Experiment 2: Music player APP design

12010508华羽霄

Requirements：
1. 使用RecyclerView来得到歌曲列表（不少于5首）。

item_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    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"/>
    <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=""
            android:textSize="15sp"
            android:textColor="#000000"/>
    </RelativeLayout>
</RelativeLayout>
```

2. 实现音乐的播放、暂停、继续、退出功能。

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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: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=""
        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_alignParentEnd="true"
            android:text="00:00"/>
    </RelativeLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
        <Button
            android:id="@+id/btn_play"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_margin="8dp"
            android:layout_weight="1"
            android:background="@drawable/btn_bg_selector"
            android:text="播放"/>
        <Button
            android:id="@+id/btn_pause"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_margin="8dp"
            android:layout_weight="1"
            android:background="@drawable/btn_bg_selector"
            android:text="暂停"/>
        <Button
            android:id="@+id/btn_continue"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_margin="8dp"
            android:layout_weight="1"
            android:background="@drawable/btn_bg_selector"
            android:text="继续"/>
        <Button
            android:id="@+id/btn_exit"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_margin="8dp"
            android:layout_weight="1"
            android:background="@drawable/btn_bg_selector"
            android:text="退出"/>
    </LinearLayout>

</LinearLayout>
```

activity_main.xml

```
package com.example.ver1;

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;
//这是一个Service服务类
public class MusicService extends Service {
    //声明一个MediaPlayer引用
    private MediaPlayer player;
    //声明一个计时器引用
    private Timer timer;
    //构造函数
```

```
    public MusicService() {}
    @Override
    public  IBinder onBind(Intent intent){
        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timer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;
                    int duration=player.getDuration();//获取歌曲总时长
                    int currentPosition=player.getCurrentPosition();//获取播放进度
                    Message msg= MusicActivity.handler.obtainMessage();//创建消息对象
                    //将音乐的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msg消息对象中
                    msg.setData(bundle);
                    //最后将消息发送到主线程的消息队列
                    MusicActivity.handler.sendMessage(msg);
                }
            };
            //开始计时任务后的5毫秒，第一次执行task任务，以后每500毫秒（0.5s）执行一次
            timer.schedule(task,5,500);
        }
    }
    //Binder是一种跨进程的通信方式
    class MusicControl extends Binder{
        public void play(int i){//String path
            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);
                player.start();//播放音乐
                addTimer();//添加计时器
            }catch(Exception e){
                e.printStackTrace();
            }
        }
        //下面的暂停继续和退出方法全部调用的是MediaPlayer自带的方法
        public void pausePlay(){
            player.pause();//暂停播放音乐
        }
        public void continuePlay(){
            player.start();//继续播放音乐
        }
        public void seekTo(int progress){
            player.seekTo(progress);//设置音乐的播放位置
        }
    }
    //销毁多媒体播放器
    @Override
    public void onDestroy(){
        super.onDestroy();
        if(player==null) return;
        if(player.isPlaying()) player.stop();//停止播放音乐
        player.release();//释放占用的资源
        player=null;//将player置为空
    }
}
```

3. 实现进度条的计时功能。

MusicActivity.java

```
//handler机制，可以理解为线程间的通信，我获取到一个信息，然后把这个信息告诉你，就这么简单
    public static Handler handler=new Handler(){//创建消息处理器对象
        //在主线程中处理从子线程发送过来的消息
        @SuppressLint("HandlerLeak")
        @Override
        public void handleMessage(Message msg){
            Bundle bundle=msg.getData();//获取从子线程发送过来的音乐播放进度
            //获取当前进度currentPosition和总时长duration
            int duration=bundle.getInt("duration");
            int currentPosition=bundle.getInt("currentPosition");
            //对进度条进行设置
            sb.setMax(duration);
            sb.setProgress(currentPosition);
            //歌曲是多少分钟多少秒钟
            int minute=duration/1000/60;
            int second=duration/1000%60;
            String strMinute=null;
            String strSecond=null;
            if(minute<10){//如果歌曲的时间中的分钟小于10
                strMinute="0"+minute;//在分钟的前面加一个0
            }else{
                strMinute=minute+"";
            }
            if (second<10){//如果歌曲中的秒钟小于10
                strSecond="0"+second;//在秒钟前面加一个0
            }else{
                strSecond=second+"";
            }
            //这里就显示了歌曲总时长
            tv_total.setText(strMinute+":"+strSecond);
            //歌曲当前播放时长
            minute=currentPosition/1000/60;
            second=currentPosition/1000%60;
            if(minute<10){//如果歌曲的时间中的分钟小于10
                strMinute="0"+minute;//在分钟的前面加一个0
            }else{
                strMinute=minute+" ";
            }
            if (second<10){//如果歌曲中的秒钟小于10
                strSecond="0"+second;//在秒钟前面加一个0
            }else{
                strSecond=second+" ";
            }
            //显示当前歌曲已经播放的时间
            tv_progress.setText(strMinute+":"+strSecond);
        }
    };
```

4. APP应具有页面跳转功能，即可从歌曲列表跳转至音乐播放界面，并在音乐播放界面实现音乐播放的控制。

MainActivity.java

```
package com.example.ver1;

import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentTransaction;

import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {
    //创建需要用到的控件的变量
    private TextView tv;
    private FragmentManager fm;
    private FragmentTransaction ft;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

```
        //绑定控件
        tv=(TextView)findViewById(R.id.menu);
        //设置监听器，固定写法
        tv.setOnClickListener(this);
        //若是继承FragmentActivity，fm=getFragmentManger();
        fm=getSupportFragmentManager();
        //fm可以理解为Fragment显示的管理者，ft就是它的改变者
        ft=fm.beginTransaction();
        //默认情况下就显示frag1
        ft.replace(R.id.content,new frag());
        //提交改变的内容
        ft.commit();
    }
    @Override
    //控件的点击事件
    public void onClick(View v){
        ft=fm.beginTransaction();
        //切换选项卡
        if (v.getId() == R.id.menu) {
            ft.replace(R.id.content, new frag());
        }
        ft.commit();
    }
}
```

frag.java

```
package com.example.ver1;

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 frag 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={"只因你太美","幻昼（DJ版）","waitwaitwait","情人","没有意外"};
    public static int[] icons={R.drawable.music0,R.drawable.music1,R.drawable.music2,R.drawable.music3,R.drawable.music4};
    @Override
    public View onCreateView(final LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState){
        //绑定布局，只不过这里是用inflate()方法
        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();
        //列表设置适配器
        listView.setAdapter(adapter);
        //列表元素的点击监听器
        listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
                //创建Intent对象，参数就是从frag1跳转到MusicActivity
                Intent intent=new Intent(frag.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;
    }
    //这里是创建一个自定义适配器，可以作为模板
    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VIew，然后绑定控件
            View view=View.inflate(frag.this.getContext(),R.layout.item_layout,null);
            TextView tv_name=view.findViewById(R.id.item_name);
            ImageView iv=view.findViewById(R.id.iv);
            //设置控件显示的内容，就是获取的歌曲名和歌手图片
            tv_name.setText(name[i]);
            iv.setImageResource(icons[i]);
            return view;
        }
    }
}
```