第 3 章

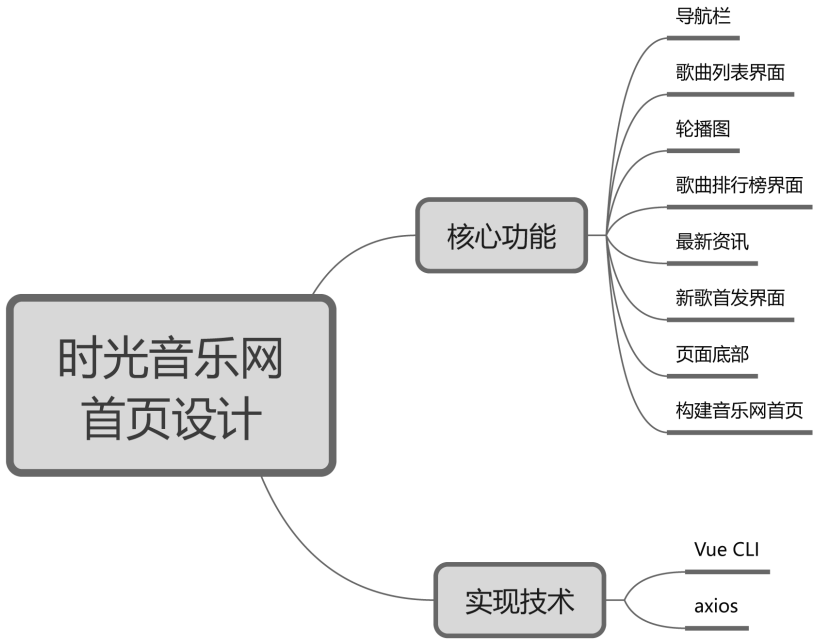
时光音乐网首页设计

——Vue CLI + axios

音乐类网站是当今流行音乐的主流发布平台。随着网络技术的发展，用户对音乐网站的要求逐渐提高。音乐网站为了吸引更多的用户，需要对界面进行不断的美化。本章将使用 Vue CLI 和 axios 设计一个简洁大方的音乐网首页。

项目微视频

本项目的核心功能及实现技术如下：



3.1 开 发 背 景

在人们的日常生活中，音乐已经成为不可或缺的一部分。随着互联网技术的不断发展，人们对于音乐的需求也日益增加，这促使在线音乐网站应运而生。该类网站可以为用户提供丰富的音乐资源和个性化的音乐推荐，使人们能在线欣赏不同风格、不同国家的歌曲，实现资源共享。本章将使用 Vue CLI 和 axios 开发一个音乐网站的首页，其实现目标如下：

- ☑ 通过歌曲列表展示歌曲信息。
- ☑ 通过轮播图展示歌曲封面。
- ☑ 根据选项卡切换不同类别的歌曲列表。
- ☑ 间断滚动最新音乐资讯。
- ☑ 分页展示首发歌曲信息。

3.2 系 统 设 计

3.2.1 开发环境

本项目的开发及运行环境如下：

☑ 操作系统：推荐 Windows 10、Windows 11 或更高版本，同时兼容 Windows 7（SP1）。
☑ 开发工具：WebStorm。
☑ 开发框架：Vue.js 3.0。

3.2.2 业务流程

时光音乐网首页由多个单文件组件构成，包括导航栏组件、歌曲列表组件、轮播图组件、歌曲排行榜组件、最新音乐资讯组件、新歌首发组件和页面底部组件。

根据该网站首页的业务需求，我们设计如图 3.1 所示的业务流程图。

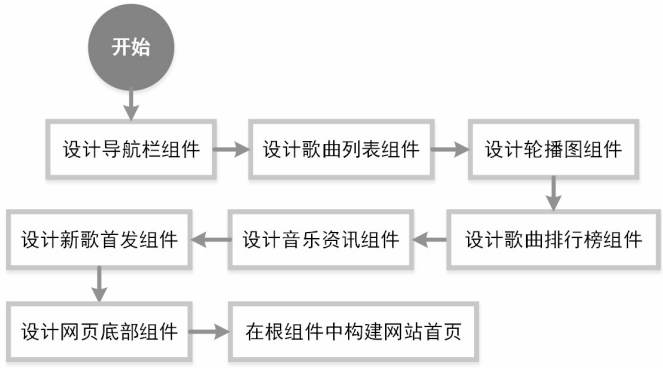


图 3.1 业务流程图

3.2.3 功能结构

本项目的功能结构已经在章首页中给出，其实现的具体功能如下：

☑ 导航栏：鼠标指向某个导航菜单项，下方显示该菜单项对应的子项。
☑ 歌曲列表：展示歌曲信息，包括歌曲图片、歌曲名称和歌曲简介。
☑ 轮播图：单击某个数字按钮可切换到对应的图片。
☑ 歌曲排行榜：根据选项卡切换不同类别的歌曲列表。
☑ 最新音乐资讯：从下到上间断循环滚动音乐资讯。
☑ 新歌首发：分页展示首发歌曲信息。
☑ 页面底部：展示版权等方面的信息。

3.3 技 术 准 备

在开发时光音乐网首页时，我们主要应用 Vue CLI 脚手架工具和 axios。下面将简要概述本项目所用的

Vue.js 核心技术点及其各自的具体作用。

1．单文件组件

将一个组件的 HTML、JavaScript 和 CSS 应用各自的标签写在一个文件中，这样的文件即为单文件组件。单文件组件是 Vue 自定义的一种文件，以.vue 作为文件的扩展名。示例代码如下：

```
<template>
    <p>{{ msg }}</p>
</template>
<script>
    export default {
        data: function () {
            return {
                msg: '长风破浪会有时，直挂云帆济沧海。'
            }
        }
    }
</script>
<style scoped>
    p {
        font-size: 36px;                    /*设置文字大小*/
        text-align: center;                 /*设置文字居中显示*/
        color: #FF00FF;                     /*设置文字颜色*/
    }
</style>
```

2．Vue CLI

Vue CLI 是一个基于 Vue.js 进行快速开发的完整系统。最新版本的 Vue CLI 的包名从原来的 vue-cli 被更改为了@vue/cli。Vue CLI 是使用 Node.js 编写的命令行工具，需要进行全局安装来使用。要安装最新版本的 Vue CLI，需要在命令提示符窗口中输入如下命令：

```
npm install -g @vue/cli
```

要使用 Vue CLI 创建项目，可以使用 vue create 命令。例如，要创建一个名称是 myapp 的项目，需要输入以下命令：

```
vue create myapp
```

执行该命令后只需遵循命令行工具提供的指示步骤，就能顺利完成项目的创建。

3．axios

axios 是一个基于 Promise 的 HTTP 客户端，它既可以在浏览器中使用，也可以在 Node.js 中运行。使用 axios，可以实现 Ajax 请求，进而实现本地与服务器之间的通信。在项目中使用 axios 时，可以通过 npm 方式进行安装。在命令提示符窗口中，输入如下命令进行安装：

```
npm install axios --save
```

安装 axios 之后，需要在项目中引入它，代码如下：

```
import axios from 'axios'
```

axios 常用的请求方法包括 GET 和 POST 等。其中：GET 请求主要用于从服务器上获取数据，传递的数据量比较小；POST 请求主要用于向服务器传递数据，传递的数据量比较大。使用 axios 无论发送 GET 请求还是 POST 请求，在发送请求后都需要使用回调函数对请求的结果进行处理。如果请求成功，则需要使用.then()方法处理请求的结果；如果请求失败，则需要使用.catch()方法处理请求的结果。示例代码如下：

```
axios.get('/book',{
    params:{                                    //传递的参数
```

```
            type : 'Vue',
            number : 10
        }
}).then(function(response){
        console.log(response.data);
}).catch(function(error){
        console.log(error);
})
```

注意

这两个回调函数各自拥有独立的作用域，如果在它们内部访问 Vue 实例，就不能直接使用 this 关键字。为了解决这个问题，需要在回调函数的后面添加.bind(this)来确保 this 指向正确的 Vue 实例。

有关《Vue.js 从入门到精通》一书详细地讲解了单文件组件、Vue CLI、axios 等基础知识。对于这些知识不太熟悉的读者，可以参考该书的相应章节进行深入学习。

3.4 功 能 设 计

设计时光音乐网首页各功能模块之前，需要先使用 Vue CLI 创建项目，项目名称设置为 music。创建项目之后，即可开始首页中各功能模块的设计。

3.4.1 导航栏的设计

网站首页的导航栏包含 6 个菜单项。除了“首页”，还有 5 个菜单项，分别表示 5 种不同类型的音乐风格：流行音乐、摇滚音乐、民族音乐、古典音乐和金属乐。当鼠标悬停在某个导航菜单项上时，其下方会显示对应的子项。网站首页的导航栏效果如图 3.2 所示。

图 3.2 网站首页的导航栏效果

设计导航栏的关键步骤如下。

（1）在 components 目录中创建 nav 文件夹，在该文件夹下创建 6 个与导航菜单项对应的子项组件。然后在 components 目录中创建 MusicNav.vue 文件。在 MusicNav.vue 文件的<template>标签中定义导航栏，将导航菜单项的类名和定义的数据进行绑定。当鼠标悬停在菜单项上时，触发 mouseover 事件，并通过为数据赋值的方式改变当前渲染的组件，再使用动态组件的形式实现 6 个子项组件之间的切换。在<script>标签中引入 6 个子项组件，并对 6 个组件进行注册。具体代码如下：

```
<template>
  <div class="menu">
    <div class="main_menu">
      <ul>
        <li :class="{active: current === 'Home'}" @mouseover="current = 'Home'">首页</li>
        <li :class="{active: current === 'Pop'}" @mouseover="current = 'Pop'">流行音乐</li>
        <li :class="{active: current === 'Rock'}" @mouseover="current = 'Rock'">摇滚音乐</li>
        <li :class="{active: current === 'Nation'}" @mouseover="current = 'Nation'">民族音乐</li>
        <li :class="{active: current === 'Classical'}" @mouseover="current = 'Classical'">古典音乐</li>
        <li :class="{active: current === 'Metal'}" @mouseover="current = 'Metal'">金属乐</li>
      </ul>
```

```
      </div>
      <div class="sub_menu">
        <component :is="current"></component>
      </div>
    </div>
</template>
<script>
import Home from "@/components/nav/HomeNav";
import Pop from "@/components/nav/PopNav";
import Rock from "@/components/nav/RockNav";
import Nation from "@/components/nav/NationNav";
import Classical from "@/components/nav/ClassicalNav";
import Metal from "@/components/nav/MetalNav";
export default {
  name: "MusicNav",
  data: function (){
    return {
      current: 'Home'                                    //当前导航菜单项组件名
    }
  },
  components: {
    Home,                                                //注册 Home 组件
    Pop,                                                 //注册 Pop 组件
    Rock,                                                //注册 Rock 组件
    Nation,                                              //注册 Nation 组件
    Classical,                                           //注册 Classical 组件
    Metal                                                //注册 Metal 组件
  }
}
</script>
```

（2）在 assets 目录中创建 css 文件夹，在该文件夹下创建 style.css 文件，在该文件中编写 CSS 代码，为导航栏组件设置样式。代码如下：

```
.main_menu{
    width: 1000px;                                       /*设置宽度*/
    margin: 0 auto;                                      /*设置外边距*/
    border-top:solid 1px #D0D0D0;                        /*设置上边框*/
    border-bottom:solid 1px #D0D0D0;                     /*设置下边框*/
    font-weight:bold;                                    /*设置文字粗细*/
    letter-spacing:2px;                                  /*设置文字间距*/
}
ul,li{
    list-style: none;                                    /*设置列表无样式*/
}
.main_menu li{
    float: left;                                         /*设置左浮动*/
    width: 165px;                                        /*设置宽度*/
    height: 42px;                                        /*设置高度*/
    line-height: 42px;                                   /*设置行高*/
    text-align: center;                                  /*设置文本水平居中显示*/
    cursor: pointer;                                     /*设置鼠标形状*/
}
.main_menu li:not(:last-child){
    border-right: 1px solid #D0D0D0;                     /*设置右边框*/
}
.sub_menu{
    width: 1000px;                                       /*设置宽度*/
    margin: 0 auto;                                      /*设置外边距*/
    height: 42px;                                        /*设置高度*/
}
.sub_menu span,.sub_menu li{
```

```
    height: 42px;                          /*设置高度*/
    line-height: 42px;                     /*设置行高*/
}
.sub_menu li{
    float: left;                           /*设置左浮动*/
    text-align:left;                       /*设置文本水平向左显示*/
    padding-left:12px;                     /*设置左内边距*/
    margin-left:25px;                      /*设置左外边距*/
}
```

3.4.2 歌曲列表展示界面

在时光音乐网首页中，主显示区域的左侧是一个歌曲列表展示界面。该界面共展示了 9 首歌曲，每首歌曲都配有对应的歌曲图片、歌曲名称和歌曲简介。界面效果如图 3.3 所示。

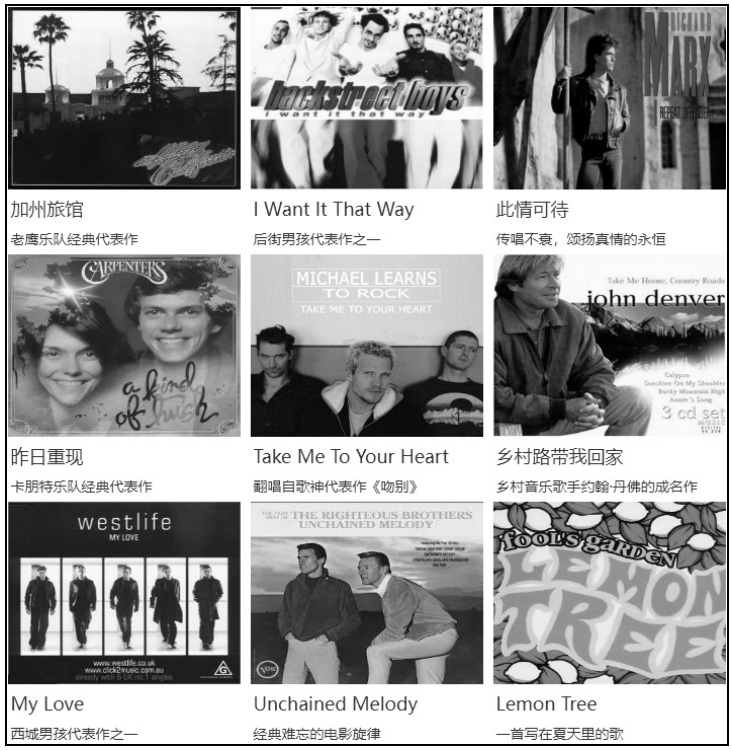


图 3.3 歌曲列表展示界面

歌曲列表展示界面的实现步骤如下。

（1）在 components 目录中创建 SongList.vue 文件。在 SongList.vue 文件的<template>标签中，定义一个 ul 列表，并使用 v-for 指令对保存歌曲信息的数组 song_list 进行遍历。在<script>标签中，定义一个歌曲列表数组 song_list，该数组包括歌曲图片 URL、歌曲名称和歌曲简介等相关信息。具体代码如下：

```
<template>
  <div class="left">
    <ul v-for="value in song_list" :key="value">
      <li><img :src="value.imgUrl"></li>
      <li>{{value.name}}</li>
      <li>{{value.intro}}</li>
    </ul>
  </div>
</template>
```

```
<script>
export default {
  name: "SongList",
  data: function (){
    return {
      song_list: [                                          //歌曲列表数组
        {
          imgUrl: require('@/assets/images/1.jpg'),         //歌曲图片 URL
          name: '加州旅馆',                                  //歌曲名称
          intro: '老鹰乐队经典代表作'                        //歌曲简介
        },
        {
          imgUrl: require('@/assets/images/2.jpg'),
          name: 'I Want It That Way',
          intro: '后街男孩代表作之一'
        },
        {
          imgUrl: require('@/assets/images/3.jpg'),
          name: '此情可待',
          intro: '传唱不衰，颂扬真情的永恒'
        },
        {
          imgUrl: require('@/assets/images/4.jpg'),
          name: '昨日重现',
          intro: '卡朋特乐队经典代表作'
        },
        {
          imgUrl: require('@/assets/images/5.jpg'),
          name: 'Take Me To Your Heart',
          intro: '翻唱自歌神代表作《吻别》'
        },
        {
          imgUrl: require('@/assets/images/6.jpg'),
          name: '乡村路带我回家',
          intro: '乡村音乐歌手约翰·丹佛的成名作'
        },
        {
          imgUrl: require('@/assets/images/7.jpg'),
          name: 'My Love',
          intro: '西城男孩代表作之一'
        },
        {
          imgUrl: require('@/assets/images/8.jpg'),
          name: 'Unchained Melody',
          intro: '经典难忘的电影旋律'
        },
        {
          imgUrl: require('@/assets/images/9.jpg'),
          name: 'Lemon Tree',
          intro: '一首写在夏天里的歌'
        }
      ]
    }
  }
}
</script>
```

（2）在 style.css 文件中编写 CSS 代码，为歌曲列表组件设置样式。代码如下：

```
.main .left{
    float: left;                                            /*设置左浮动*/
    width: 720px;                                           /*设置宽度*/
}
.main .left img{
```

```
    width: 230px;                                    /*设置宽度*/
    height: 180px;                                   /*设置高度*/
}
.main .left ul{
    float: left;                                     /*设置左浮动*/
    width: 240px;                                    /*设置宽度*/
}
.main .left ul li:nth-child(2){
    height: 30px;                                    /*设置高度*/
    line-height: 30px;                               /*设置行高*/
    font-size: 18px;                                 /*设置文字大小*/
    margin-left: 3px;                                /*设置左外边距*/
}
.main .left ul li:nth-child(3){
    height: 30px;                                    /*设置高度*/
    line-height: 30px;                               /*设置行高*/
    font-size: 14px;                                 /*设置文字大小*/
    margin-left: 3px;                                /*设置左外边距*/
}
```

3.4.3 轮播图的设计

在首页主显示区右侧的最上方，有一个轮播图的展示界面。该界面有 3 张轮播图片，每张轮播图片下方都有 3 个数字按钮，用户可以通过单击其中一个数字按钮来切换到对应的图片。界面效果分别如图 3.4 和图 3.5 所示。

图 3.4 轮播图 1

图 3.5 轮播图 3

轮播图的实现步骤如下。

（1）在 components 目录中创建 MusicBanner.vue 文件。在 MusicBanner.vue 文件的<template>标签中定义轮播图片和用于切换图片的数字按钮。在<script>标签中定义数据、方法和钩子函数。在方法中，通过 next()方法设置下一张图片的索引，通过 toggle()方法设置当单击某个数字按钮后显示对应的图片。在 mounted 钩子函数中使用 setInterval()方法每隔 3 秒调用一次 next()方法，以实现图片自动轮播的效果。具体代码如下：

```
<template>
  <div class="right">
    <!--切换的图片-->
    <div class="banner">
      <transition-group name="effect" tag="div">
        <span v-for="(v,i) in bannerURL" :key="i" v-show="(i+1)===index?true:false">
          <img :src="v">
        </span>
      </transition-group>
```

```
    </div>
    <!--切换的小按钮-->
    <ul class="numBtn">
      <li v-for="num in 3" :key="num">
        <a href="javascript:;" :style="{background:num===index?'#ff9900':'#CCCCCC'}"
          @click='toggle(num)' class='num'>{{num}}</a>
      </li>
    </ul>
  </div>
</template>

<script>
export default {
  name: "MusicBanner",
  data: function (){
    return {
      bannerURL : [
        require('@/assets/images/10.jpg'),
        require('@/assets/images/11.jpg'),
        require('@/assets/images/12.jpg')
      ],                                                    //图片 URL 数组
      index : 1,                                            //图片的索引
      flag : true,                                          //是否可以单击数字按钮
      timer : null,                                         //定时器 ID
    }
  },
  methods: {
    next : function(){
      this.index = this.index + 1 === 4 ? 1 : this.index + 1;
    },
    toggle : function(num){
      //单击按钮切换到对应图片
      if(this.flag){
        this.flag = false;
        //过 1 秒后可以再次单击按钮切换图片
        setTimeout(()=>{
          this.flag = true;
        },1000);
        this.index = num;                                   //切换为选中的图片
        clearTimeout(this.timer);                           //取消定时器
        //过 3 秒图片轮换
        this.timer = setInterval(this.next,3000);
      }
    }
  },
  mounted : function(){
    this.timer = setInterval(this.next,3000);               //过 3 秒图片轮换
  }
}
</script>
```

（2）在 style.css 文件中编写 CSS 代码，为轮播图组件设置样式。代码如下：

```
.right{
    float: right;                                           /*设置右浮动*/
    position: relative;                                     /*设置相对定位*/
    width: 260px;                                           /*设置宽度*/
}
.banner img{
    width: 260px;                                           /*设置宽度*/
}
.banner{
    position: relative;                                     /*设置相对定位*/
```

```
    height: 260px;                          /*设置高度*/
}
.banner span{
    position: absolute;                     /*设置绝对定位*/
    top:0;                                  /*设置距离父元素顶部的距离*/
    left: 0;                                /*设置距离父元素左侧的距离*/
}
.numBtn{
    width: 200px;                           /*设置宽度*/
    position:absolute;                      /*设置绝对定位*/
    left:50%;                               /*设置距离父元素左侧的距离*/
    top:86%;                                /*设置距离父元素顶部的距离*/
    text-align:center;                      /*设置文字居中显示*/
}
.numBtn li{
    list-style:none;                        /*设置列表无样式*/
    border-radius: 50%;                     /*设置圆角边框*/
    float:left;                             /*设置左浮动*/
}
.numBtn li a{
    display: block;                         /*设置块状元素*/
    width: 20px;                            /*设置宽度*/
    height: 20px;                           /*设置高度*/
    line-height: 20px;                      /*设置行高*/
    border-radius: 50%;                     /*设置圆角边框*/
    margin: 5px;                            /*设置外边距*/
    color:#FFFFFF;                          /*设置文字颜色*/
    font-weight:bolder;                     /*设置文字粗细*/
    text-decoration:none;                   /*设置文字无下画线*/
}
.numBtn li a.num{
    transition:all .6s ease;                /*设置按钮过渡效果*/
}
/*设置过渡属性*/
.effect-enter-active, .effect-leave-active{
    transition: all 1s;
}
.effect-enter-from, .effect-leave-to{
    opacity: 0;
}
```

3.4.4 歌曲排行榜

在首页主显示区右侧的中间是歌曲排行榜的展示界面。这里使用了选项卡切换的效果。当单击“推荐歌曲”选项卡时，该选项卡下方会显示推荐歌曲列表；当单击“热门歌曲”选项卡时，该选项卡下方会显示热门歌曲列表。界面效果分别如图 3.6 和图 3.7 所示。

图 3.6 推荐歌曲列表

图 3.7 热门歌曲列表

歌曲排行榜展示界面的实现步骤如下。

（1）在 components 目录中创建 tab 文件夹，并在该文件夹下创建两个与选项卡对应的歌曲列表组件，分别是推荐歌曲列表组件 RecommendTab.vue 和热门歌曲列表组件 HotTab.vue。

推荐歌曲列表组件的代码如下：

```
<template>
  <div class="songlist">
    <div v-for="(item,index) in recommend" :key="index">
      <span>{{++index}}</span>
      <span>{{item.name}}</span>
      <span>{{item.singer}}</span>
    </div>
  </div>
</template>
<script>
export default {
  name: "RecommendTab",
  data: function (){
    return {
      recommend : [                                                    //推荐歌曲数组
        { name : '寂静之声', singer : '保罗・西蒙' },
        { name : '英雄', singer : '玛丽亚・凯莉' },
        { name : '说你、说我', singer : '莱昂纳尔・里奇' },
        { name : '卡萨布兰卡', singer : '贝蒂・希金斯' },
        { name : '加州旅馆', singer : '老鹰乐队' },
        { name : '嘿，朱迪', singer : '披头士乐队' },
        { name : '尊重', singer : '奥蒂斯・雷丁' },
        { name : '你若即若离', singer : 'U2 乐队' }
      ]
    }
  }
}
</script>
```

热门歌曲列表组件的代码如下：

```
<template>
  <div class="songlist">
    <div v-for="(item,index) in hot" :key="index">
      <span>{{++index}}</span>
      <span>{{item.name}}</span>
      <span>{{item.singer}}</span>
    </div>
  </div>
</template>
<script>
export default {
  name: "HotTab",
  data: function (){
    return {
      hot : [                                                          //热门歌曲数组
        { name : '我希望它保持那样', singer : '后街男孩' },
        { name : '想念你', singer : '滚石乐队' },
        { name : '快乐的结局', singer : '艾薇儿・拉维尼' },
        { name : '寂静之声', singer : '保罗・西蒙' },
        { name : '我的爱', singer : '西城男孩' },
        { name : '管不住的音符', singer : '丹尼尔・波特' },
        { name : '情歌', singer : '阿黛尔・阿德金斯' },
        { name : '远离家乡', singer : '舞动精灵乐团' }
      ]
    }
  }
}
</script>
```

（2）在 components 目录中创建 MusicTabs.vue 文件。在该文件的<template>标签中，定义“推荐歌曲”和“热门歌曲”选项卡，并应用<component>元素将 data 中的 curtab 属性动态地绑定到它的 is 属性上。在

<script>标签中，引入两个歌曲列表组件，并定义数据和方法，以便对这两个歌曲列表组件进行注册。具体代码如下：

```
<template>
  <div class="tabs">
    <div class="top">
      <div class="title">歌曲排行榜</div>
      <ul class="tab">
        <li :class="{actived : actived}" v-on:mouseover="toggleAction('recommend')">推荐歌曲</li>
        <li :class="{actived : !actived}" v-on:mouseover="toggleAction('hot')">热门歌曲</li>
      </ul>
    </div>
    <component :is="curtab"></component>
  </div>
</template>
<script>
import recommend from "@/components/tab/RecommendTab";
import hot from "@/components/tab/HotTab";
export default {
  name: "MusicTabs",
  data: function (){
    return {
      actived : true,                                          //是否使用标签类名
      curtab : 'recommend'                                     //歌曲排行榜组件名
    }
  },
  methods: {
    toggleAction : function(value){
      this.curtab = value;                                     //获取当前组件名
      value === 'recommend' ? this.actived = true : this.actived = false;
    }
  },
  components: {
    recommend,
    hot
  }
}
</script>
```

（3）在 style.css 文件中编写 CSS 代码，为歌曲排行榜组件设置样式。代码如下：

```
.tabs{
    float: right;                                              /*设置右浮动*/
    width:260px;                                               /*设置宽度*/
    margin:20px auto;                                          /*设置外边距*/
}
.top{
    height:26px;                                               /*设置高度*/
    line-height: 26px;                                         /*设置行高*/
}
.title{
    display:inline-block;                                      /*设置行内块元素*/
    font-size:18px;                                            /*设置文字大小*/
}
ul.tab{
    display:inline-block;                                      /*设置行内块元素*/
    list-style:none;                                           /*设置列表样式*/
    margin-left:20px;                                          /*设置左外边距*/
}
ul.tab li{
    margin: 0;                                                 /*设置外边距*/
    padding: 0;                                                /*设置内边距*/
    float:left;                                                /*设置左浮动*/
```

```
    width:80px;                                    /*设置宽度*/
    height: 26px;                                  /*设置高度*/
    line-height: 26px;                             /*设置行高*/
    font-size:14px;                                /*设置文字大小*/
    cursor:pointer;                                /*设置鼠标光标形状*/
    text-align:center;                             /*设置文本居中显示*/
}
ul.tab li.actived{
    display:block;                                 /*设置块元素*/
    width:60px;                                    /*设置宽度*/
    height: 26px;                                  /*设置高度*/
    line-height: 26px;                             /*设置行高*/
    background-color:#66CCFF;                      /*设置背景颜色*/
    color:#FFFFFF;                                 /*设置文字颜色*/
    cursor:pointer;                                /*设置鼠标光标形状*/
}
.songlist{
    clear:both;                                    /*设置清除浮动*/
    margin-top:10px;                               /*设置上外边距*/
}
.songlist div{
    width:260px;                                   /*设置宽度*/
    height:31px;                                   /*设置高度*/
    line-height:31px;                              /*设置行高*/
    border-bottom-width: 1px;                      /*设置下边框宽度*/
    border-bottom-style: dashed;                   /*设置下边框样式*/
    border-bottom-color: #333333;                  /*设置下边框颜色*/
    background-color: #FFFFFF;                     /*设置背景颜色*/
    font-size:14px;                                /*设置文字大小*/
}
.songlist div span{
    margin-left:10px;                              /*设置左外边距*/
}
.songlist div span:last-child{
    float:right;                                   /*设置右浮动*/
    margin-right:10px;                             /*设置右外边距*/
}
```

3.4.5 最新音乐资讯

在首页主显示区右侧的最下方是最新音乐资讯的展示界面，其中包含 5 条音乐资讯，这些资讯从下到上间断地循环滚动。为了展示这种滚动效果，该界面一次只展示 4 条音乐资讯。具体的界面效果如图 3.8 所示。

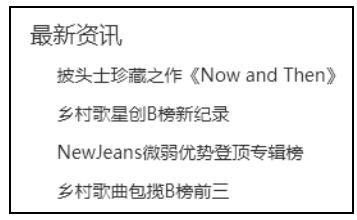


图 3.8 最新音乐资讯的展示界面

最新音乐资讯的展示界面使用 axios 发送请求和获取响应数据。因此，首先需要安装并引入 axios。实现最新音乐资讯的展示界面步骤如下。

（1）在 components 目录中创建 MusicNews.vue 文件。在该文件的<template>标签中定义一个 ul 列表，并将该列表的类名和定义的数据进行绑定。当鼠标移入列表时，触发 mouseenter 事件并调用 stop()方法；当鼠标移出列表时，触发 mouseleave 事件并调用 up()方法。在列表项中，使用 v-for 指令对音乐资讯列表 news_list 进行遍历。在<script>标签中定义数据、方法和钩子函数，在钩子函数中，使用 axios 发送 GET 请求，以获取 data.json 中的音乐资讯列表数据。代码如下：

```
<template>
  <div class="news">
    <div class="news_title">最新资讯</div>
    <div class="scroll">
      <ul class="list" :class="{anim:animate}" @mouseenter="stop" @mouseleave="up">
```

```
            <li v-for="value in news_list" :key="value">{{value}}</li>
        </ul>
    </div>
  </div>
</template>
<script>
import axios from 'axios'
export default {
  name: "MusicNews",
  data: function (){
    return {
      animate:false,                                    //是否使用指定类名
      timerID: null,                                    //定时器 ID
      news_list:[],                                     //最新资讯列表
    }
  },
  methods: {
    scrollUp: function (){
      var t = this;
      this.timerID = setInterval(function (){
        t.animate = true;                               //添加指定类名
        setTimeout(function (){
          t.news_list.push(t.news_list[0]);             //将数组第一个元素添加到数组末尾
          t.news_list.shift();                          //删除数组的第一个元素
          t.animate = false;                            //移除指定类名
        },500)
      }, 2000);
    },
    stop: function() {
      clearInterval(this.timerID);                      //停止向上滚动操作
    },
    up: function() {
      this.scrollUp();                                  //执行向上滚动操作
    }
  },
  mounted : function(){
    axios.get('/data.json').then(function (response){   //发送 GET 请求
      this.news_list = response.data;                   //获取响应数据
    }.bind(this));
    this.scrollUp();                                    //执行向上滚动操作
  }
}
</script>
```

说明

为了能正常获取 data.json 文件中的数据，需要将该文件放在 public 文件夹中。

（2）在 style.css 文件中编写 CSS 代码，为最新音乐资讯组件设置样式。代码如下：

```
.news{
    float: right;                                       /*设置右浮动*/
    width:260px;                                        /*设置宽度*/
    margin:0 auto;                                      /*设置外边距*/
}
.news_title{
    font-size:18px;                                     /*设置文字大小*/
}
.scroll{
    margin-left:5px;                                    /*设置左外边距*/
    margin-top:5px;                                     /*设置上外边距*/
    width:260px;                                        /*设置宽度*/
    height:120px;                                       /*设置高度*/
```

```
    overflow:hidden;                                    /*设置溢出内容隐藏*/
}
.scroll li{
    width:260px;                                        /*设置宽度*/
    height:30px;                                        /*设置高度*/
    line-height:30px;                                   /*设置行高*/
    margin-left:16px;                                   /*设置左外边距*/
}
.scroll li a{
    font-size:14px;                                     /*设置文字大小*/
    color:#333;                                         /*设置文字颜色*/
    text-decoration:none;                               /*设置链接文字无下画线*/
}
.scroll li a:hover{
    color:#66CCFF;                                      /*设置文字颜色*/
}
.anim{
    transition: all 0.5s;                               /*设置过渡效果*/
    margin-top: -30px;                                  /*设置上外边距*/
}
```

3.4.6 新歌首发

在歌曲列表展示界面的下方是新歌首发的展示界面。由于歌曲数量较多，因此采用分页的方式来展示首发歌曲的信息。具体的界面效果分别如图 3.9 和图 3.10 所示。

图 3.9 第一页新歌首发界面

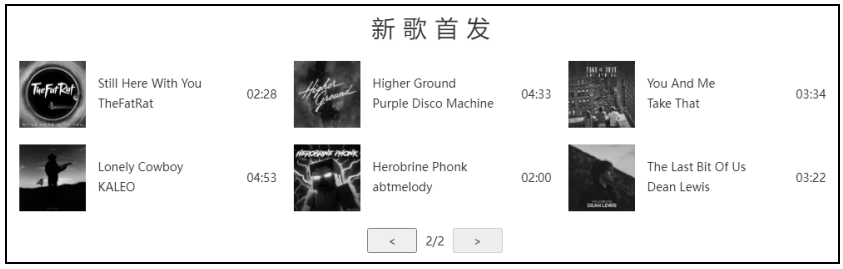


图 3.10 第二页新歌首发界面

新歌首发界面的实现步骤如下。

（1）在 components 目录中创建 NewSong.vue 文件。在该文件的<template>标签中，调用 DataPage 组件，并将首发歌曲列表和每页显示的歌曲数量作为 Prop 属性进行传递给该组件。在<script>标签中，首先使用 import 引入 DataPage 组件，然后在 components 选项中注册该组件，并定义首发歌曲列表数组。代码如下：

```
<template>
  <DataPage :items = "dataList" :pageSize = "6" />
</template>
<script>
import DataPage from "@/components/DataPage.vue";
```

```
export default {
  name: "NewSong",
  components: {
    DataPage
  },
  data: function (){
    return {
      dataList: [                                          //首发歌曲列表数组
        {
          imgUrl: require('@/assets/images/new/1.jpg'),    //歌曲图片 URL
          songName: 'Co-Pathetic',                         //歌曲名称
          singer: 'Novo Amor',                             //歌手名称
          duration: '03:55'                                //歌曲时长
        },
        {
          imgUrl: require('@/assets/images/new/2.jpg'),
          songName: 'Like That (Explicit)',
          singer: 'Future',
          duration: '04:27'
        },
        {
          imgUrl: require('@/assets/images/new/3.jpg'),
          songName: 'Run Run Run',
          singer: 'The Libertines',
          duration: '02:53'
        },
        {
          imgUrl: require('@/assets/images/new/4.jpg'),
          songName: 'Been Like This',
          singer: 'Meghan Trainor /T-Pain',
          duration: '02:25'
        },
        {
          imgUrl: require('@/assets/images/new/5.jpg'),
          songName: 'Team Side feat.RCB',
          singer: 'Alan Walker /Sofiloud',
          duration: '03:00'
        },
        {
          imgUrl: require('@/assets/images/new/6.jpg'),
          songName: 'Count Me Out',
          singer: 'Vicetone /Emily Falvey',
          duration: '03:13'
        },
        {
          imgUrl: require('@/assets/images/new/7.jpg'),
          songName: 'Still Here With You',
          singer: 'TheFatRat',
          duration: '02:28'
        },
        {
          imgUrl: require('@/assets/images/new/8.jpg'),
          songName: 'Higher Ground',
          singer: 'Purple Disco Machine',
          duration: '04:33'
        },
        {
          imgUrl: require('@/assets/images/new/9.jpg'),
          songName: 'You And Me',
          singer: 'Take That',
          duration: '03:34'
        },
        {
          imgUrl: require('@/assets/images/new/10.jpg'),
```

```
            songName: 'Lonely Cowboy',
            singer: 'KALEO',
            duration: '04:53'
          },
          {
            imgUrl: require('@/assets/images/new/11.jpg'),
            songName: 'Herobrine Phonk',
            singer: 'abtmelody',
            duration: '02:00'
          },
          {
            imgUrl: require('@/assets/images/new/12.jpg'),
            songName: 'The Last Bit Of Us',
            singer: 'Dean Lewis',
            duration: '03:22'
          }
        ]
      }
    }
}
</script>
```

（2）在 components 目录中创建 DataPage.vue 文件。在该文件的<template>标签中，使用 v-for 指令遍历当前页的首发歌曲列表信息，包括歌曲图片、歌曲名称、歌手名称和歌曲时长。同时，添加“<”和“>”按钮进行分页控制。单击“<”按钮会调用 prevPage()方法，单击“>”按钮会调用 nextPage()方法。在<script>标签中定义父组件传递的 Prop 属性、数据、计算属性和方法。totalPages 计算属性用于获取总页数，currentPageItems 计算属性用于获取当前页的首发歌曲列表，prevPage()方法和 nextPage()方法分别实现当前页减 1 和加 1 的操作。代码如下：

```
<template>
  <div class="page">
    <div class="page_title">新歌首发</div>
    <div class="page_item" v-for="item in currentPageItems" :key="item">
      <div class="img"><img :src="item.imgUrl"></div>
      <div class="name">
        <div>{{item.songName}}</div>
        <div>{{item.singer}}</div>
      </div>
      <div class="duration">{{item.duration}}</div>
    </div>
    <div class="page_control">
      <button @click="prevPage" :disabled = "currentPage === 1">&lt;</button>
      <span>{{currentPage}}/{{totalPages}}</span>
      <button @click="nextPage" :disabled = "currentPage === totalPages">&gt;</button>
    </div>
  </div>
</template>
<script>
export default {
  name: "DataPage",
  props: ['items','pageSize'],
  data: function (){
    return {
      currentPage: 1                                          //当前页
    }
  },
  computed: {
    totalPages(){                                             //总页数
      return Math.ceil(this.items.length / this.pageSize);
    },
    currentPageItems(){                                       //当前页数据
```

```
            let start = (this.currentPage - 1) * this.pageSize;
            let end = start + this.pageSize;
            return this.items.slice(start, end);
        }
    },
    methods: {
        prevPage(){
            if(this.currentPage > 1) this.currentPage -= 1;                    //当前页减 1
        },
        nextPage(){
            if(this.currentPage < this.totalPages) this.currentPage += 1;      //当前页加 1
        }
    }
}
</script>
```

（3）在 style.css 文件中编写 CSS 代码，为新歌首发组件设置样式。代码如下：

```
.page{
    clear: both;                                                               /*清除浮动*/
}
.page .page_title{
    height: 80px;                                                              /*设置高度*/
    line-height: 80px;                                                         /*设置行高*/
    font-size:28px;                                                            /*设置文字大小*/
    letter-spacing: 10px;                                                      /*设置文字间距*/
    text-align: center;                                                        /*设置文本水平居中显示*/
}
.page .page_item{
    float: left;                                                               /*设置左浮动*/
    width: 33%;                                                                /*设置宽度*/
    height: 100px;                                                             /*设置高度*/
}
.page img{
    width: 80px;                                                               /*设置宽度*/
    height: 80px;                                                              /*设置高度*/
}
.page .page_item>div{
    float: left;                                                               /*设置左浮动*/
}
.page .page_item .name{
    margin-top: 12px;                                                          /*设置上外边距*/
    margin-left: 15px;                                                         /*设置左外边距*/
}
.page .page_item .name div{
    margin-top: 5px;                                                           /*设置上外边距*/
}
.page .page_item .duration{
    float: right;                                                              /*设置右浮动*/
    margin-top: 30px;                                                          /*设置上外边距*/
    margin-right: 20px;                                                        /*设置右外边距*/
}
.page .page_control{
    clear: both;                                                               /*清除浮动*/
    text-align: center;                                                        /*设置文本水平居中显示*/
    margin-bottom: 10px;                                                       /*设置下外边距*/
}
.page .page_control span{
    margin: 0 10px;                                                            /*设置外边距*/
}
.page .page_control button{
    width: 60px;                                                               /*设置宽度*/
    height: 30px;                                                              /*设置高度*/
}
```

3.4.7 首页底部的设计

网站首页的底部展示了版权等方面的信息。首页底部的效果如图 3.11 所示。

图 3.11 首页底部效果

首页底部的实现步骤如下。

（1）在 components 目录中创建 MusicBottom.vue 文件。在该文件的<template>标签中，定义 ul 列表，并在该列表中使用 v-for 指令对版权信息列表 b_list 进行遍历。在<script>标签中，定义版权信息列表 b_list。具体代码如下：

```
<template>
  <div class="bottom">
    <ul>
      <li v-for="value in b_list" :key="value">{{value}}</li>
    </ul>
    <div class="copyright">© 2023-2050 时光音乐网 版权所有</div>
  </div>
</template>
<script>
export default {
  name: "MusicBottom",
  data: function (){
    return {
      b_list: ['歌曲入库','版权声明','联系我们','历史合作','友情链接','帮助中心']
    }
  }
}
</script>
```

（2）在 style.css 文件中编写 CSS 代码，为页面底部组件设置样式。代码如下：

```
.bottom{
    width: 1000px;                              /*设置宽度*/
    height: 80px;                               /*设置高度*/
    margin: 0 auto;                             /*设置外边距*/
    border-top: 1px solid #3366FF;              /*设置上边框*/
    text-align: center;                         /*设置文本水平居中显示*/
}
.bottom ul{
    padding-top: 15px;                          /*设置上内边距*/
    margin-bottom: 10px;                        /*设置下外边距*/
}
.bottom li{
    display: inline-block;                      /*设置行内块元素*/
    width: 80px;                                /*设置宽度*/
    height: 16px;                               /*设置高度*/
    line-height: 16px;                          /*设置行高*/
    text-align: center;                         /*设置文本水平居中显示*/
}
.bottom li:not(:last-child){
    border-right: 1px solid #BBBBBB;            /*设置右边框*/
}
```

3.4.8 在根组件中构建音乐网首页

修改 App.vue 文件，首先在<script>标签中使用 import 引入构建音乐网首页的多个组件，然后在 components 选项中注册各个组件，并在<template>标签中调用各个组件，最后在<style>标签中引入公共 CSS

文件 style.css。代码如下：

```
<template>
  <div id="app">
    <div class="logo"><img src="./assets/images/ball.png"></div>
    <MusicNav/>
    <div class="main">
      <SongList/>
      <MusicBanner/>
      <MusicTabs/>
      <MusicNews/>
      <NewSong/>
    </div>
    <MusicBottom/>
  </div>
</template>
<script>
import MusicNav from './components/MusicNav.vue'
import SongList from './components/SongList.vue'
import MusicBanner from './components/MusicBanner.vue'
import MusicTabs from './components/MusicTabs.vue'
import MusicNews from './components/MusicNews.vue'
import NewSong from "@/components/NewSong.vue";
import MusicBottom from './components/MusicBottom.vue'
export default {
  name: 'App',
  components: {
    MusicNav,                                    //网站导航栏组件
    SongList,                                    //歌曲列表组件
    MusicBanner,                                 //轮播图组件
    MusicTabs,                                   //选项卡组件
    MusicNews,                                   //最新资讯组件
    NewSong,                                     //新歌首发组件
    MusicBottom                                  //页面底部组件
  }
}
</script>
<style lang="scss">
@import 'assets/css/style.css';
</style>
```

3.5 项 目 运 行

通过前述步骤，我们已经设计并完成了“时光音乐网首页设计”项目的开发。接下来，我们运行该项目，以检验我们的开发成果。首先打开命令提示符窗口，切换到项目所在目录，然后执行 npm run serve 命令运行该项目，如图 3.12 所示。

在浏览器地址栏中输入 http://localhost:8080 并按 Enter 键后，即可成功运行该项目，具体效果如图 3.13 所示。

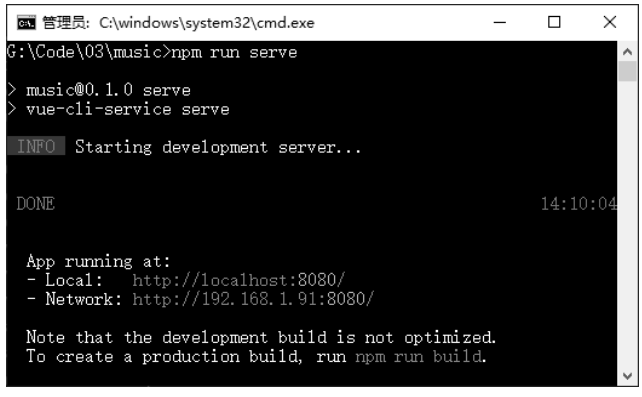


图 3.12 运行项目

图 3.13　时光音乐网首页设计效果

3.6　源 码 下 载

本章虽然详细地讲解了如何编码实现“时光音乐网首页设计”的各个功能，但给出的代码都是代码片段，而非完整的源代码。为了方便读者学习，本书提供了该项目的完整源代码，读者可以扫描右侧的二维码进行下载。

源码下载