**Get Current Weather Web application Design Document**

**Author: J.Yang**

**06 03, 2016**

Overview

Web application to display current weather for 3 Australian cities: Sydney, Melbourne and Wollongong.

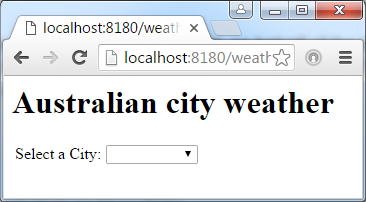
2.) Functional Requirements Description

Please refer to Senior Java Developer Test - Pactera - Get Current Weather V1 0.pdf.

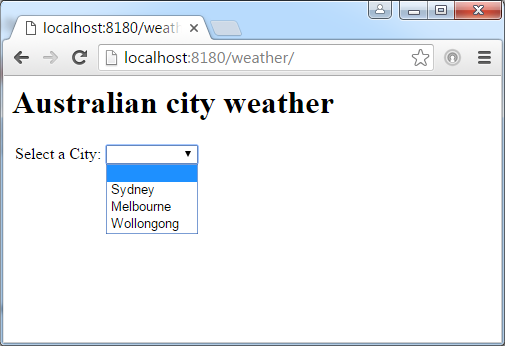
3.) Testing

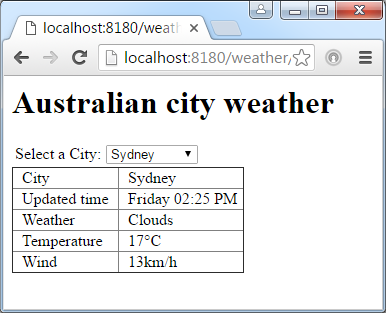
* Test case 1 – City selection page

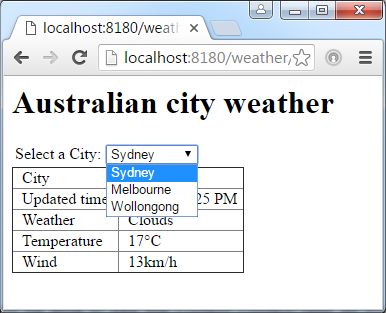
Type <http://localhost:8180/weather/> to Browser address bar.



* Test case 2 – Select a city







4.) Design

Write cityWeatherList.jsp view page to display the city selection and city weather. There're two sections on this page. The first section display drop down city selection. The second section display the weather information for a city in the table format.

When access the application via http://localhost:8080/weather/, we will only show the first section with a default to blank because there is no cityname. And the second section will not be generated. The Controller will use a cityWeather bean with blank cityName as command backing object so the view page knows not to generate the second section.

When click the drop down selection to select a city, javascript onchange function selectCity(me) will append the current selected cityName to the web root URL and send to the web app to get the city weather info like http://localhost:8080/weather/Sydney, we will display both sections. The first city selection will then become a drop down list of all city names without blank value. The Controller will use a new cityWeather bean with a valid cityName as command backing object so the view page knows to generate the second section.

4.1.  Useful Technical Implementation Detail

* weather\src\main\webapp\WEB-INF\pages\cityWeatherList.jsp  
  This is the JSP view page used to display the city selection and city weather. There're two sections on this page.  
  The first section uses
  + command.cityName binding to a <form:select>
  + javascript onchange function selectCity(me) to append the current selected cityName to the web root URL to generate a RequestMethod.GET and send to the web app to get the city weather info.

The second section uses CityWeather command backing object to display the weather information in a table.

When access the application via http://localhost:8080/weather/, we will only show the first section - city selection with a default to blank because there is no cityName. And the second section will not be generated.

When click the drop down selection to select a city, javascript onchange function selectCity(me) will append the current selected cityName to the web root URL and send to the web app to get the city weather info like http://localhost:8080/weather/Sydney, we will display both sections. The first city selection will be a drop down list of all city names without blank value.

* weather\src\main\webapp\WEB-INF\mvc-dispatcher-servlet.xml

Spring configuration xml file.

Define the String ArrayList cityListBean contains all city names. We can add more city to the list if required.

Define the default no cityName cityWeather bean and inject cityListBean and onlineWeatherAPI web address. Change the onlineWeatherAPI value if need to switch to another online weather API.

* weather\src\main\java\com\weather\controller\CityWeatherController.java

This is the main Spring MVC Controller for the Weather Web application.

When accessing the application without a cityName like http://localhost:8080/weather/, spring will pass the control to getDefaultCityWeather(). The getDefaultCityWeather() will use the default cityWeather bean as command backing object and pass to view "cityWeatherList". Because there is no cityName in the default cityWeather bean, the cityWeatherList.jsp will only show the city drop down selection.

When accessing the application with a cityName like http://localhost:8080/weather/Sydney, spring will pass the control to getCityWeather(). The getCityWeather() will create a new newCityWeather(cityName) bean as command backing object and pass to view "cityWeatherList". Because there is cityName in the command backing object, the cityWeatherList.jsp will show both city drop down selection and city weather info.

* weather\src\main\java\com\weather\model\CityWeather.java

Define CityWeather Bean.

We provide a default constructor because we want spring to create a default no cityName CityWeather Bean. We provide a factory to get the city weather info from the web to construct a CityWeather bean.

Define cityList and onlineWeatherAPI as static because the values will not change for different instances.

There is also a utility method readUrl(String url) can be used to read content from URL.

* weather\src\test\java\com\weather\unittest\TestMajorCityWeather.java

In this junit TestCase, we test the following,

* + Test the return json from online weather API contains Sydney.
  + Test construction of Sydney CityWeather Object.
  + Test construction of Melbourne CityWeather Object.
  + Load the onlineWeatherAPI from testConfig.properties.
* weather\src\test\java\com\weather\unittest\TestCityWeatherAnnotation.java

Use the junit annotation framework to test the web application HTML pages.

weather web application must be deployed and running before run this test, otherwise the test will fail.

We test the following,

* + Test default generated HTML page.
  + Test generated Sydney weather HTML page.

5.0 Deployment

5.1  Files

|  |  |
| --- | --- |
| **Directory Path and File Name** | **File Type** |
| weather\src\main\webapp\WEB-INF\pages\cityWeatherList.jsp | JSP view page. |
| weather\src\main\webapp\WEB-INF\mvc-dispatcher-servlet.xml | Spring config file |
| weather\src\main\webapp\WEB-INF\web.xml | Web app config file |
| weather\src\main\java\com\weather\controller\CityWeatherController.java | Spring Controller java source |
| weather\src\main\java\com\weather\model\CityWeather.java | Spring Bean java source |
| weather\src\test\java\com\weather\unittest\TestMajorCityWeather.java | Junit test java source |
| weather\src\test\java\com\weather\unittest\TestCityWeatherAnnotation.java | Junit test java source |