武汉纺织大学

Web应用开发课程设计

**天气预报**

**学 院： 数学与计算机学院**

**班 级： 物联网11802**

**姓 名： 胡世杰**

**学 号： 1804240912**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年12月15日**

目 录

[1 需求分析 1](#_Toc59968863)

[1.1 数据来源 1](#_Toc59968864)

[1.2 数据呈现 1](#_Toc59968865)

[1.3各项功能 1](#_Toc59968866)

[1.3.1 切换城市 1](#_Toc59968867)

[1.3.2 显示当日天气 1](#_Toc59968868)

[1.3.3 显示未来七天天气 1](#_Toc59968869)

[1.3.4 绘制响应折线图 1](#_Toc59968870)

[1.3.5 根据气象情况动态改变背景图片 1](#_Toc59968871)

[1.3.6 根据更新时间动态改变背景图片 2](#_Toc59968872)

[1.3.7 背景图片淡入效果 2](#_Toc59968873)

[1.3.8 为背景图片添加遮罩层 2](#_Toc59968874)

[1.3.9 为输入框动态修改背景颜色 2](#_Toc59968875)

[1.3.10 动态改变天气图标 2](#_Toc59968876)

[1.3.11 各种图标的点击效果 2](#_Toc59968877)

[1.3.12 小贴士旋转过渡 2](#_Toc59968878)

[1.3.13 背景音乐 3](#_Toc59968879)

[2 系统实现 —— 后端 3](#_Toc59968880)

[2.1 项目结构 3](#_Toc59968881)

[2.2 配置文件 3](#_Toc59968882)

[2.3 控制器 4](#_Toc59968883)

[3 系统实现 —— 前端 6](#_Toc59968884)

[3.1 项目结构 6](#_Toc59968885)

[3.2 前端构建 7](#_Toc59968886)

[3.3 网络请求 8](#_Toc59968887)

[3.4 echart图表绘制 9](#_Toc59968888)

[4 系统测试 10](#_Toc59968889)

[5 系统总结 10](#_Toc59968890)

# 1 需求分析

设计一个天气预报页面，具体实现功能有：显示当日天气，显示七天天气，切换城市，绘制响应折线图等，具体细节如下：

## 1.1 数据来源

整个项目设计的数据来源于天气预报API：<http://www.tianqiapi.com/index/doc?version=v1>

这个API可以免费试用2000次，提供的数据也比较多，当然，如果掌握爬虫技术可以去中国气象局获取数据，中国气象局的数据是允许获取的。

## 1.2 数据呈现

当我们拿到数据之后，我们需要将数据呈现在页面上，有些页面可能是死数据，比如新闻信息，一旦加载出来就不会改变了，但是天气预报用到的数据是需要动态变化的，比如我们在页面上点击一下未来七天天气中的任何一天，我想要改变折线图，根据天气情况改变图片背景，那么在不刷新页面的情况下，就需要数据能够动态发生变化，这里用到的核心技术就是“响应式”，前端实现响应式我采用的是响应式框架“Vue”

## 1.3各项功能

### 1.3.1 切换城市

页面加载之初，页面上并没有切换城市的输入框，当用户点击“切换城市”之后，页面上出现输入框，用户在输入框中输入相应城市并敲击回车或者点击查找按钮之后，天气预报的城市自动切换为输入的城市。

### 1.3.2 显示当日天气

页面加载之初默认显示IP地址所在区域的天气情况，当用户通过“切换城市”功能切换城市之后，显示对应城市的当日天气。其中相应的小信息点，比如日出时间，日落时间，空气质量等。

### 1.3.3 显示未来七天天气

页面加载之初默认显示IP地址所在区域未来七天天气情况，当用户通过“切换城市”功能切换城市之后，显示对应城市未来七天的天气情况。

### 1.3.4 绘制响应折线图

页面加载之初，通过echart绘制当前城市未来七天的气温趋势图，当用户点击未来七天预报的任意一天，则echart将绘制响应天的24小时气温变化图，当用户点击当日天气上的小信息点时，echart将绘制响应的信息变化图，如“未来七天的日出时间变化图”，“未来七天的日落时间变化图”，“未来七天的空气污染指数变化图”等等。

### 1.3.5 根据气象情况动态改变背景图片

根据“当前天气”的气象情况，动态改变背景图片，分别对应“阴”“晴”“雨”“雪”“雷”“多云”“沙尘”“冰雹”“大雾”九种气象情况。

### 1.3.6 根据更新时间动态改变背景图片

根据api天气数据更新的时间，动态改变背景图片，九种气象图片，分别有两种主题，“白天”和“黑夜”，当前天气数据如果在18：00 ~ 6:00内更新，那么背景图片显示对应气象的“黑夜”版本，反之显示“白天”版本。

### 1.3.7 背景图片淡入效果

不同的日期有不同的天气情况，不同的天气情况对应不同的背景图片，所以，当我们切换日期的时候，往往背景图片也会跟着切换，如果不进行特殊的处理，一下子从一张图片变换到另外一张图片是非常突兀的，所以这里采用了背景淡入的处理，目的是为了解决背景图片切换时的突兀感，提升用户体验

### 1.3.8 为背景图片添加遮罩层

“白天”主题图片饱和度普遍偏高，导致白色主题的天气预报不能很好的呈现出来，所以，当整个页面处于“白天”主题的时候，动态添加遮罩层，降低背景图片的饱和度，让主题的天气数据能够更好的呈现出来。

### 1.3.9 为输入框动态修改背景颜色

“黑夜”主题图片普遍饱和度偏低，导致背景颜色为白色的搜索框显得非常的突兀，所以，当整个页面处于“黑夜”主题的时候，动态修改输入框背景颜色为黑色，让输入框在显示出来时候没有那么强的反差突兀感，提升用户体验感。

### 1.3.10 动态改变天气图标

根据不同的九种天气情况，找到九种匹配的天气图标，根据每天不同的天气情况动态选择对应的天气图标进行显示。

### 1.3.11 各种图标的点击效果

为了提升界面的交互性，希望在用户进行点击的时候，能有一种点击效果，这种点击效果，在本页面中通过“鼠标悬浮”，“鼠标按下”，“鼠标抬起”三种事件改变样式实现

### 1.3.12 小贴士旋转过渡

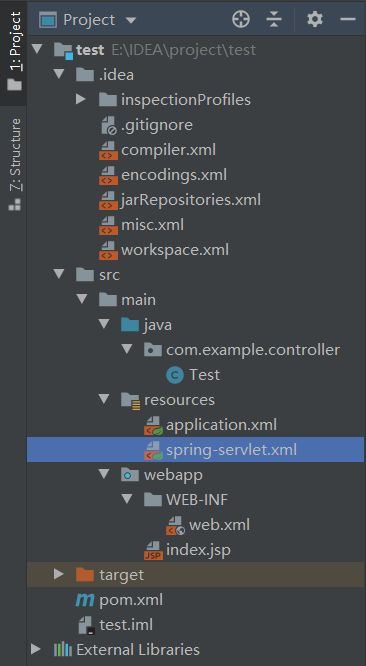
小贴士在设计之初希望达成的显示效果是，界面上列出各条小贴士和对应的等级情况，如“空气质量 等级：优”，然后，当鼠标悬浮在某一个小贴士上的时候，小贴士能够自动切换为对应的“建议”，例如“空气真好，适合外出运动”，这样一方面提升了页面提供的信息量，一方面增加了界面的交互性，提升了用户的体验感。但是在实现功能之后，发现在文字从“小贴士等级”切换到“小贴士建议”的过程中，没有任何的过渡效果，导致文字直接切换看上去非常的突兀，不好看，于是在hover时间中添加了响应的过渡动画，尝试了很多种过渡动画，包括2D旋转，3D旋转等，最终发现最简单的普通旋转好像效果是最好的，于是选定了这种过渡方式

### 1.3.13 背景音乐

在上述所有功能设计完毕之后，还是觉得页面少了一些什么东西，最终决定在页面上加上背景音乐，不过有点遗憾的是，并没有实现音乐的自动播放，因为谷歌出于用户体验，担心用户浪费掉不必要的流量而屏蔽掉了自动播放这个功能…

# 2 系统实现 —— 后端

## 2.1 项目结构



## 2.2 配置文件

在项目下新建Source Folder，名为resources，在创建application.xml配置文件

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans-3.0.xsd

http://www.springframework.org/schema/context

http://www.springframework.org/schema/context/spring-context-3.0.xsd ">

<!-- 下面是配置扫描包的位置,包名为com.ework.upms.server,也就是说,我们的试图解析器应该放在com.ework.upms.server包下. -->

<context:component-scan base-package="com.example.controller"/>

<bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<!-- 前缀,我们的视图文件应该放到/WEB-INF/views/目录下,这里我们需要在WEB-INF下面创建view文件夹 -->

<property name="prefix" value="/WEB-INF/views/" />

<!-- 设置后缀为.jsp -->

<property name="suffix" value=".jsp" />

</bean>

</beans>

## 2.3 控制器

发送请求，调用api接口，作为二传手将api的数据传送到前端

package com.example.controller;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.\*;

import java.io.\*;

import java.net.\*;

import java.util.Map;

@Controller

@CrossOrigin

public class Test {

@RequestMapping(value = "weather.api",method = RequestMethod.GET)

@ResponseBody

public String test(@RequestParam (value = "city",required = false, defaultValue = "")String city) throws IOException {

//外接口路径

String urlStr="http://v0.yiketianqi.com/api";

//get请求参数拼接

String params = "?appid=22266748&appsecret=pTIzTk6N&version=v9&vue=1";

String cityDecoded = "";

if(city != "") {

cityDecoded = "&city="+URLEncoder.encode(city,"UTF-8");

}

String urlStrFinal = urlStr + params + cityDecoded;

StringBuffer result = new StringBuffer();

URL url = new URL(urlStrFinal);

URLConnection urlConnection = url.openConnection();

InputStream inputStream = urlConnection.getInputStream();

BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(inputStream));

String line = "";

while((line = bufferedReader.readLine()) !=null){

result.append(line);

}

bufferedReader.close();

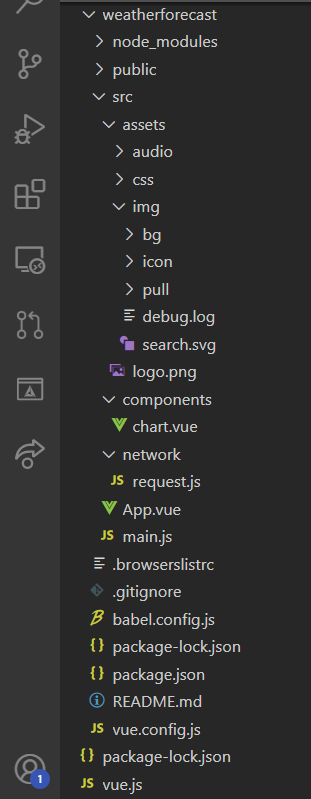
return result.toString();

}

}

# 3 系统实现 —— 前端

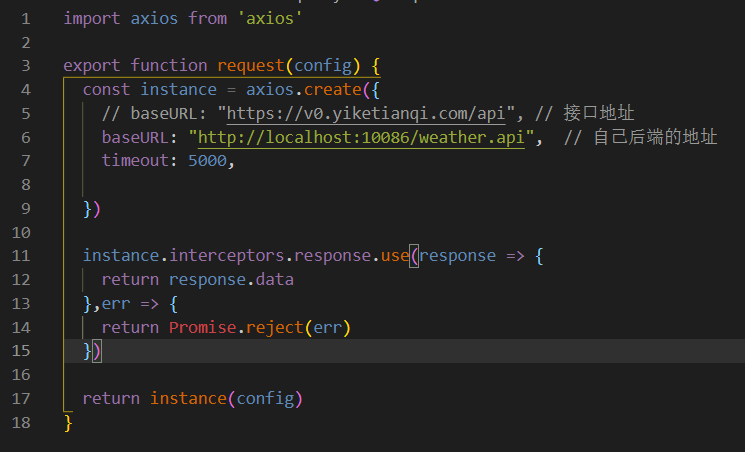
## 3.1 项目结构



## 3.2 前端构建



## 3.3 网络请求



## 3.4 echart图表绘制



# 4 系统测试

测试结果录制了一个小视频，见此目录下的“录制视频”

# 5 系统总结

主要问题：

因为天气预报这个课设题目其实后端是调用的API接口，工作量小，所以我把重心放在了前端上，而前端，我认为重点就是给用户一个良好的使用体验，所以我更加注重“背景图片的选取”，“过渡动画”，“点击效果”等美观或者有交互的设计，其中给我感受最深的就是，每个人的审美不同，当我选出一组背景图片给其他同学看的时候，有的人认为非常好看，而有的同学认为完全不行。光背景图片的选取我加在一起就花费了不下两天的时间。

解决过程：

一开始陷入了误区，我选择背景图片的时候完全没有考虑到他只是一张背景图片，而不是摄影作品，背景图片不同于摄影背景的区别就在于，摄影作品，我们所有的重心都放在这一张图片上，他是所有内容的呈现，并不单纯，有很多层次。而背景图片并不是整个页面的主题，我们的重心在天气预报的呈现上，我们不需要那么多的花花草草，其实越单纯的天气图片，用户越容易把重心放在天气预报的内容上，这样整体看上去更加主次分明，效果更好。后面我在找图片的时候找的都是一些很单纯的天气图片，没有去找华丽好看的摄影作品，摄影作品也许作为电脑桌面背景很适合，但是作为我这个天气预报页面的背景不太恰当。

系统存在的不足：

存在不足的地方确实是有几点的，

1. 在设置背景淡入动画的时候，动画执行时间是两秒，动画结束之后，才会去更新动画效果，那么这就出现一个问题，如果用户在两秒的冷却时间内疯狂的切换城市或者某一天的天气，那么淡入动画也只能显示一次，并不能跟上用户的速度。
2. 背景音乐因为谷歌出于用户方面的考虑，导致在谷歌中无法实现自动播放，而其实是否自动播放对于用户体验的影响是非常大的，如果天气预报界面在进来的时候能够自动播放背景音乐，那么用户会有很好的体验感，但是如果还需要用户自己去点击然后才能播放音乐，一方面用户可能并不会去点击，那么这个功能形同虚设，另外一方面就算用户手动点击了音乐的播放按钮，那也只是一个简单的音乐播放，没有了那种“惊喜”感，一切都在用户的意料之中，其实体验是比较差的。

课程的收获：

Web应用开发当之无愧是本学期收获最大的课程，不管是后台的springMVC框架，还是通过老师建议之后自学的前端Vue框架，都是非常好也非常流行的开发框架，并且Web本身就是一种开发出来很让人有成就感的东西，在学会了Vue框架的使用并且制作完这次的课设内容之后，我觉得我已经深深的爱上了JavaWeb。

发展方向：

现在也已经是大三上学期的期末了，放假的时候在网上深入学习一下spring，springMVC，springBoot 等相关的后端框架，如果有时间学习一下服务器的开发，这段时间应该是本科阶段最后一次学习新的知识了。放假之后再来就已经是大三的下学期，那个时候应该要开始准备考研了。如果考上研究生，方向就很多了，但是不管从事那个方向，我觉得JavaWeb都是一个很好的呈现形式，可以把很多东西放进JavaWeb中，通过网页的形式进行数据的可视化，这非常不错。

自评成绩：

JavaWeb的课设，认认真真的做了8天，其实在第一天的下午就已经实现了老师题目中给出的要求，但是我没有停手，一方面是我还有一个星期的时间才答辩，想更加熟练的运用Vue框架，另外一方面我对这个页面的功能确实也不是很满意，之后的每一天都在前一天的基础上加上了一些内容，其中有两天也是专门用于找各种背景图片，可以说付出了很多心血，如果让我给自己一个自评成绩，我会给自己95分

致谢：

这门课作为我这个学期收获最大的一门课，当之无愧要感谢聂刚老师开设这门课，并且很细心的撰写教案以及对我们自学方向的推荐，我就是自学了聂老师推荐的Vue框架，才能比较游刃有余的完成这次的课设内容，并且奠定了我寒假自学的方向。非常感谢聂老师的付出！