Android 背景音乐设计文档

1.基本信息:

(1).软件名称：Android 背景音乐设计文档

(2)完成人：袁达强

(3)学号：20152100037

(4)完成时间：2017 年 12 月 16 日

2 软件内容简介：

背景音乐设置有主界面中的 menuitem 进入：

背景音乐界面有：

(1).1 个 Switch 用来控制背景音乐的开关。

(2)5 个 Button。其中一个“返回”按钮用来切换至主界面，其他 4 个实现“上一首”，“下一首”，“单曲循环”，“顺序播放”的功能。

(3)2 个 Textview 用来显示当前播放模式和当前目录下的所有音乐文件。

3.界面设计

如上所述的 1 个 Switch 和 5 个操作按钮，以及 2 个 Textview，依次设置每个控件之后界面如下：

4.代码设计

（1）layout 设计:

依次设计每个控件，并设置其 id，如 Switch 控件代码为：

```
<Switch
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/swt_setting"
    android:layout_centerHorizontal="true"
    android:switchMinWidth="50dp"
    android:layout_marginTop="100dp"
    android:textOn="开启"
    android:textOff="关闭"/>
    <TextView
```

（2）MainActivity 设计（Service 设计基本和教程一样，适当加入了自己的功能）：

因为该软件主要响应控件为“上一首”和“下一首”以及“单曲循环”，“顺序播放”按钮。首先指定音乐文件的播放位置“Music”，然后用 File 数组储存该目录下所有音乐文件。对“上一首”，“下一首”按钮实现按下时播放相应音乐文件。“单曲循环”则在 Service 中的 Mediaplayer 中通过 mediaplayer.setLooping(true)实现，“顺序播放”则通过 mediaplayer.setOncompletion 监听当前音乐是否播放完毕，若是，则播放下一首音乐。

1. “上一首”按钮响应代码如下：

```
btn_pre.setOnClickListener((view) -> {
        SDpath= Environment.getExternalStorageDirectory();
        file=new File(SDpath,pathway);
        File b[]=file.listFiles();
        if(i==0) {
            i = k-1;
        }
        else {
            i--;
        }
        path=b[i].getAbsolutePath();
        if (file.exists()&&file.length()>0)
```

```
        path=b[i].getAbsolutePath();
        if (file.exists()&&file.length()>0)
        {

            binder.nextplay(path);
            Toast.makeText(MusicActivity.this, "更换背景音乐为:   " + b[i].getName(), Toast.
        }
});
```

2.“下一首”按钮响应代码：

```
btn_next.setOnClickListener((view) -> {
        SDpath = Environment.getExternalStorageDirectory();
        file = new File(SDpath, pathway);
        File a[] = file.listFiles();
        if (i == k-1) {
            i = 0;
        }
        else {
            i++;
        }
        path = a[i].getAbsolutePath();
        if (file.exists() && file.length() > 0) {
```

```
        path = a[i].getAbsolutePath();
        if (file.exists() && file.length() > 0) {
            binder.nextplay(path);
            Toast.makeText(MusicActivity.this, "更换背景音乐为:   " +a[i].getName(), Toast.
        }
});
bindService(intent, conn, BIND_AUTO_CREATE);
}
```

3.获取目录下音乐文件个数代码：

```
    private void getFiles(String string) {
        // TODO Auto-generated method stub
        File file1 = new File(SDpath,string);
        File[] files = file1.listFiles();
        for (int j = 0; j < files.length; j++) {
            String name = files[j].getName();
            if (files[j].isFile() & name.endsWith(".mp3")||name.endsWith(".wma")||name.endsWith(".wa
                k++;
            }
        }
    }
}
```

4.单曲循环在 Service 中通过 mediaplayer.setLooping(true)实现,代码:

```
    public void sinplay(String path)
    {
        try
        {
            if(mediaPlayer==null)
            {
                mediaPlayer=new MediaPlayer();
                mediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);
                mediaPlayer.setDataSource(path);
                mediaPlayer.prepare();
                mediaPlayer.setOnPreparedListener((mp) -> {
                        mediaPlayer.start();
                        mediaPlayer.setLooping(true);
                });
            }
            else
            {
                int position=getCurrentProgress();
                mediaPlayer.seekTo(position);
                try
                {
```

```
                    mediaPlayer.start();
                    mediaPlayer.setLooping(true);
                });
            }
            else
            {
                int position=getCurrentProgress();
                mediaPlayer.seekTo(position);
                try
                {
                    mediaPlayer.prepare();
                }
                catch (Exception e)
                {
                    e.printStackTrace();
                }
                mediaPlayer.start();
                mediaPlayer.setLooping(true);
            }
        }
        catch (Exception e)
        {
```

5.顺序播放在 Service 中通过 mediaplayer.setOncompletion 监听当前音乐是否播放完毕来实现，代码：

```
    public void orderplay( File b[],int i, int k)
    {
        try
        {
            if(mediaPlayer==null)
            {
                mediaPlayer=new MediaPlayer();
                mediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);
                mediaPlayer.setDataSource(b[i].getAbsolutePath());

                mediaPlayer.prepare();
                mediaPlayer.setOnPreparedListener((mp) -> {
                        mediaPlayer.start();

                });
            }
            else
            {
                int position=getCurrentProgress();
```

```
            }
            else
            {
                int position=getCurrentProgress();
                mediaPlayer.seekTo(position);
                try
                {
                    mediaPlayer.prepare();
                }
                catch (Exception e)
                {
                    e.printStackTrace();
                }
                mediaPlayer.start();
            }
        }
        catch (Exception e)
        {
            e.printStackTrace();
```

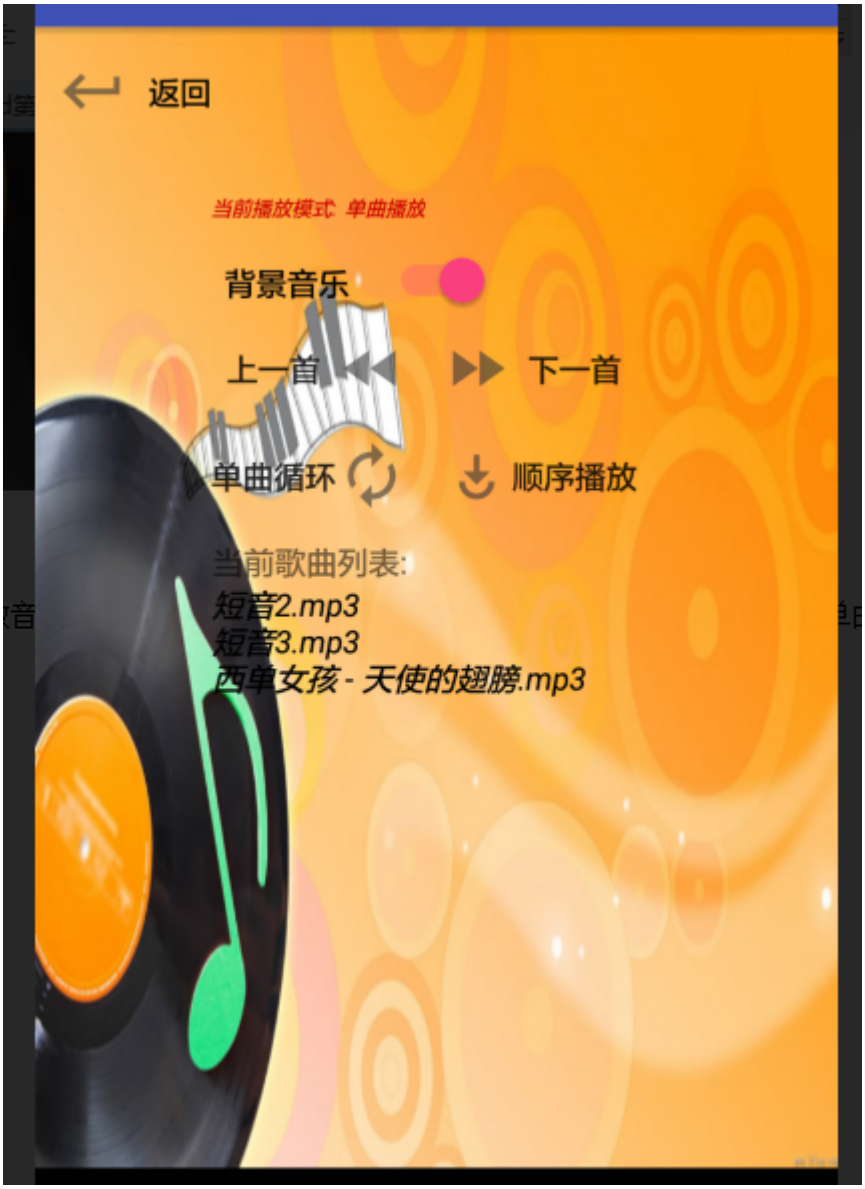
```
            e.printStackTrace();
        }
    final int tempi=i+1;
    final  int tempk=k;
    final   File c[]=b;
      mediaPlayer.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {
            @Override
            public void onCompletion(MediaPlayer mediaPlayer) {
                    orderplay(c, tempi, tempk);
            }
        });
    }
```

6.软件操作流程

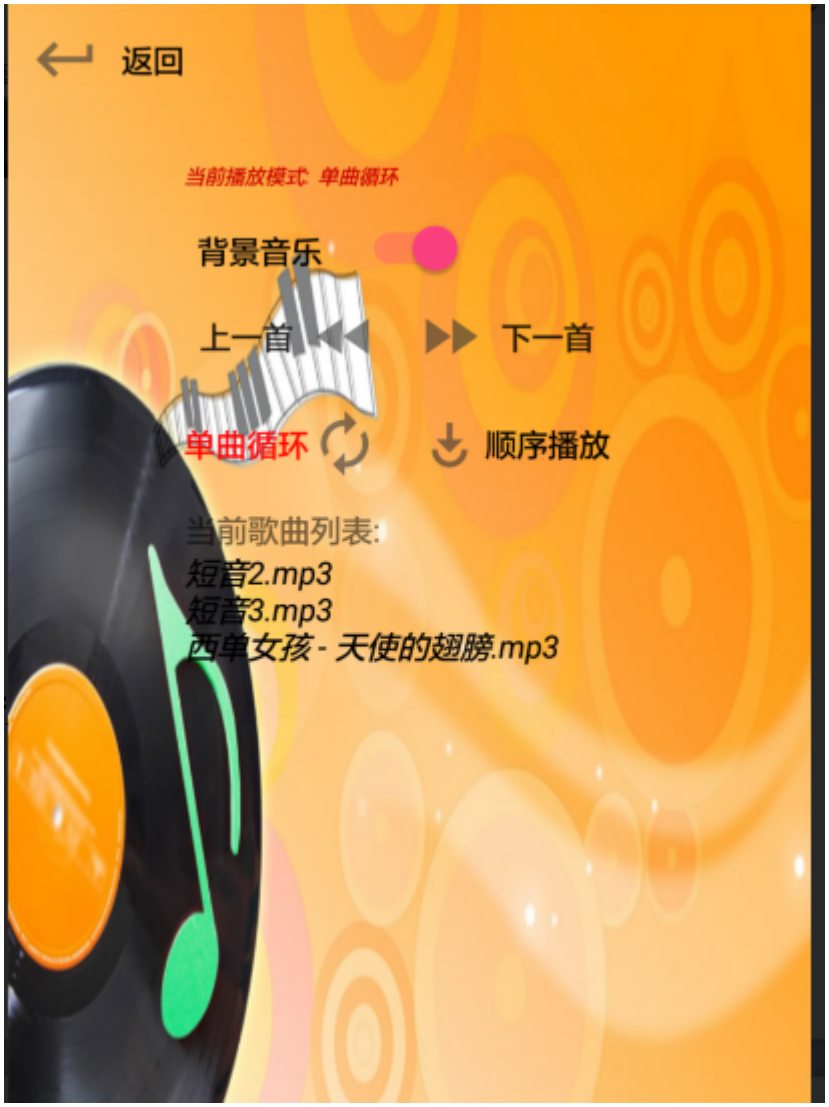
软件操作流程为：
Layout 界面设计--->layout 代码设计-->activity 代码设计。
具体操作如上所示
具体实现效果如下:

1. 进入背景音乐设置界面时:

2. 点击播放音乐，即播放第一首音乐，并显示当前列表，显示默认播放模式为单曲播放:



3. 点击单曲循环或顺序播放，相应字体变红色，并播放模式：



4. 点击“上一首”，“下一首”则立即切换歌曲：

5. 难点和解决方案

（1）已解决：

通过 mediaplayer.setLooping(true) 实现单曲播放，通过 mediaplayer.setOncompletion 监听音乐是否播放完毕来实现“顺序播放”。

（2）未解决：

当返回主界面时，重新进入 music 界面，则不能再控制上次实现的后台操作，对于“顺序播放”，不能很好的递归实现顺序播放

7.今后的设想

对 Service 有一定的理解，但还不能灵活的在 Service 中实现应有的功能，要更深入去了解。