微信小程序的设计与实现

音乐类小程序分析

1 小程序的特点

1 轻应用

微信为了避免开发者将小程序功能做的过于复杂，限制了小程序包的大小。音乐 APP 平台不可能，也没必要再复制一个到小程序，微信小程序对于大小方面的相关限制：
1.不限制分包的数量；
2.所有分包的大小不能超过 20m；
3.单个包的大小不能超过 2m；

2 社交属性

微信给了小程序一个重要的入口，可以直接在聊天窗口中打开，且有特别的展示卡片设计。这是一个微信将其社交属性赋予给了小程序。

3 自传播属性

微信生态信奉的是用户自己主动传播，被动传播和诱导传播被严厉禁止。这就倒逼开发者们专注于产品，给用户主动分享的理由。

1.2 音乐小程序现状分析

为什么目前音乐 APP 在小程序表现一般？结合上述对小程序特点的阐述，我的看法如下：

1、只要有听歌需求的用户，手机内都会装音乐 APP，且现在版权之争，有的装有多个音乐 APP。，加上小程序只能提供最基础的听歌需求，性能体验上小程序比不上原生 APP，这就导致用户没有在小程序内选择听歌小程序听歌的需求和动力。

2、音乐 APP 在小程序内布局的其他小程序，如上述听歌功能，小程序只是简单的做了个简配版的，用户没有使用的需求和动力。

3、缺乏传播属性。目前只是单纯的将小程序做出来了，并未深入利用小程序的社交传播属性。

最重要的一点：

当前音乐 APP 为小程序设置的目标是给自家 APP 引流，这就必然导致了小程序流量低，来的用户没有下载 APP 的就直接去下载 APP 了，有 APP 的就直接打开 APP 了。基于这点来看，小程序流量低反而符合音乐 APP 的最初目标了。

1.3 如何破局

第一，明确小程序的定位。定位必然是为了自家 APP 引流，提高自家 APP 的下载量和用户活跃度。

第二，将小程序分成两类。

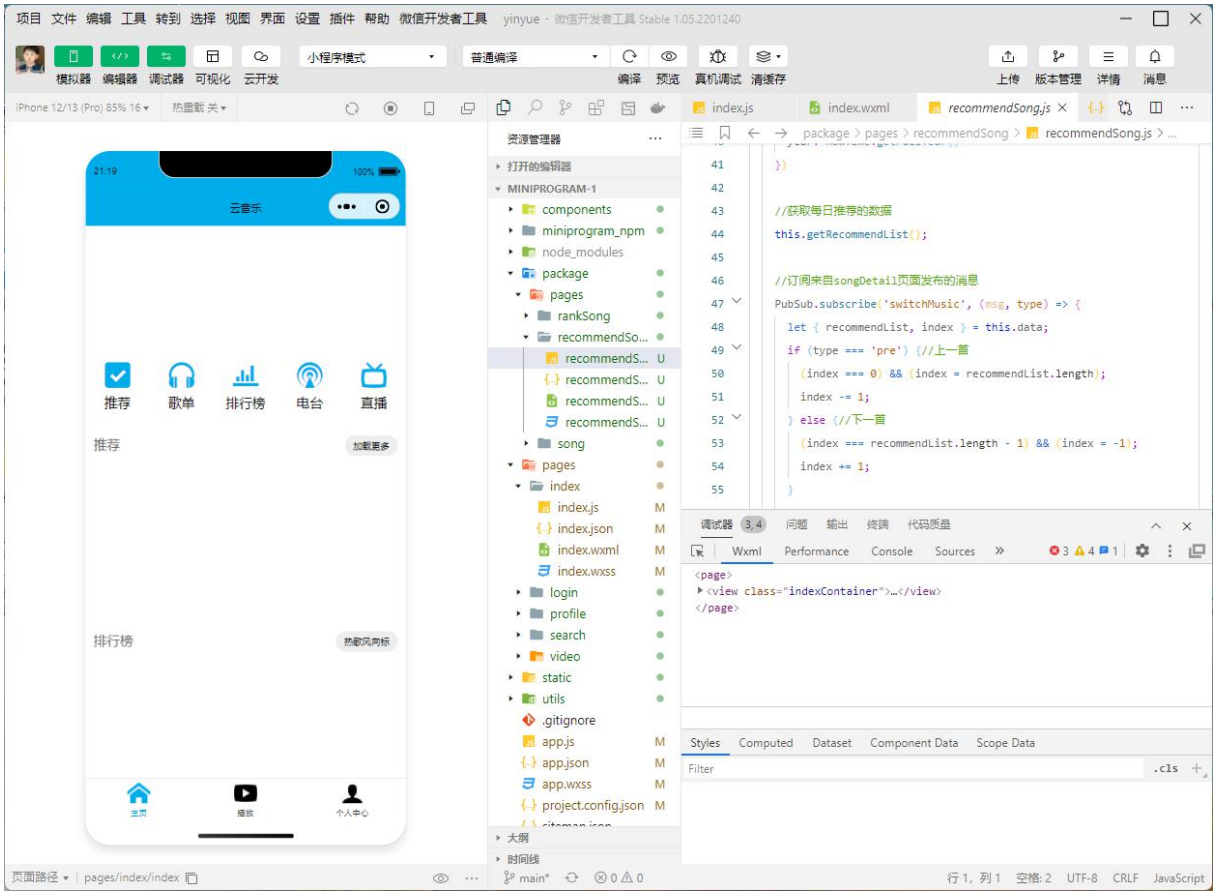
（1）从繁到简。将音乐 APP 内一些核心功能拆分和组合，在小程序内做成一个简版，保留其核心内容。同时提升小程序的传播属性（分享能力）。

（2）从简到繁。在音乐 APP 内挖掘一些有潜力需求和功能，但目前在 APP 内处于边缘地带，功能还比较初级，但具有想象空间的；同时满足适合做在小程序内且能发挥其社交传播属性。找准这个点后，从简到繁。

2 开发工具介绍

微信开发者工作是微信官方提供的针对微信小程序的开发工具，集中了开发，调试，预览，上传等功能。微信团队发布了微信小程序开发者工具、微信小程序开发文档和微信小程序设计指南，全新的开发者工具，集成了开发调试、代码编辑及程序发布等功能，帮助开发者简单和高效地开发微信小程序。启动工具时，开发者需要使用已在后台绑定成功的微信号扫描二维码登录，后续所有的操作都会基于这个微信的帐号

界面布局：



小程序项目创建成功后，会自动进入开发调试环境，从图中可以看出，微信开发者工具的主界面主要由菜单栏、工具栏、模拟器、编辑器和调试器组成。接下来对这些功能分别进行讲解

菜单栏：

通过菜单栏可以访问微信开发者工具的大部分功能，常用的菜单如下:

- 项目:用于新建项目，或打开一个现有的项目
- 文件:用于新建文件、保存文件或关闭文件
- 编辑:用于编辑代码，对代码进行格式化
- 工具:用于访问一些辅助工具,如自动化测试、代码仓库等
- 界面:用于控制界面中各部分的显示和隐藏
- 设置:用于对外观、快捷键、编辑器等进行设置
- 微信开发者工具:可以进行切换账号、更换开发模式、调试等操作

工具栏：

工具栏提供了一些常用功能的快捷按钮，具体解释如下：

- 个人中心:位于工具栏最左边的第一个按钮，显示当前登录用户的用户名、头像

• 模拟器、编辑器和调试器:用于控制相应工具的显示和隐藏

云开发：开发者可以使用云开发来开发小程序、小游戏，无需搭建服务器，即可使用云端能力。云开发能力从基础库 2.2.3 开始支持

• 模式切换下拉菜单:用于在小程序模式、搜索动态页和插件之间进行切换

• 编译下拉菜单:用于切换编译模式，默认为普通编译，可以添加其他编译模式

。编译:编写小程序的代码后，需要编译才能运行。默认情况下，直接按 Ctrl+S 快捷键保存代码文件，微信开发者工具就会自动编译运行。若要手动编译，则单击“编译”按钮即可

。预览:单击“预览”按钮会生成一个二维码，使用手机中的微信扫码二维码，即可在微信中预览小程序的实际运行效果

真机调试:可以实现直接利用开发者工具，通过网络连接对手机上运行的小程序进行调试，帮助开发者更好地定位和查找在手机上出现的问题

• 切后台:用于模拟小程序在手机中切后台的效果

清缓存:用于将代码上传到小程序管理后台，可以在“开发管理”中查看上传的版本，将代码提交审核。需要注意的是，如果在创建项目时使用 ApplD 为测试号,则不会显示“上传”按钮

• 版本管理:用于通过 Git 对小程序进行版本管理

模拟器：

上方的 iPhone 12 表示手机型号，单击可以切换成其他手机。由于不同手机屏幕的 CSS 像素不同，宽高比也不同，在开发小程序时应对常见的手机屏幕进行适配。100% 表示缩放百分比，可以调节预览画面的大小。WIFI 表示网络环境，还可以切换成 2G、3G、4G 或 Offline(离线)，不同环境的网速不同，从而可以测试小程序的网络加载速度。模拟器的底部状态栏显示了当前的页面路径为/pages/index/index

调试器：

调试器类似于 Google Chrome 浏览器中的开发者工具。下面对调试器中的各个面板的功能进行简要介绍。

- Console:“控制台”面板，用于输出调试信息，也可以直接编写代码执行

Source:“源代码”面板，可以查看或编辑源代码．并支持代码调试。

Network:“安全”面板，用于调试页面的安全和认证等信息，如 HTTPS

AppData:“App 数据”面板，可以查看或编辑当前小程序运行时的数据

Audits:“审计”面板，用于对小程序进行体验评分

Sensor:“传感器”面板，用于模拟地理位置、重力感应

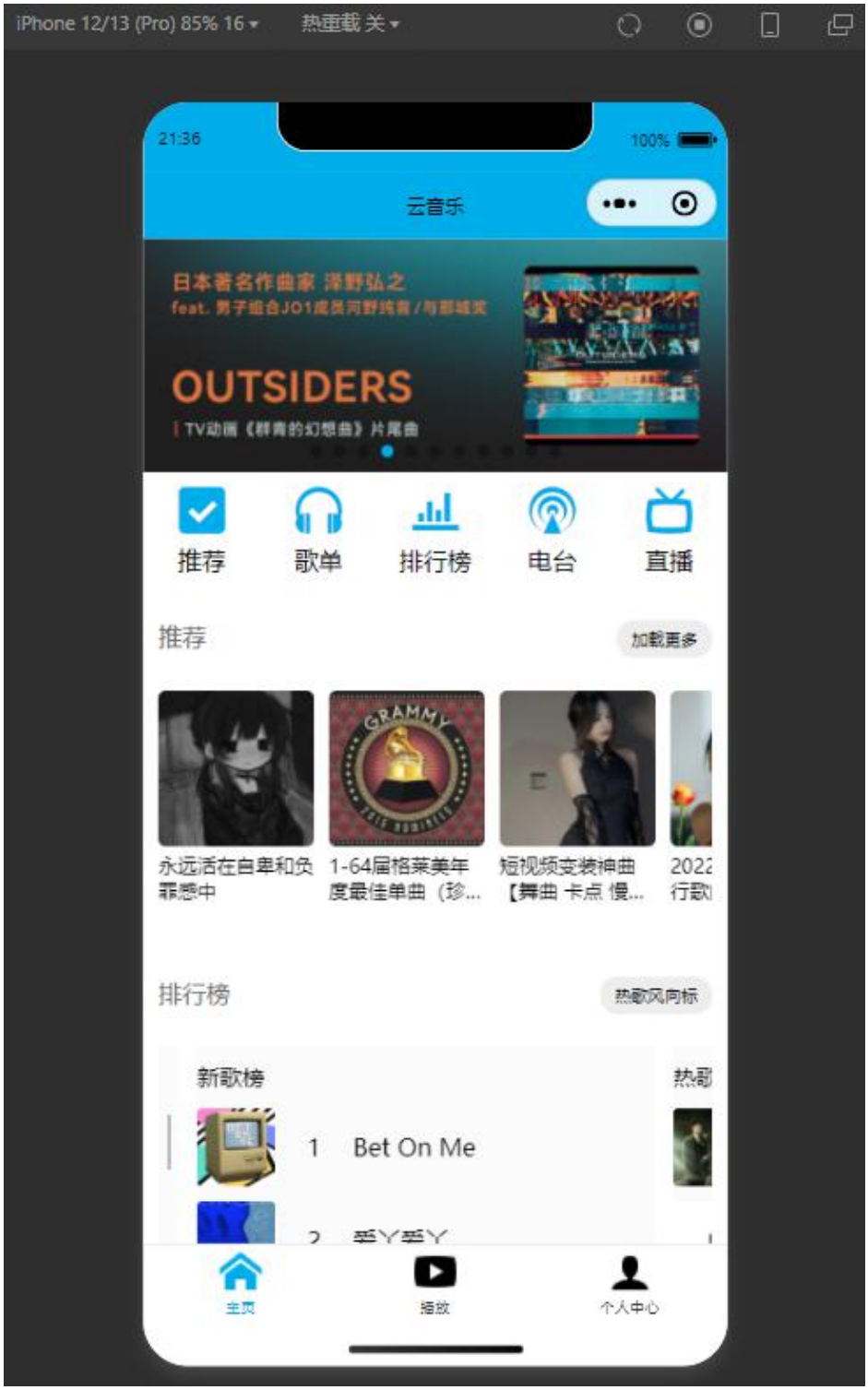
Storage:“存储”面板，用于查看和管理本地数据缓存.

Trace:“跟踪”面板,用于真机调试时跟踪调试信息.

Wxml: Wxml 面板.用于查看和调试 WXML 和 WXSS

3 具体页面设计

3.1 小程序首页（index）



1. swiper 组件

swiper 组件是滑块视图容器，是纯 javascript 打造的滑动特效插件，面向手机、平板电脑等移动终端，能实现触屏焦点图、触屏 Tab 切换、触屏轮播图切换等常用效果。

还能制作多种 3D 切换效，如 cube、coverflow、flip、cards 和 creative 可以轻松的实现 3D 过渡，还能够通过 progress 自定义过度效果

小程序的 swiper 组件中只可放置 swiper-item 组件，用到的属性有：

属性	类型	
indicator-dots	boolean	是否显示面板指示点
indicator-color	color	指示点颜色
indicator-active-color	color	当前选中的指示点颜色
autoplay	boolean	是否自动切换
interval	number	自动切换时间间隔
circular	boolean	是否采用衔接滑动

2. 字体图标导航栏

字体图标：

字体图标简单的说，就是一种特殊的字体，通过这种字体，显示给用户的就像一个个图片一样。字体图标最大的好处，在于它不会变形和加载速度快。字体图标可以像文字一样，随意通过 CSS 来控制它的大小和颜色，对于建网站来说，非常方便。

字体图标是使用微小图像而不是字母形式的字体。 与字符一样，每个图标字体都是矢量元素，可根据需要进行伸缩，并可以使用 CSS 样式进行修改。

使用图标字体的主要原因是你可以轻松更改大小，颜色，形状。 图标字体本质上是透明的，因此你可以将它们放在任何颜色或类型的背景上，还可以添加笔触或更改图标的不透明度。

所有的设置操作均可以使用 CSS 完成，因此你不必为设计中的每个新图标实例创建唯一的图像(这对于保持网站设计和代码的轻量级非常有用)。

图标字体是矢量图像，这意味着它们是可随意进行缩放而不失真的。 与其他字体一样，你可以根据需要将它们设置更大或更小。 仅将图标字体用作艺术元素，甚至在其他文本字段中。

使用字体图标的优劣势

1.优势

· 只要 CSS 中存在相应的类，图标就易于缩放;

· 你可以在互联网上轻松获得既有的大型图标库;

· 更改图标属性非常容易，无论你使用什么来构建网站的;

· 将图标添加到网站设计上的任何页面都很容易(只需插入图标名称);

· 有数千种各种样式的图标可供选择;

· 如果需要更高的自定义，你甚至可以创建自己的图标字体。

2.劣势

· 图标是单色或渐变色(在大多数情况下，不过一些可以提供更多自定义选项的服务正在改变这一点);

· 如果无法加载到图标字体，则没有其他的备用字体供使用;

· 你可能不需要完整的图标集;

· 一些浏览器器还不能或不完全支持图标字体;

· 你可能找不到适合你的需求的资源。

此小程序使用的是 icomoon（字体图标网站），他的优点有：

1.支持上传自定义图标，包括多色块图标

2.当然，你也可以挑选线上他人共享的图标

3.支持在线对图标进行编辑

4.将所有选择的图标合成一个图标文件，类似“雪碧图”功能，并可以本地预览。

在调试过程中，字体图标在手机上运行时会有错误显示，于是我想到了转化为图片形式，在使用了 png 格式的图标时调试台报错，提示只能使用网络图片或者 BASE64 格式图片。我使用 https://transfonter.org/网站转化成了可识别的 BASE64 格式图片

3. 推荐歌曲

推荐歌曲栏为横向的可滚动区域，利用 scroll-view 组件包裹内容，设置相应的属性达到预期效果，数据和内容是通过 promise 异步请求过来的数据，进行处理后使用 wx:for 遍历数据并展示具体实现如下：

```
<scroll-view class="recommendScroll" enable-flex scroll-x>
  <view class="scrollItem" wx:for="{{recommendList}}" wx:key="id">
    <image src="{{item.picUrl}}"></image>
    <text>{{item.name}}</text>
  </view>
</scroll-view>
```

排行榜区域

排行榜包括了 10 类歌曲排行，热歌帮、新歌榜、原创榜、飙升榜等，每个榜单中记录了十首歌曲的排行，每首歌曲都可以点击跳转至歌曲页面播放，通过 promise 请求的数据嵌套了两层数组，第一层是多个榜单数组，第二层是每个榜单包含的歌曲信息

实现思路：

由于数据有两层关系，所以使用了双重的 wx：for 遍历，并用 scroll-view 组件包裹第二次 wx：for 的数据，使用 swiper 组件包裹第一层数据，实现排行榜的左右滑动效果及榜单的上划查看更多歌曲效果，实现代码如下：

```
<swiper class="topListSwiper" circular next-margin="50rpx" previous-margin="50rpx">
  <swiper-item wx:for="{{topList}}" wx:key="name">
    <view class="swiperItem">
      <view class="title">{{item.name}}</view>
      <scroll-view class="musicList" scroll-y="{{true}}">
        <view class="musicItem" wx:for="{{item.tracks}}" wx:key="id" wx:for-item="music
Item" bindtap="toSong" data-id='{{musicItem.id}}'>
          <image src="{{musicItem.al.picUrl}}"></image>
          <text class="count">{{index + 1}}</text>
          <text class="musicName">{{musicItem.name}}</text>
        </view>
      </scroll-view>
```

```
    </view>
  </swiper-item>
</swiper>
```

3.2 小程序视频页（video）

此页面为视频播放页，进入页面时会对登录信息进行判断，如果用户没有进行登录则跳转至登录页面。首先页面对导航栏和视频区域分别进行了 scroll-view 组件的应用，

并设置了切换列表数据时导航栏的过度动画，每个列表的数据都是在用户点击时再动态获取数据更新，加快了页面的首次加载速度；接着使用了 wx:if 属性进行判断存放视频数据的 video 标签是否更新到页面中，以减少页面空白时间给用户更好的浏览体验，此外设置了分享功能，并区别用户是点击页面分享（右上角三个点）还是视频的分享，以此设置分享的标题与展示的图片

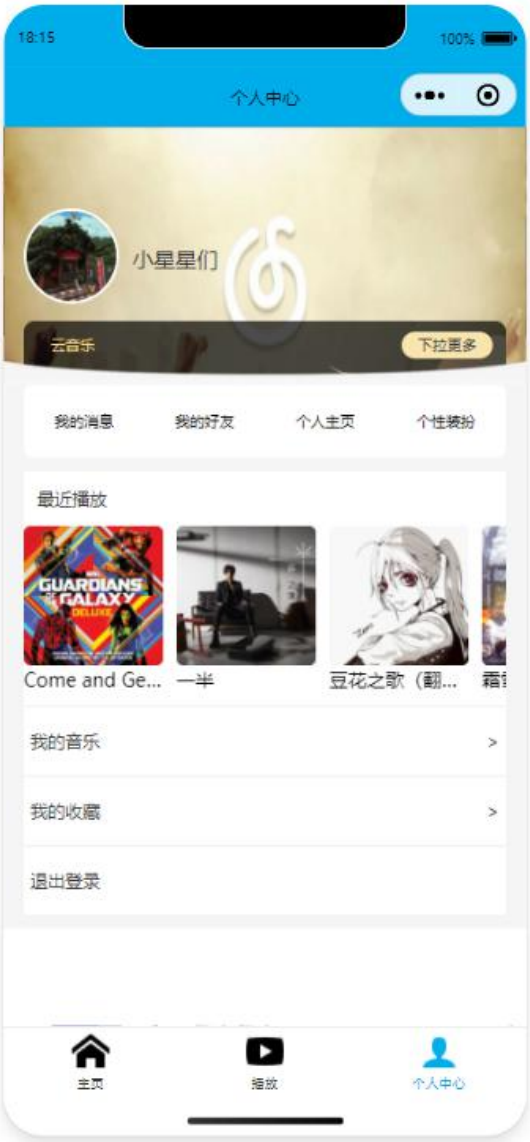
视频页加载时会先用图片替代视频所在位置，当用户点击时动态替换为视频，实现的思路是用 wx:if 对 js.data 中的数据进行判断并监听用户的点击，用户点击时 wx:if 判断为 true 此时<image>会被替换为<video>也就是图片替换为视频进行播放。

视频播放时用户点击下一个视频会导致多个视频同时播放，需要用户手动关闭上一个视频，极大的影响了用户体验，因此在每个视频元素中添加了点击事件，用户点击时会传入每个视频的唯一标识，用户点击下一个时会对唯一标识进行判断，唯一标识不同代表用户点击了另一个视频，此时需要主动调用 this.videoContext.stop()事件暂停上一个视频。

主要代码：

```
/**
 * 播放 Video
 * 关闭上一个正在播放的 video
 */
playVideo(event) {
  let pid = event.currentTarget.id;
  pid !== this.data.pid && this.videoContext && this.videoContext.stop();
  this.videoContext = wx.createVideoContext(pid, this)
  this.videoContext.play();
  this.setData({
    pid
  })
},
play(event) {
  let videoId = event.currentTarget.id;
  this.setData({
    videoId
  })
},
```

3.3 个人主页（profile）



个人信息页首先会用户信息进行查找，找不到本地用户信息将会显示默认的头像并有立即登录提示，点击后会跳转至登录页面。如果本地已经有了用户的信息将会展示用户的头像、昵称和最近播放记录，并对最近播放的歌曲进行排列展示，点击后可以跳转至歌曲页面播放。在最下方设置了退出登录的功能，原理是：获取到的用户信息会存储至本地 storage 中，只需要调用清除的函数传入要删除的 key 值便可以清除用户数据，最后跳转至登录页，此时个人主页会被销毁，再次进入会重新加载页面数据

在个人头像下方制作了下拉加载隐藏页面的效果，当用户下拉时被隐藏的页面会随着下拉逐渐显现，当用户下拉一定的距离时页面变为最终效果，否则页面会恢复。下拉效果完成时可以点击箭头，使页面过渡至初始状态，主要实现代码：

```
handleTouchStart(event) {
    this.setData({
      coveTransition: ''
    })
    // 获取手指起始坐标
    startY = event.touches[0].clientY;
  },
  handleTouchMove(event) {
    if (this.data.flag1 == 0) {
      moveY = event.touches[0].clientY;
      moveDistance = moveY - startY;

      if (moveDistance <= 0) {
        this.setData({
          flag: moveDistance
        })
      }
      if (moveDistance > 0 && moveDistance < 120) {
        this.setData({
          flag: moveDistance
        })
      }
      if (moveDistance >= 120) {
        moveDistance = 120;
      }
      // 动态更新 coverTransform 的状态值
      if (this.data.flag > 0) {
        this.setData({
          coverTransform: `translateY(${moveDistance}rpx)`
        })
      }
    }

  },
  handleTouchEnd() {
```

```
  if (this.data.flag1 == 0) {
    // 动态更新 coverTransform 的状态值
    if (moveDistance == 120) {
      this.setData({
        coverTransform: `translateY(400rpx)`,
        coveTransition: 'transform 1s linear',
        flag1: true
      })
    } else if (this.data.flag > 0 && this.data.flag < 120) {
      this.setData({
        coverTransform: `translateY(0rpx)`,
        coveTransition: 'transform 1s linear',

      })
    }
  }

},
goBack() {
  this.setData({
    flag1: false,
    coverTransform: `translateY(0rpx)`,
    coveTransition: 'transform 1s linear',
  })
},
```

3.4 歌曲推荐 / 热歌排行榜（ranksong / recommendsong）

歌曲推荐页与热歌排行榜页逻辑大致相同，下面以歌曲推荐页作为介绍

歌曲推荐页头部通过 Date（）获取并展示当前的日期，页面的歌曲列表都是以具体时间进行更新。点击列表的歌曲时会通过路由传参的方法将点击歌曲的 ID 传至歌曲页面，通过 ID 获取点击歌曲的具体信息

3.5 歌曲播放页

歌曲播放页通过不同页面点击时通过路由传入的 ID 进行歌曲数据的获取，将歌曲

信息展示在页面中，并对点击歌曲播放制作了摇杆动画以及磁盘动画。开始播放歌曲时会将实时的歌词展示在页面中，并通过 CSS 动画实现了进度条效果。

摇杆动画和磁盘动画使用的都是 CSS 进行制作，相较于使用 javascript 动画，CSS 性能更好而且可以调用 GPU 加速，在轻量化的小程序中尤为合适。动画的原理是通过 CSS 的 transform 属性及 animation 属性。transform 可以让图片进行旋转，通过判断当前歌曲的播放状态调动摇杆图片旋转，在暂停歌曲时再次回调修改 transfrom 动画达到动画效果。Animation 的基本语法是：animation：name keeping-time animate-function delay times iteration final；动画过程使用@-webkit-keyframes 进行设置，设置方法有两种分别为 关键帧和 from to，关键帧模式通过动画过程百分比的形式制作效果，可以制作比较复杂，变化较多的动画，此项目使用的方法为 from to，只进行旋转动画。

歌曲可通过上一首下一首图标进行歌单中歌曲的切换，但是此时页面只有点击时传入的歌曲 ID，无法获取列表所有歌曲数据，于是将推荐页的所有歌曲通过路由传入，运行后发现报错，此时传入的数据只有不完整的一段，查阅官方文档发现路由传参有大小限制，不足以传入所有歌曲信息，于是下载使用了第三方 npm 包 pubsub-js。此包传递数据的原理是通过订阅者-发布者模式进行数据传递，具体方法为：

歌曲推荐页将所有歌曲设置不同的下标值

歌曲播放页面通过 PubSub.publish 发布消息数据给推荐歌曲页面并传入所需歌曲的下标值

歌曲推荐页通过 PubSub.subscribe 接受到歌曲播放页传入的下标值，将对应下标值的歌曲信息发布至歌曲播放页

歌曲播放页面获取到歌曲数据后动态更新页面并加载歌曲

3.6 登录页面

此页面结构较为简单，使用 input 表单标签作为输入框，在点击登录时会对所填的表单进行前端验证，具体判断的结果有：‘手机号为空’、‘手机号格式错误’、‘密码为空’，通过前端验证后会将表单数据提交至服务器进行后端验证，通过返回的网络状态码判断，结果有：‘400 = 手机号错误’、‘502 = 密码错误’、‘501 = 账号不存在’，当前端验证宇后端验证都通过时，为将获取的用户信息通过 wx.setStorageSync() 方法保存至本地，具体的逻辑为：

前端验证

1) 验证用户信息(账号，密码)是否合法

2) 前端验证不通过就提示用户，不需要发请求给后端

3) 前端验证通过了，发请求(携带账号, 密码)给服务器端

后端验证

1) 验证用户是否存在

2) 用户不存在直接返回，告诉前端用户不存在

3) 用户存在需要验证密码是否正确

4) 密码不正确返回给前端提示密码不正确

5) 密码正确返回给前端数据，提示用户登录成功(会携带用户的相关信息)

代码实现：

```
if (!phone) {
    // 提示用户
    wx.showToast({
      title: '手机号不能为空',
      icon: 'none'
    })
    return;
  }
  // 定义正则表达式
  let phoneReg = /^1(3|4|5|6|7|8|9)\d{9}$/;
  if (!phoneReg.test(phone)) {
    wx.showToast({
      title: '手机号格式错误',
      icon: 'none'
    })
    return;
  }
  if (!password) {
    wx.showToast({
      title: '密码不能为空',
      icon: 'none'
    })
    return;
```

```
}
// 后端验证
let result = await request('/login/cellphone', { phone, password, isLogin: true })
if (result.code === 200) { // 登录成功
  wx.showToast({
    title: '登录成功'
  })
  // 将用户的信息存储至本地
  wx.setStorageSync('userInfo', JSON.stringify(result.profile))

  // 跳转至个人中心 personal 页面
  wx.reLaunch({
    url: '/pages/profile/profile'
  })
} else if (result.code === 400) {
  wx.showToast({
    title: '手机号错误',
    icon: 'none'
  })
} else if (result.code === 502) {
  wx.showToast({
    title: '密码错误',
    icon: 'none'
  })
} else if (result.code === 501) {
  wx.showToast({
    title: '账号不存在，请注册',
    icon: 'none'
  })
} else {
  wx.showToast({
    title: '登录失败，请重新登录',
    icon: 'none'
  })
```