武汉纺织大学

Web应用开发课程设计

**天气预报查询**

**学 院： 数学与计算机学院**

**班 级： 物联网11801**

**姓 名： 欧振威**

**学 号： 1805230232**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年12月10日**

目 录

[1 需求分析 1](#_Toc22456)

[1.1运行程序 1](#_Toc6247)

[1.2天气预报查询功能 1](#_Toc15319)

[2 系统设计 1](#_Toc21868)

[3 系统实现](#_Toc20261) 2

[3.1 项目结构 2](#_Toc17930)

[3.2 配置文件 2](#_Toc14003)

[3.2.1 WeatherServlet.java 2](#_Toc32289)

[3.2.2 WeatherService.java 3](#_Toc20299)

[3.2.3 HttpUtil.java 4](#_Toc19038)

[3.3 css文件 6](#_Toc7838)

[3.4 img文件 6](#_Toc24038)

[3.5 index.jsp文件 6](#_Toc1705)

[4 系统测试 7](#_Toc7918)

[5 系统总结 7](#_Toc18618)

# 1 需求分析

设计一个天气预报查询系统，前端页面输入城市中文名称，服务器端调用天气预报接口查询当天天气和未来一周天气，具体要求如下：

## 1.1运行程序

当程序运行时，打开天气预报查询的前端页面，系统运行程序后，因为聚合数据天气API接口查询限制次数，为了节约页面打开初始的是假数据，输入城市点击查询就是真实的数据了。

## 1.2天气预报查询功能

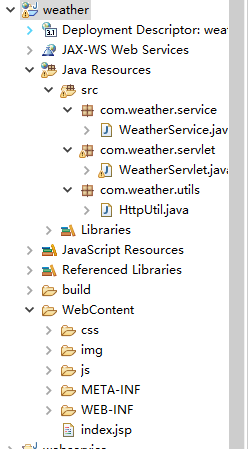
当在天气预报查询系统左上方输入用户所需查询的天气后，点击查询，页面就会显示出你所要查询的城市的数据，页面左边给出查询当天的日期、天气、温度、湿度、风向、紫外线，还给出了当天的穿衣指数和旅游指数。右边则是该城市一周的天气预报。

# **2 系统设计**

需要调用天气预报接口，调用聚合数据天气接口返回的是json格式的字符串，所以我们需要导入一个join.jar包，由于曲线图插件只支持在一个节点显示数据，所以曲线图的温度只会显示一个最高温度。

# **3 系统实现**

## 3.1 项目结构



## 3.2 配置文件

在src下新建三个包，取名分别为com.weather.service、com.weather.servlet和com.weather.utils

## 3.2.1 WeatherServlet.java

这个类是一个servlet，用于接收前端页面查询城市天气的请求以及响应

// 通过注解实现servlet的映射

@WebServlet("/getWeather")

public class WeatherServlet extends HttpServlet {

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

// 获取城市

String cityName = req.getParameter("cityName");

WeatherService service = new WeatherService();

resp.setCharacterEncoding("utf-8");

resp.getWriter().write(service.getWeather(cityName));

}

}

这个类代码相对较少，主要说明的有两点，第一个采用了@WebServlet注解进行servlet的映射，相比传统在web.xml文件中配置简便很多。第二个就是设置了响应数据的编码为utf-8，避免页面中文出现乱码。

## 3.2.2 WeatherService.java

这个类主要功能是在servlet接受请求后，通过调用该类的getWeather方法进行查询天气，并对返回的结果进行解析，组装自己需要的数据。

public class WeatherService {

private static final String KEY = "a9cd96368bb174cb77b55a611c12b4f6";

private static final String BASE\_URL = "http://v.juhe.cn/weather/index?format=2";

public String getWeather(String cityName) {

String url = BASE\_URL + "&cityname=" + cityName +"&key=" + KEY;

String resp = HttpUtil.doGet(url);

try {

// 通过map用来组装页面的数据

Map<String, Object> page = new HashMap<>();

// 解析获取的天气数据

JSONObject respObj = new JSONObject(resp);

JSONObject result = respObj.getJSONObject("result");

// 今天天气数据

JSONObject today = result.getJSONObject("today");

page.put("cityName", cityName);

.....

JSONObject weather\_id = today.getJSONObject("weather\_id");

String fa = weather\_id.getString("fa"); // 天气标识，用来识别天气种类

String imgUrl = "img/" + fa + ".png";// 拼接天气图标url

page.put("imgUrl", imgUrl);

// 当前天气实况数据

JSONObject sk = result.getJSONObject("sk");

page.put("temp", sk.getString("temp")); // 温度

.....

// 原始的未来一周天气

JSONArray future = result.getJSONArray("future");

// 创建链表，用来存储解析出来的一周天气数据

ArrayList<Map<String,Object>> futureList = new ArrayList<>();

for (int i = 0; i < future.length(); i++) {

// 每天的天气用一个map存储

Map<String, Object> itemMap = new HashMap<>();

// 循环获取原始的一周中的每天天气

JSONObject item = future.getJSONObject(i);

itemMap.put("weather", item.getString("weather"));

....

futureList.add(itemMap);

}

// 将自己组装好的一周天气添加到page中

page.put("futureList", futureList);

// 最后将map转换成json返回

return new JSONObject(page).toString();

} catch (JSONException e) {

e.printStackTrace();

}

return "";

}

}

这个类相当于最主要的，在getWeather方法中有一些关键地方。

**3.2.21**

调用的聚合天气接口返回的是json格式的字符串，所以为了方便处理需要先将数据转换成Json对象，这里用到了json.jar包，需要自己导入。

**3.2.22**

为了方便前端页面处理数据，在方法中将原有的天气数据进行解析，只选择页面需要的数据。为了方便数据的组装，采用Map将需要的数据通过键值对存入Map中。

**3.2.23**

在解析天气数据时还有个地方也是要注意的，因为页面要显示天气图标，所以需要获取天气的标识id，通过这个标识id去对应相应的天气图片，然后将图片的url拼接好，存入map中；还有就是未来一周天气数据，需要通过一个链表来保存，然后将链表存入map中。

## 3.2.3 HttpUtil.java

这个类是一个http请求的工具类，有需要http请求的地方调用方法就行，这里面关键就在读取返回数据，以及最后需要做关闭链接和流的操作。

public class HttpUtil {

public static String doGet(String httpurl) {

HttpURLConnection connection = null;

InputStream is = null;

BufferedReader br = null;

String result = null;// 返回结果字符串

try {

// 创建远程url连接对象

URL url = new URL(httpurl);

// 通过远程url连接对象打开一个连接，强转成httpURLConnection类

connection = (HttpURLConnection) url.openConnection();

// 设置连接方式：get

connection.setRequestMethod("GET");

// 发送请求

connection.connect();

// 通过connection连接，获取输入流

if (connection.getResponseCode() == 200) {

is = connection.getInputStream();

// 封装输入流is，并指定字符集

br = new BufferedReader(new InputStreamReader(is, "UTF-8"));

// 存放数据

StringBuffer sbf = new StringBuffer();

String temp = null;

while ((temp = br.readLine()) != null) {

sbf.append(temp);

sbf.append("\r\n");

}

result = sbf.toString();

}

} catch (MalformedURLException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

} finally {

// 关闭资源

if (null != br) {

try {

br.close();

} catch (IOException e) {

e.printStackTrace();

}

}

if (null != is) {

try {

is.close();

} catch (IOException e) {

e.printStackTrace();

}

}

connection.disconnect();// 关闭远程连接

}

return result;

}

}

## 3.3 css文件

在该文件下建立一个reset.css文件，用于优化前端页面的样式

/\*清除页面元素默认样式\*/

body,h1,h2,h3,h4,h5,h6,p,ol,ul{

margin: 0;

}

ol,ul,p,input{

padding: 0;

}

ol,ul{

/\*取消列表元素前面的原点\*/

list-style: none;

}

a{

/\*去掉a标签的下划线\*/

text-decoration: none;

}

img{

border: none;

/\*可以设置元素为块级或者内嵌元素\*/

display: block;

}

## 3.4 img文件

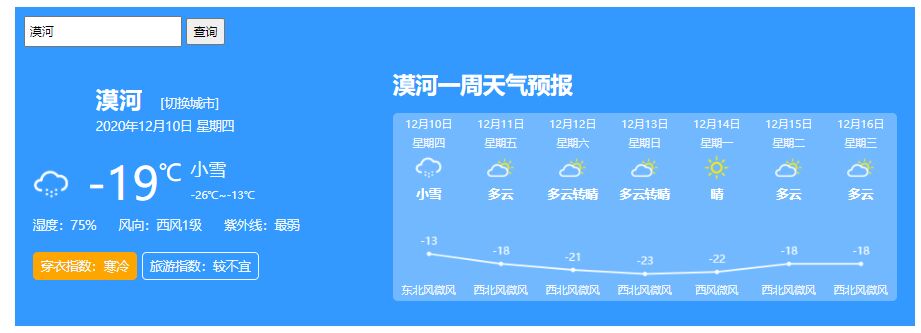
该文件下放置天气预报查询系统中所有用到的天气图标。

## 3.5 index.jsp

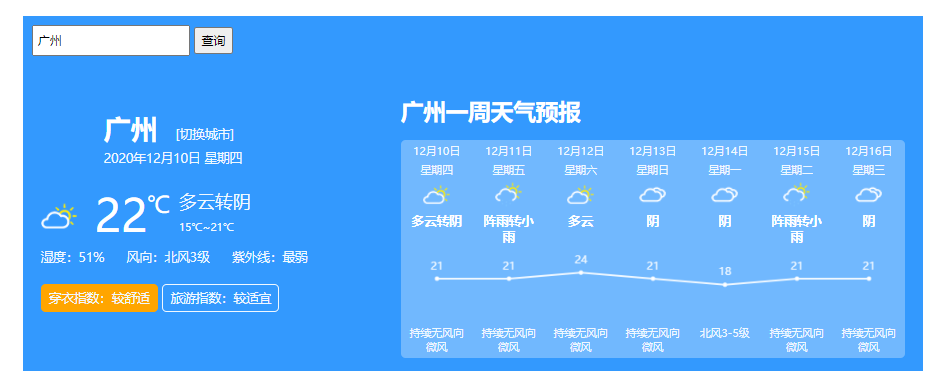
页面的布局，为了节省天气查询次数，所以初始打开页面是一个里面定义好的假的数据，当用户输入城市后才会显示真实数据。

# **4 系统测试**

在左上方输入漠河，点击查询，查询到的数据。



在左上方输入城市广州点击查询，查询到的数据。



# **5 系统总结**

开发过程中首先第一个问题是页面布局，页面整体相对来说细节很多，在网上查阅了很多css相关的内容，去解决布局的问题。再一个就是聚合接口返回的数据都是json格式，这个只能是转换成Json对象才能方便快捷处理，所以也查阅的java中处理json的相关内容，通过导入json相关的jar包来创建Json对象。一个学期的学习结束了，每一次的web应用开发课程都让我受益匪浅，通过这门课程的学习，我在大学毕业前，又掌握了一项新的前沿的开发技能，也有了更多的发展方向，这在以后找工作的过程中无疑为我增加了砝码，也可以成为我的一项兴趣爱好，可以根据自己的需要设计一些小程序。这门课更深的提升了我的编程技巧和编程方法，也更加深了对java的认识。