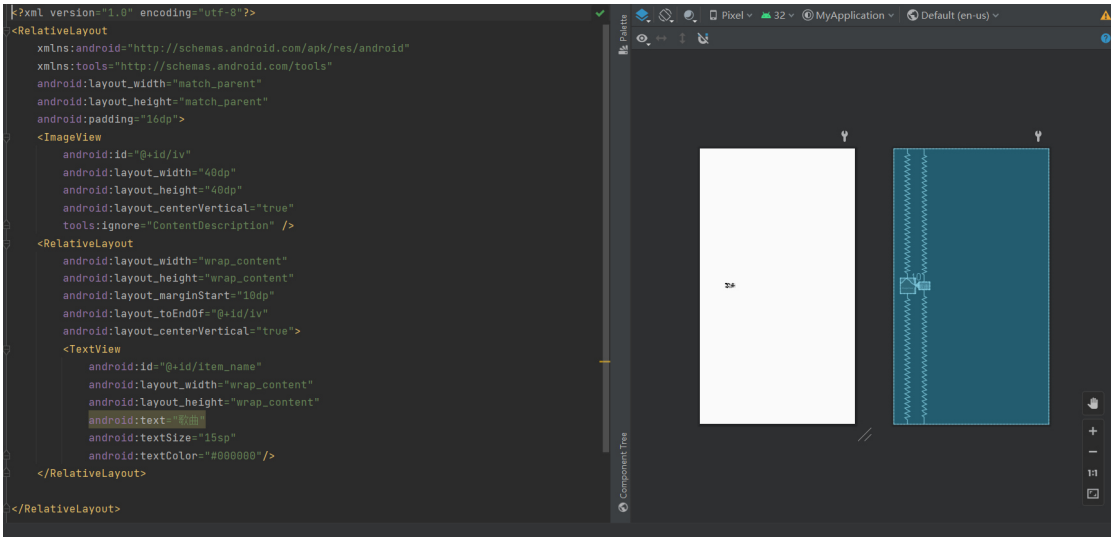
曲名的方式排列

music_list.xml 的代码及页面如下：



```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:orientation="vertical"
    android:background="@drawable/list">
    <ListView
        android:id="@+id/lv"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        />
</LinearLayout>
```

item_layout.xml 的代码及页面如下：



```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp">
    <ImageView
        android:id="@+id/iv"
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:layout_centerVertical="true"
        tools:ignore="ContentDescription" />
    <RelativeLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:layout_toEndOf="@+id/iv"
        android:layout_centerVertical="true">
        <TextView
            android:id="@+id/item_name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="歌曲"
            android:textSize="15sp"
            android:textColor="#000000"/>
    </RelativeLayout>

</RelativeLayout>
```

Frag1.java 的代码如下：

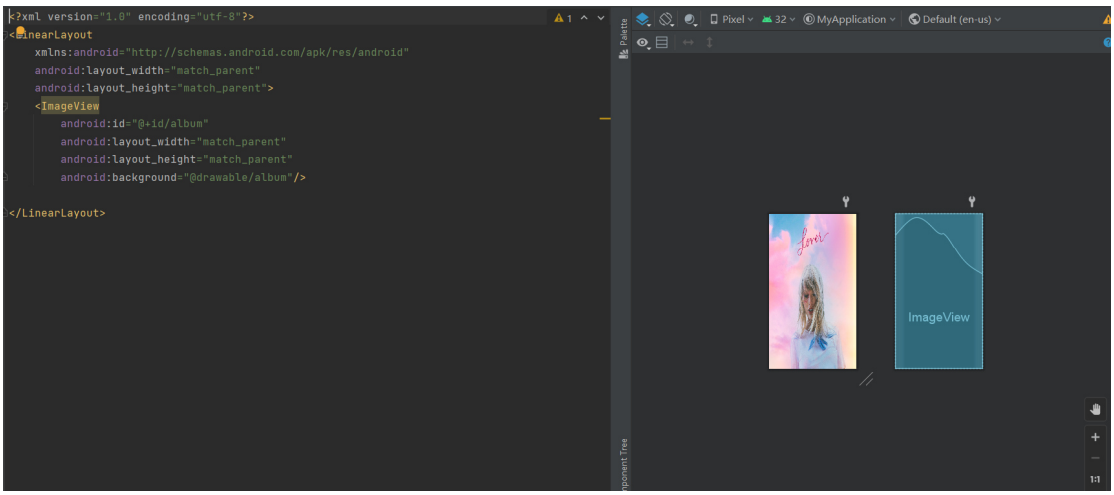
```
package com.example.myapplication;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.BaseAdapter;
```

```
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;

import androidx.fragment.app.Fragment;

public class frag1 extends Fragment {
    private View view;
    //创建歌曲的 String 数组和歌手图片的 int 数组
    public String[] name={"Two Steps From Hell _ Thomas Bergersen - Victory","Wiz Khalifa _ Charlie Puth - See You Again (feat_ Charlie Puth)","鹿先森乐队 - 春风十里"};
    public static int[] icons={R.drawable.music0,R.drawable.music1,R.drawable.music2};



    @Override
    public View onCreateView(final LayoutInflater inflater,ViewGroup container,Bundle savedInstanceState){
        //绑定布局
        view= inflater.inflate(R.layout.music_list,null);
        //创建 listView 列表并且绑定控件
        ListView listView = view.findViewById(R.id.lv);
        //实例化一个适配器
        MyBaseAdapter adapter = new MyBaseAdapter();
        listView.setAdapter(adapter);
        //列表元素的点击监听器
        listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> adapterView, View view, int position, long id) {
                //创建 Intent 对象，参数从 frag1 跳转到 MusicActivity
                Intent intent = new Intent(frag1.this.getContext(),MusicActivity.class);
                //将歌曲名和歌曲的下标存入 Intent 对象
                intent.putExtra("name",name[position]);
                intent.putExtra("position",String.valueOf(position));
                //开始跳转
                startActivity(intent);
            }
        });
```

```
        return view;
    }

    //创建自定义适配器
    class MyBaseAdapter extends BaseAdapter{

        @Override
        public int getCount() {
            return name.length;
        }

        @Override
        public Object getItem(int i) {
            return name[i];
        }

        @Override
        public long getItemId(int i) {
            return i;
        }

        @Override
        public View getView(int i, View convertView, ViewGroup parent) {
            //绑定好 Vlew，然后绑定控件
            @SuppressLint("ViewHolder") View view =
View.inflate(frag1.this.getContext(),R.layout.item_layout,null);
            TextView tv_name = view.findViewById(R.id.item_name);
            ImageView iv=view.findViewById(R.id.iv);
            tv_name.setText(name[i]);
            iv.setImageResource(icons[i]);
            return view;
        }
    }

}
```

3、然后进行 frag2 的设计，与 frag2 对应的布局文件为 frag2_layout.xml，里面放置专辑图片，也可以放置歌手信息等，可自由拓展

frag2_layout.xml 的代码及页面如下：



Frag2.java 的代码如下：

```
ackage com.example.myapplication;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.core.app.BundleCompat;
import androidx.fragment.app.Fragment;

public class frag2 extends Fragment {
    //创建一个 View
    private View album;
    //显示布局
    public View onCreateView(final LayoutInflater inflater, ViewGroup container,
Bundle savedInstanceState){
        album=inflater.inflate(R.layout.frag2_layout,null);
        return album;
    }
}
```

4、最后进行播放服务（MusicService.java）、播放页面

（activity_music.xml）、播放程序（MusicActivity.java）的设计

播放页面含有背景图片、转动光盘、歌名、进度条、当前播放时长、总时长、上一首、暂停、播放、下一首、返回

activity_music.xml 的代码如下：

```
<?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:background="@drawable/background"
    tools:context=".MusicActivity"
    android:gravity="center"
    android:orientation="vertical">
    <ImageView
        android:id="@+id/iv_music"
        android:layout_width="240dp"
        android:layout_height="240dp"
        android:layout_gravity="center_horizontal"
        android:layout_margin="15dp"
        android:src="@drawable/music0"/>
    <TextView
        android:id="@+id/song_name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="歌曲名"
        android:textSize="20sp"
        />
    <SeekBar
        android:id="@+id/sb"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

    <RelativeLayout
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:paddingLeft="8dp"
        android:paddingRight="8dp">

        <TextView
            android:id="@+id/tv_progress"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="00:00" />

        <TextView
            android:id="@+id/tv_total"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentRight="true"
            android:text="00:00" />
    </RelativeLayout>
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <Button
            android:id="@+id/btn_continue_play"
            android:layout_width="80dp"
            android:layout_height="80dp"
            android:layout_centerHorizontal="true"
            android:background="@drawable/play"/>

        <Button
            android:id="@+id/btn_pause"
            android:layout_width="80dp"
            android:layout_height="80dp"
            android:layout_centerHorizontal="true"
            android:background="@drawable/pause" />
        <Button
            android:id="@+id/btn_play"
            android:layout_width="80dp"
            android:layout_height="80dp"
            android:layout_centerHorizontal="true"
            android:background="@drawable/play"/>
        <Button
            android:id="@+id/btn_last"
            android:layout_width="60dp"
            android:layout_height="60dp"
            android:layout_centerVertical="true"
```

```
            android:layout_marginRight="10dp"
            android:layout_toLeftOf="@id/btn_play"
            android:background="@drawable/last"/>
        <Button
            android:id="@+id/btn_next"
            android:layout_width="60dp"
            android:layout_height="60dp"
            android:background="@drawable/next"
            android:layout_centerVertical="true"
            android:layout_marginLeft="10dp"
            android:layout_toRightOf="@id/btn_play"
            />
        <Button
            android:id="@+id/btn_exit"
            android:layout_width="60dp"
            android:layout_height="60dp"
            android:background="@drawable/exit"
            android:layout_centerVertical="true"
            android:layout_marginLeft="30dp"
            android:layout_toRightOf="@id/btn_next"/>
    </RelativeLayout>

</LinearLayout>
```

页面如下：

播放服务中设置计时器、MusicControl

MusicService.java 的代码如下：

```
package com.example.myapplication;

import android.app.Service;
import android.content.Intent;
import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Binder;
import android.os.Bundle;
import android.os.IBinder;
import android.os.Message;

import java.util.Timer;
import java.util.TimerTask;

public class MusicService extends Service {
    //声明一个 MediaPlayer 引用
    private MediaPlayer player;
    //声明一个计时器
    private Timer timer;
    public MusicService() {
    }

    @Override
    public IBinder onBind(Intent intent) {
        // TODO: Return the communication channel to the service.
        return new MusicControl();
    }
    @Override
    public void onCreate(){
        super.onCreate();
        //创建音乐播放器对象
        player=new MediaPlayer();
    }

    //添加计时器用于设置音乐播放器中的播放进度条
    public void addTimer(){
        //如果计时器不存在，也就是没有引用实例
        if (timer==null){
            //创建计时器对象
            timer=new Timer();
            TimerTask task=new TimerTask() {
                @Override
                public void run() {
                    if (player==null)
```

```
                    return;
                else {
                    //获取歌曲总时长
                    int duration = player.getDuration();
                    //获取歌曲当前播放时长
                    int currentPosition = player.getCurrentPosition();
                    //创建消息对象
                    Message message = MusicActivity.handler.obtainMessage();
                    //将音乐总时长和播放进度封装至 bundle 中
                    Bundle bundle = new Bundle();
                    bundle.putInt("duration",duration);
                    bundle.putInt("currentPosition",currentPosition);
                    //再将 bundle 封装到 message 消息对象中
                    message.setData(bundle);
                    MusicActivity.handler.sendMessage(message);
                }
            }
        };
        //开始即使任务后的 5 毫秒，第一次执行 task 任务，以后 1000 毫秒（1s）执
  行一次
        timer.schedule(task,5,1000);
    }
}

//创建一个内部类 MusicControl，功能是让主程序控制 service 里面的多媒体对象。
  IBinder 是 Binder 的子类，因此要返回 MusicControl 给 IBinder。
class MusicControl extends Binder{
    public void play(int i){
        //下面这步有点奇怪
        Uri                                                                    uri
  =Uri.parse("android.resource://"+getPackageName()+"/raw/"+"music"+i);
        try {
            //重置音乐播放器
            player.reset();
            //加载多媒体文件
            player=MediaPlayer.create(getApplicationContext(),uri);
            //播放音乐
            player.start();
            //添加计时器
            addTimer();
        }catch (Exception exception){
            exception.printStackTrace();
        }
    }
```

```
        //暂停播放音乐
        public void pausePlay(){
            player.pause();
        }
        //继续播放音乐
        public void continuePlay(){
            player.start();
        }
        //设置音乐播放位置
        public void    seekTo(int progress){
            player.seekTo(progress);
        }
    }
    //停止播放并销毁
    @Override
    public void onDestroy(){
        super.onDestroy();
        if (player==null){
            return;
        }
        if (player.isPlaying()){
            player.stop();//停止播放音乐
        }
        player.release();//释放占用资源
        player=null;
        try {
            timer.cancel();//计时器归零
        }catch (Exception exception){
          exception.printStackTrace();
        }
    }
}
```

播放程序中设置按钮点击事件、光盘转动事件、进度条监听事件和服务连接事件

MusicActivity.java 的代码如下：

```
package com.example.myapplication;

import static java.lang.Integer.min;
import static java.lang.Integer.parseInt;

import androidx.annotation.NonNull;
```

```
import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AppCompatActivity;

import android.animation.ObjectAnimator;
import android.content.ComponentName;
import android.content.Intent;
import android.content.ServiceConnection;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.IBinder;
import android.os.Looper;
import android.os.Message;
import android.view.View;
import android.view.animation.LinearInterpolator;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.SeekBar;
import android.widget.TextView;

@RequiresApi(api = Build.VERSION_CODES.KITKAT)
public class MusicActivity extends AppCompatActivity implements View.OnClickListener{

    //定义歌名名称的数组
    public String[] musicName={"Two Steps From Hell _ Thomas Bergersen - Victory","Wiz
       Khalifa _ Charlie Puth - See You Again (feat_ Charlie Puth)",
                                        "鹿先森乐队 - 春风十里"};
    private static SeekBar sb;//定义进度条
    //定义开始和总时长，歌名
    private static TextView tv_progress,tv_total,name_song;
    //定义旋转的动画
    private ObjectAnimator animator;
    //声明 MusicService 中的音乐控制器
    private MusicService.MusicControl musicControl;

    //定义各个按钮
    private Button play;
    private Button pause;
    private Button continue_play;
    private Button last;
    private Button next;
    private Button exit;
    private ImageView iv_music;//定义歌曲图片框
```

```
Intent intent1,intent2;//定义两个意图
MyServiceConn myServiceConn;
private boolean isUnbind = false;//记录服务器是否被解绑
public int change = 0;//记录下标的变化值，便于之后的上一首下一首操作

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_music);
    //获得意图
    intent1 = getIntent();
    //初始化控件
    init();
}

private void init(){
    //依次绑定控件
    tv_progress=findViewById(R.id.tv_progress);
    tv_total=findViewById(R.id.tv_total);
    sb=findViewById(R.id.sb);
    name_song=findViewById(R.id.song_name);
    iv_music=findViewById(R.id.iv_music);

    play=findViewById(R.id.btn_play);
    pause=findViewById(R.id.btn_pause);
    continue_play=findViewById(R.id.btn_continue_play);
    last=findViewById(R.id.btn_last);
    next=findViewById(R.id.btn_next);
    exit=findViewById(R.id.btn_exit);

    //设置监听事件
    play.setOnClickListener(this);
    pause.setOnClickListener(this);
    continue_play.setOnClickListener(this);
    last.setOnClickListener(this);
    next.setOnClickListener(this);
    exit.setOnClickListener(this);

    //创建意图对象
    intent2=new Intent(this,MusicService.class);
    myServiceConn=new MyServiceConn();//创建服务连接对象
    bindService(intent2,myServiceConn,BIND_AUTO_CREATE);//绑定服务

    //从歌曲列表传过来的歌曲名
```

```
String name = intent1.getStringExtra("name");
//设置歌曲显示
name_song.setText(name);
//定义歌曲列表传过来的下标 position
String position = intent1.getStringExtra("position");
//将字符串转化为整形 i
int i =parseInt(position);
//图像框设置为 frag1 里面的图标数组，下标为 i
iv_music.setImageResource(frag1.icons[i]);
//为滑动条设置监听事件
sb.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
    @RequiresApi(api = Build.VERSION_CODES.KITKAT)
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
        //当滑动条滑到末端时，自动播放下一首
        if(progress==seekBar.getMax()){
            change++;//自动下一首，下标加一
            String position = intent1.getStringExtra("position");
            int nowPosition = (parseInt(position)+change)% musicName.length;//当前歌曲下标
            iv_music.setImageResource(frag1.icons[nowPosition]);// 切换到 nowPosition 这首歌的封面
            name_song.setText(musicName[nowPosition]);//切换到 nowPosition 这首歌的歌名
            musicControl.play(nowPosition);//播放 nowPosition 这首歌
            musicControl.seekTo(0);//重置播放进度
            //pause.setVisibility(View.VISIBLE);
        }

    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {//滑动条开始滑动时调用
        musicControl.pausePlay();
        animator.pause();
    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {//滑动条停止滑动时调用
        //根据滑动条的进度改变音乐播放进度
        int progress = seekBar.getProgress();
        //切换到当前播放进度
        musicControl.seekTo(progress);
```

```
                musicControl.continuePlay();
                animator.resume();
            }
        });

        //执行动画的对象是 iv_music，// 动画效果是 0-360°旋转（用的是浮点数，所以加个 f）。
        animator=ObjectAnimator.ofFloat(iv_music,"rotation",0.0f,360.0f);
        animator.setDuration(10000);//动画旋转一周的时间为 10000 毫秒
        animator.setInterpolator(new LinearInterpolator());//匀速转动
        animator.setRepeatCount(-1);//无限循环
    }

    //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
    public static Handler handler = new Handler(Looper.getMainLooper()){
        //从主线程中处理从子进程发送过来的消息
        @Override
        public void handleMessage(@NonNull Message message){
            //获取从子线程中发送过来的音乐播放进度
            Bundle bundle = message.getData();
            int duration = bundle.getInt("duration");
            int currentPosition = bundle.getInt("currentPosition");
            sb.setMax(duration);
            sb.setProgress(currentPosition);

            //歌曲总时长，单位为毫秒
            int minute=duration/1000/60;
            int second=duration/1000%60;
            String strMinute=null;
            String strSecond=null;
            if (minute<10){
                strMinute="0"+minute;
            }
            else {
                strMinute=minute+"";
            }
            if (second<10){
                strSecond="0"+second;
            }
            else {
                strSecond=second+"";
            }
            tv_total.setText(strMinute+":"+strSecond);
```

```
            //歌曲当前播放时长
            minute = currentPosition / 1000 /60;
            second = currentPosition / 1000 % 60;

            if (minute<10){
                strMinute="0"+minute;
            }
            else {
                strMinute=minute+"";
            }
            if (second<10){
                strSecond="0"+second;
            }
            else {
                strSecond=second+"";
            }
            tv_progress.setText(strMinute+":"+strSecond);
        }
};

//设置点击事件
@RequiresApi(api = Build.VERSION_CODES.KITKAT)//迷惑
@Override
public void onClick(View view) {
    String position = intent1.getStringExtra("position");
    //将字符串转为整数
    int i=parseInt(position);
    switch(view.getId()){
        case R.id.btn_play:
            play.setVisibility(View.INVISIBLE);
            pause.setVisibility(View.VISIBLE);
            continue_play.setVisibility(View.INVISIBLE);
            musicControl.play(i);
            animator.start();
            break;
        case R.id.btn_last:
            if ((i+change)<1){
                change= musicName.length-1-i;
                musicControl.play(i+change);
                iv_music.setImageResource(frag1.icons[i+change]);
                name_song.setText(musicName[i+change]);
                play.setVisibility(View.INVISIBLE);
                pause.setVisibility(View.VISIBLE);
                continue_play.setVisibility(View.INVISIBLE);
```

```
                animator.start();
                break;
            }
            else{
                change--;
                musicControl.play(i+change);
                iv_music.setImageResource(frag1.icons[i+change]);
                name_song.setText(musicName[i+change]);
                play.setVisibility(View.INVISIBLE);
                pause.setVisibility(View.VISIBLE);
                continue_play.setVisibility(View.INVISIBLE);
                animator.start();
                break;
            }
        case R.id.btn_next:
            if ((i+change)== musicName.length-1){
                change=-i;
                musicControl.play(i+change);
                iv_music.setImageResource(frag1.icons[i+change]);
                name_song.setText(musicName[i+change]);
                play.setVisibility(View.INVISIBLE);
                pause.setVisibility(View.VISIBLE);
                continue_play.setVisibility(View.INVISIBLE);
                animator.start();
                break;
            }
            else{
                change++;
                iv_music.setImageResource(frag1.icons[i+change]);
                name_song.setText(musicName[i+change]);
                musicControl.play(i+change);
                play.setVisibility(View.INVISIBLE);
                pause.setVisibility(View.VISIBLE);
                continue_play.setVisibility(View.INVISIBLE);
                animator.start();
                break;
            }
        case R.id.btn_pause:
            pause.setVisibility(View.INVISIBLE);
            continue_play.setVisibility(View.VISIBLE);
            play.setVisibility(View.INVISIBLE);
            musicControl.pausePlay();
            animator.pause();
            break;
```

```
                case R.id.btn_continue_play:
                    play.setVisibility(View.INVISIBLE);
                    continue_play.setVisibility(View.INVISIBLE);
                    pause.setVisibility(View.VISIBLE);
                    musicControl.continuePlay();
                    animator.resume();
                    break;
                case R.id.btn_exit:
                    unbind(isUnbind);
                    isUnbind=true;
                    finish();
                    break;
            }
        }

        //用于实现连接服务
        class MyServiceConn implements ServiceConnection{

            @Override
            public void onServiceConnected(ComponentName componentName, IBinder
              service) {
                musicControl=(MusicService.MusicControl) service;
            }

            @Override
            public void onServiceDisconnected(ComponentName componentName) {

            }
        }

        //未解绑则解绑
        private void unbind(boolean isUnbind){
            if (!isUnbind){
                musicControl.pausePlay();
                unbindService(myServiceConn);//解绑服务
            }
        }

        @Override
        protected void onDestroy(){
            super.onDestroy();
            unbind(isUnbind);//解绑服务
        }
    }
```

运行结果将会在录制的视频中呈现，视频为下一个附件