华中师范大学伍伦贡联合研究院
Central China Normal University Wollongong Joint Institute

UNIVERSITY OF WOLLONGONG AUSTRALIA

CSIT884 Web Development

Lecture 10 – AJAX

Lecture slides are mainly taken from UOW

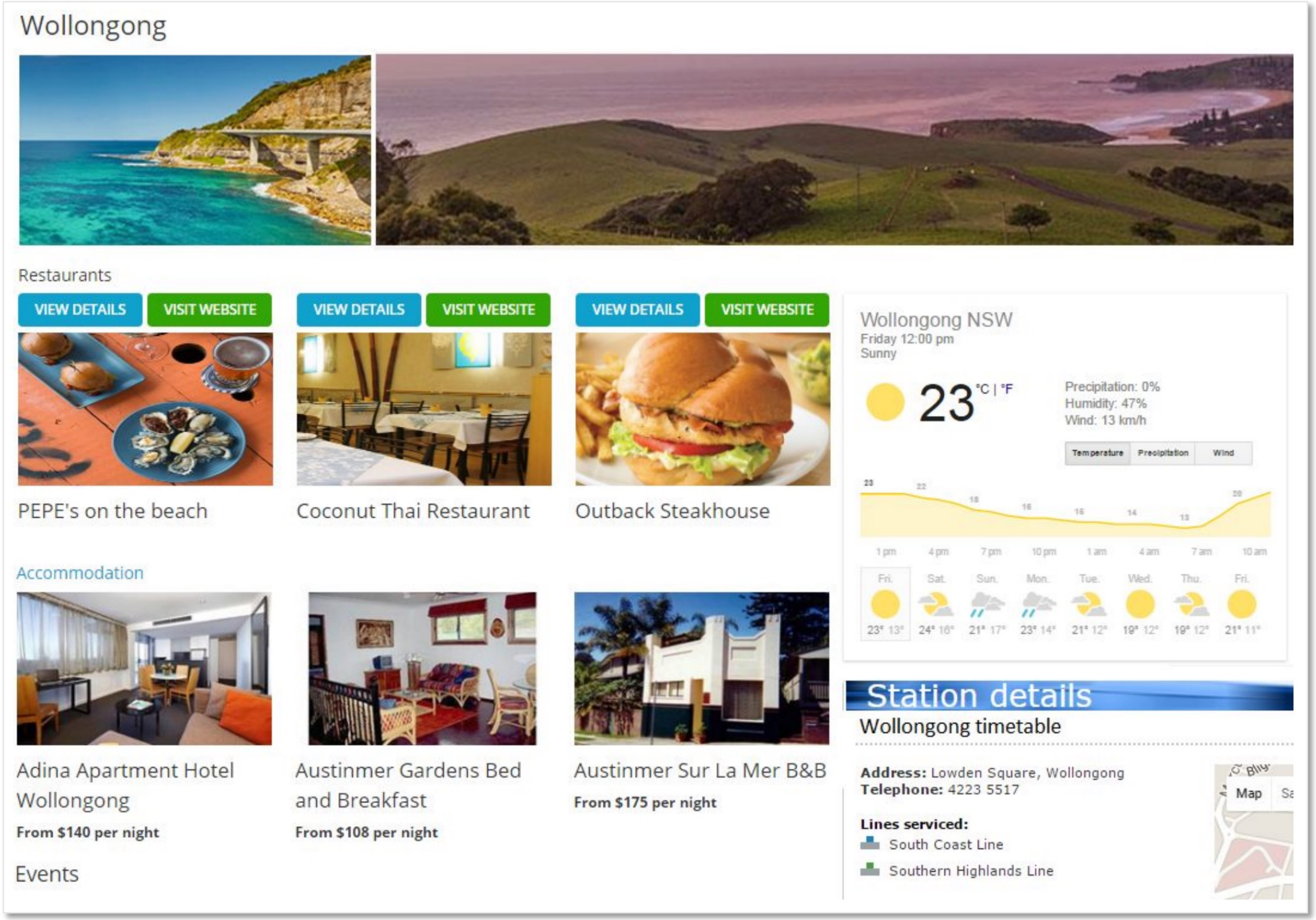
AJAX: Asynchronous JavaScript and XML

- Consider the following scenario:

Suppose we want to build a website about Wollongong. We want to display information about

  - Accommodation
  - Attractions
  - Events
  - Restaurants
  - Timetable
  - Weather

AJAX



Wollongong

Restaurants

PEPE's on the beach

Coconut Thai Restaurant

Outback Steakhouse

Accommodation

Adina Apartment Hotel Wollongong

From $140 per night

Austinmer Gardens Bed and Breakfast

From $108 per night

Austinmer Sur La Mer B&B

From $175 per night

Events

Wollongong NSW
Friday 12:00 pm
Sunny

23 °C | °F

Precipitation: 0%
Humidity: 47%
Wind: 13 km/h

Fri.	Sat.	Sun.	Mon.	Tue.	Wed.	Thu.	Fri.
23° 13°	24° 16°	21° 17°	23° 14°	21° 12°	19° 12°	19° 12°	21° 11°

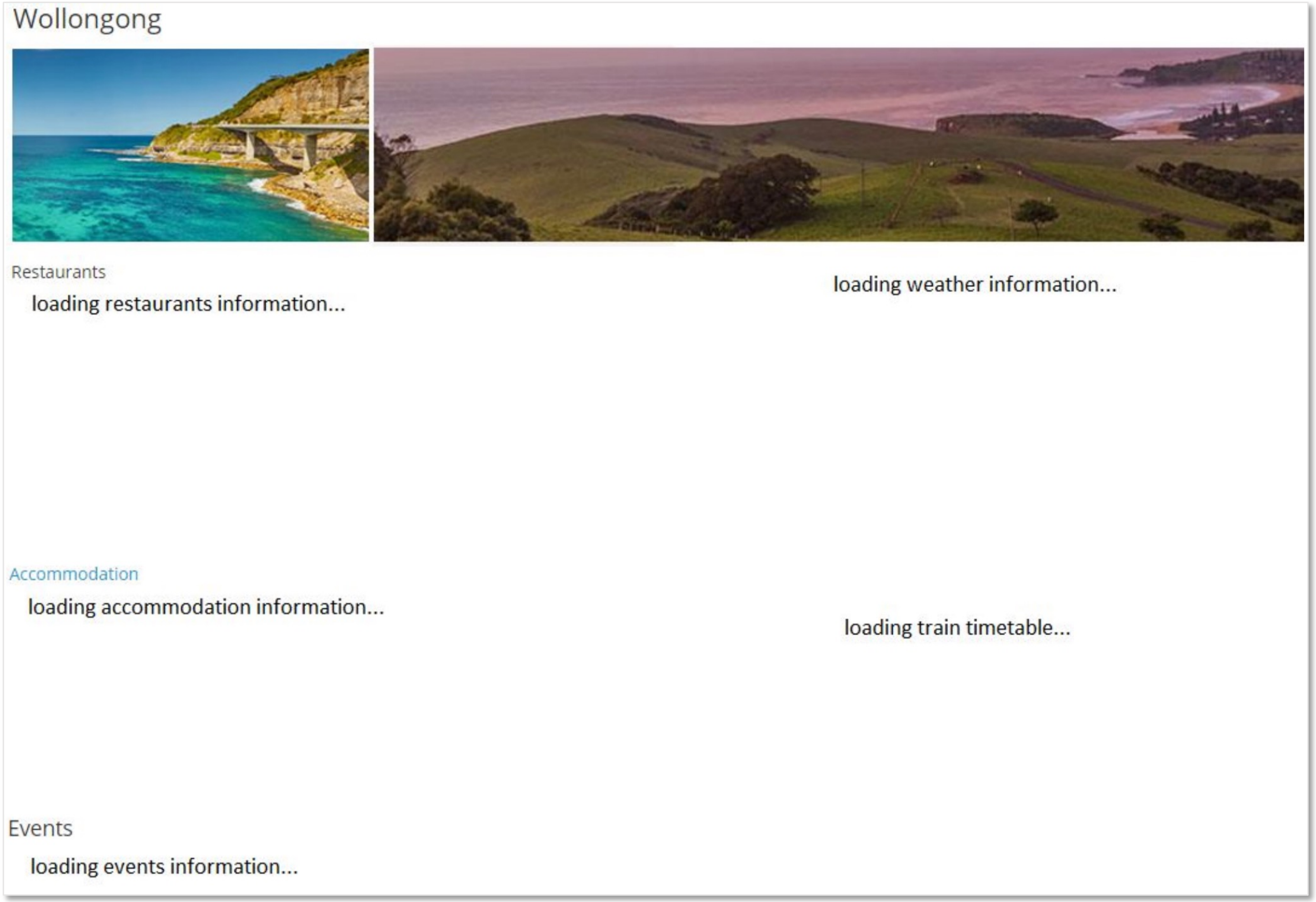
Station details

Wollongong timetable

Address: Lowden Square, Wollongong
Telephone: 4223 5517

Lines serviced:
South Coast Line
Southern Highlands Line

AJAX



Wollongong

Restaurants

loading restaurants information...

loading weather information...

Accommodation

loading accommodation information...

loading train timetable...

Events

loading events information...

AJAX

if we use synchronous calls to load information

- loading info 1…
- loading info 2…
- loading info 3…
- …

then the webpage will freeze and is not responsive during the loading.

What happens if one of these calls fails?

AJAX

AJAX allows us to send all the requests simultaneously and register **callback functions**

- sending request 1… if success then do this callback1
- sending request 2… if success then do this callback2
- sending request 3… if success then do this callback3
- …
- request 2 success -> evoke callback2 function
- request 3 success -> evoke callback3 function
- request 1 success -> evoke callback1 function
- …

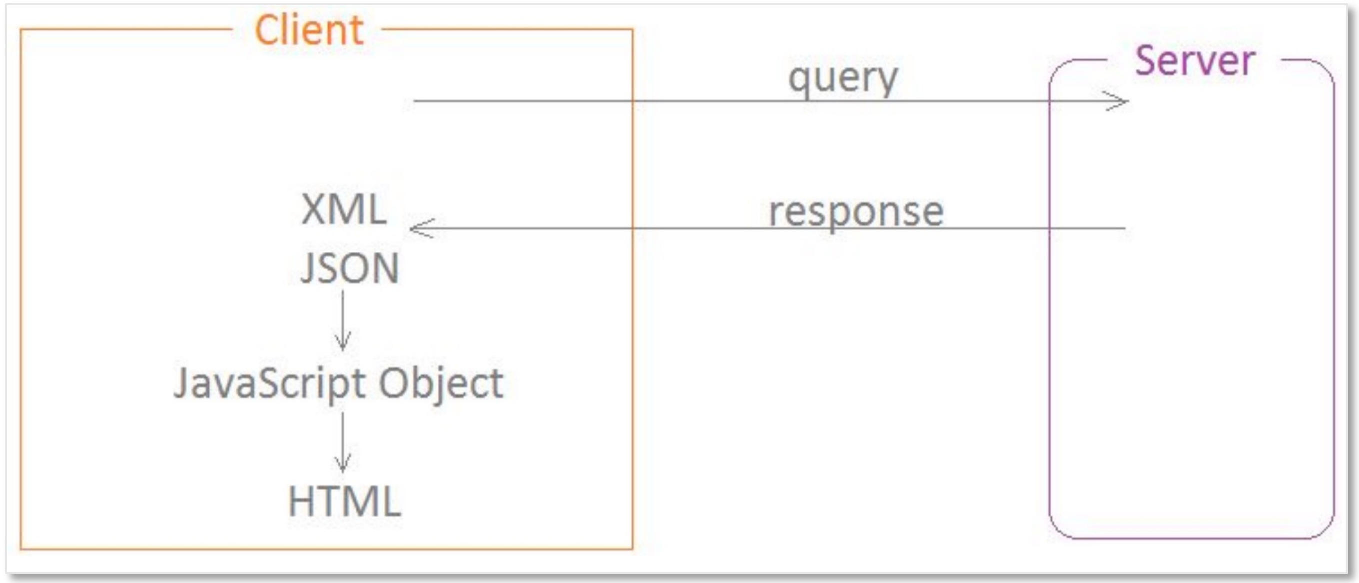
AJAX

With Ajax we can

- update a web page without reloading the page
- request data from a server - after the page has loaded
- receive data from a server - after the page has loaded
- send data to a server - in the background

Despite the name, the use of XML is not required, we can also use JSON as an alternative.

AJAX



Writing AJAX/JSON application:

- **Step 1**: Make the query
- **Step 2**: Get the response JSON
- **Step 3**: Parse the JSON response into a JavaScript object
- **Step 4**: Display the JavaScript object in an HTML page

A sample AJAX/JSON program

A sample AJAX/JSON program

This is the main function:

Step 1: Make the query

```
function makeAjaxQuery(){
 // create an XMLHttpRequest
 var xhttp = new XMLHttpRequest();
 // create a handler for the readyState change
 xhttp.onreadystatechange = function() {
   readyStateChangeHandler(xhttp);
 };
 // making query by async call
 xhttp.open("GET", "url-to-query-the-server", true);
 xhttp.send();
}
```

```
// handler for the readyState change
function readyStateChangeHandler(xhttp){ … }
```

A sample AJAX/JSON program

This is the callback function:

```
// handler for the readyState change
function readyStateChangeHandler(xhttp){
  if (xhttp.readyState == 4){
   // readyState = 4 means DONE
   if(xhttp.status == 200){
     // status = 200 means OK
     handleStatusSuccess(xhttp);
   }else{
     // status is NOT OK
     handleStatusFailure(xhttp);
   }
  }
}
```

```
// XMLHttpRequest failed
function handleStatusFailure(xhttp){ … }
// XMLHttpRequest success
function handleStatusSuccess(xhttp){ … }
```

A sample AJAX/JSON program

```
// XMLHttpRequest success
function handleStatusSuccess(xhttp){
 var jsonText = xhttp.responseText;
 // parse the json into an object
 var obj = JSON.parse(jsonText);
 // display the object on the page
 display(obj);
}
```

- **Step 2**: Get the response JSON
- **Step 3**: Parse the JSON response into a JavaScript object
- **Step 4**: Display the JavaScript object in an HTML page

A sample AJAX/JSON program

```
// parse the json into an object

 var obj = JSON.parse(jsonText);
```

- **Step 3**: Parse the JSON response into a JavaScript object

Note that this step is done by an easy function call `JSON.parse()`

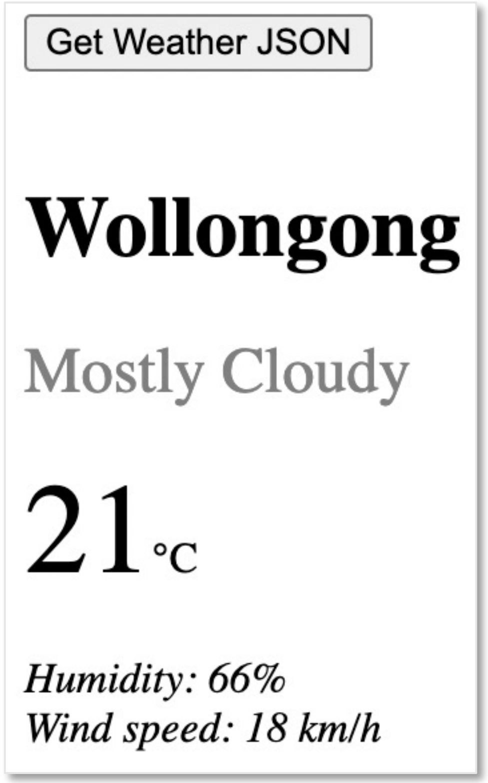
```
// display the object on the page

function display(obj){

 // construct HTML code to display the
object

 ...

}
```

- **Step 4**: Display the JavaScript object in an HTML page

The main job the AJAX/JSON program is to write the function: **`display`**

AJAX/JSON Example: Weather Forecast

This example emulates an application where a server allows the user to retrieve current weather forecast for a queried location.



Wollongong

Mostly Cloudy

21°C

Humidity: 66%
Wind speed: 18 km/h

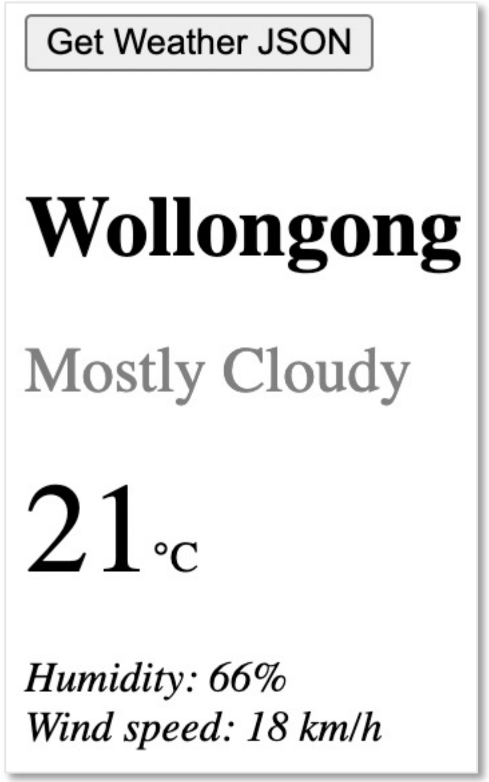
AJAX/JSON Example: Weather Forecast

The purpose of this example is

- to show how to distinguish between a failed request and a successful request
- when the request is failed, display an error message
- when the request is successfully then display the weather information:
    1. parse the JSON response to a JavaScript weather object;
    2. display the weather object on the web page.

AJAX/JSON Example: Weather Forecast

```
<button onClick="makeAjaxQueryWeather()">

Get Weather JSON

</button>

<br /><br />

<div id="display">

</div>
```



Wollongong

Mostly Cloudy

21 °C

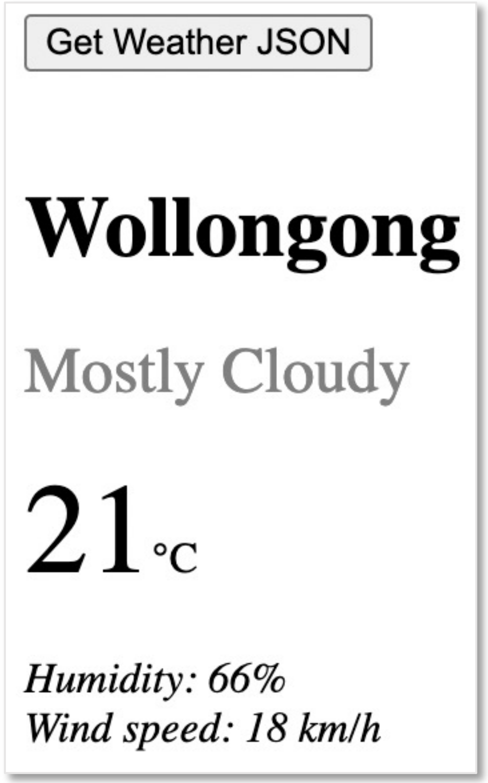
Humidity: 66%
Wind speed: 18 km/h

```
function makeAjaxQueryWeather(){

 // create an XMLHttpRequest

 var xhttp = new XMLHttpRequest();

 // create a handler for the readyState change

 xhttp.onreadystatechange = function() {

   readyStateChangeHandler(xhttp);

 };

 // get JSON file by making async call

 xhttp.open("GET", "weather.json", true);

 xhttp.send();

}
```

AJAX/JSON Example: Weather Forecast

```
// handler for the readyState change
function readyStateChangeHandler(xhttp){
  if (xhttp.readyState == 4){
    // readyState = 4 means DONE
    if(xhttp.status == 200){
      // status = 200 means OK
      handleStatusSuccess(xhttp);
    }else{
      // status is NOT OK
      handleStatusFailure(xhttp);
    }
  }
}
```

```
// XMLHttpRequest failed
function handleStatusFailure(xhttp){ … }
// XMLHttpRequest success
function handleStatusSuccess(xhttp){ … }
```



Wollongong

Mostly Cloudy

21 °C

Humidity: 66%
Wind speed: 18 km/h

AJAX/JSON Example: Weather Forecast

When the request is failed, display an error message

```
// XMLHttpRequest failed
function handleStatusFailure(xhttp){
 // display error message
 var displayDiv = document.getElementById("display");
 displayDiv.innerHTML = "XMLHttpRequest failed: status "
+ xhttp.status;
}
```

AJAX/JSON Example: Weather Forecast

When the request is successful

```
// XMLHttpRequest success
function handleStatusSuccess(xhttp){
 var jsonText = xhttp.responseText;
 // parse the json into an object
 var weatherObj = JSON.parse(jsonText);
 // display the object on the page
 displayWeather(weatherObj);
}
```

Get the response JSON

Parse the JSON response into a JavaScript object

Display the object in a HTML page

AJAX/JSON Example: Weather Forecast

```
// parse the json into an object
 var weatherObj = JSON.parse(jsonText);
```

What does the `weatherObj` look like?

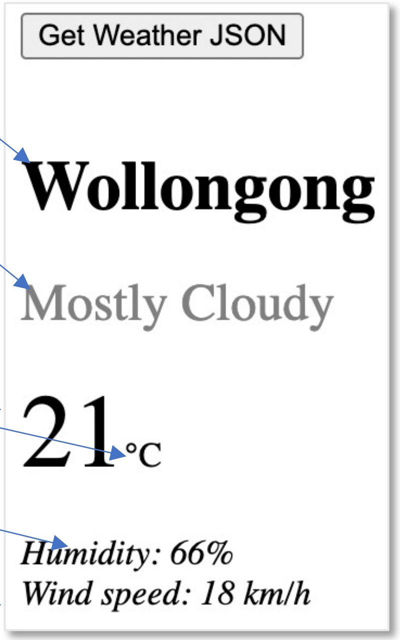
```
{
  "queryLocation": "Wollongong",
  "forecast": "Mostly Cloudy",
  "temperature": {
    "degree": "21",
    "scale": "C"
  },
  "humidity": "66%",
  "windSpeed": "18 km/h"
}
```

```
weatherObj {
  queryLocation: "Wollongong",
  forecast: "Mostly Cloudy",
  temperature: {
    degree: "21",
    scale: "C"
  },
  humidity: "66%",
  windSpeed: "18 km/h"
}
```

AJAX/JSON Example: Weather Forecast

```
// display the weather object on the page
function displayWeather(weatherObj){
 ...
}
```

```
weatherObj {
  queryLocation: "Wollongong",
  forecast: "Mostly Cloudy",
  temperature: {
    degree: "21",
    scale: "C"
  },
  humidity: "66%",
  windSpeed: "18 km/h"
}
```



Wollongong

Mostly Cloudy

21°C

Humidity: 66%
Wind speed: 18 km/h

We need to construct the following **HTML code** to display the weather information

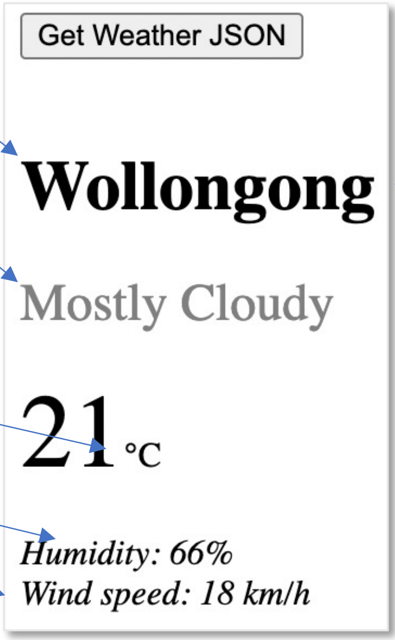
```
<h1>Wollongong</h1>
<font size='5' color='gray'>Mostly Cloudy</font>
<br /><br />
<font size='7'>21</font>
&deg; C
<br /><br />
<i>Humidity: 66%</i>
<br />
```

Q: How do we get the query location?

AJAX/JSON Example: Weather Forecast

```
// display the weather object on the page
function displayWeather(weatherObj){
 ...
}
```

```
weatherObj {
  queryLocation: "Wollongong",
  forecast: "Mostly Cloudy",
  temperature: {
    degree: "21",
    scale: "C"
  },
  humidity: "66%",
  windSpeed: "18 km/h"
}
```



Wollongong

Mostly Cloudy

21 °C

Humidity: 66%
Wind speed: 18 km/h

We need to construct the following HTML code to display the weather information

```
<h1>Wollongong</h1>
<font size='5' color='gray'>Mostly Cloudy</font>
<br /><br />
<font size='7'>21</font>
&deg; C
<br /><br />
<i>Humidity: 66%</i>
<br />
```

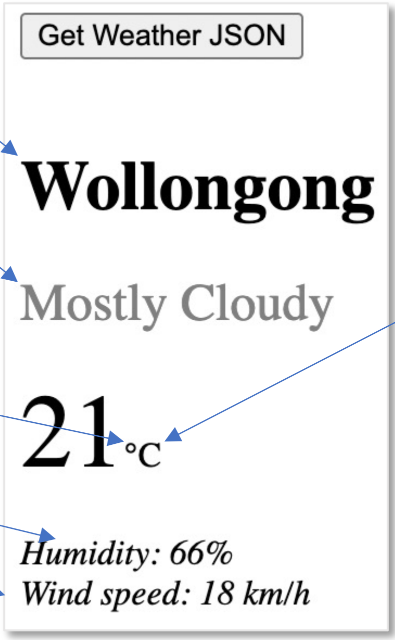
Q: How do we get the query location?

A: `weatherObj.queryLocation`

AJAX/JSON Example: Weather Forecast

```
// display the weather object on the page
function displayWeather(weatherObj){
 ...
}
```

```
weatherObj {
  queryLocation: "Wollongong",
  forecast: "Mostly Cloudy",
  temperature: {
    degree: "21",
    scale: "C"
  },
  humidity: "66%",
  windSpeed: "18 km/h"
}
```



We need to construct the following HTML code to display the weather information

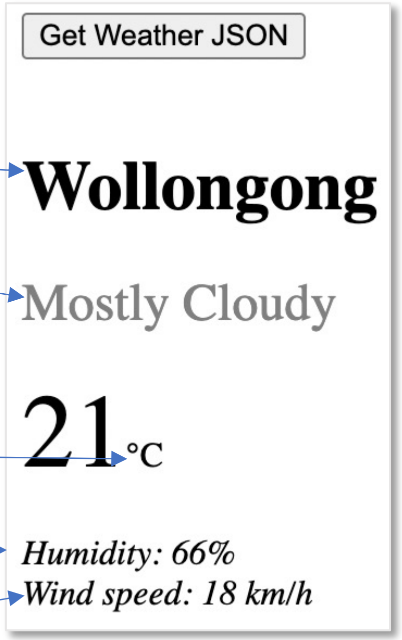
```
<h1>Wollongong</h1>
<font size='5' color='gray'>Mostly Cloudy</font>
<br /><br />
<font size='7'>21</font>
&deg; C
<br /><br />
<i>Humidity: 66%</i>
<br />
```

Q: How do we get the temperature scale?

A: `weatherObj.temperature.scale`

AJAX/JSON Example: Weather Forecast

```
// display the weather object on the page
function displayWeather(weatherObj){
 // construct HTML code to display weather information
 var html = "<h1>" + weatherObj.queryLocation + "</h1>";
 html = html + "<font size='5' color='gray'>" + weatherObj.forecast + "</font>";
 html = html + "<br /><br />";

 html = html + "<font size='7'>" + weatherObj.temperature.degree + "</font>";
 html = html + "&deg;" + weatherObj.temperature.scale;
 html = html + "<br /><br />";
 html = html + "<i>Humidity: " + weatherObj.humidity + "</i>";
 html = html + "<br />";
 html = html + "<i>Wind speed: " + weatherObj.windSpeed + "</i>";
 // show the constructed HTML code in the display div
 var displayDiv = document.getElementById("display");
 displayDiv.innerHTML = html;
}
```



Wollongong

Mostly Cloudy

21°C

Humidity: 66%
Wind speed: 18 km/h

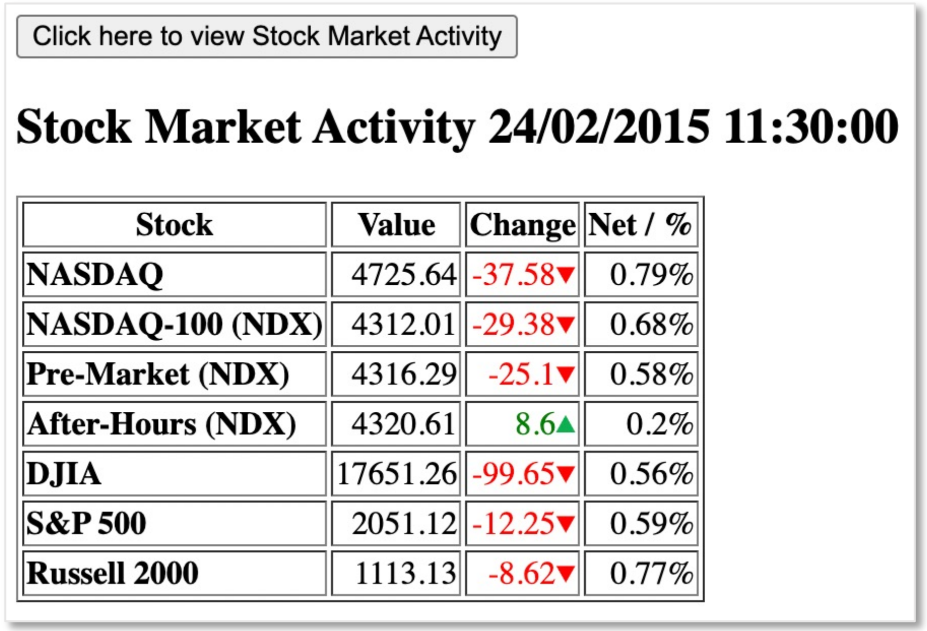
AJAX/JSON Example: Stock Market

This example emulates an application where a server allows the user to retrieve stock market information.

AJAX/JSON Example: Stock Market

Assume that there is a JSON file, called `market.json`. Write HTML and JavaScript codes that do the following:

There is a button “Click here to view Stock Market Activity”. When the user clicks on this button, make an Ajax call to get the stock information from the JSON file and display them in a table



Stock Market Activity 24/02/2015 11:30:00

Stock	Value	Change	Net / %
NASDAQ	4725.64	-37.58▼	0.79%
NASDAQ-100 (NDX)	4312.01	-29.38▼	0.68%
Pre-Market (NDX)	4316.29	-25.1▼	0.58%
After-Hours (NDX)	4320.61	8.6▲	0.2%
DJIA	17651.26	-99.65▼	0.56%
S&P 500	2051.12	-12.25▼	0.59%
Russell 2000	1113.13	-8.62▼	0.77%

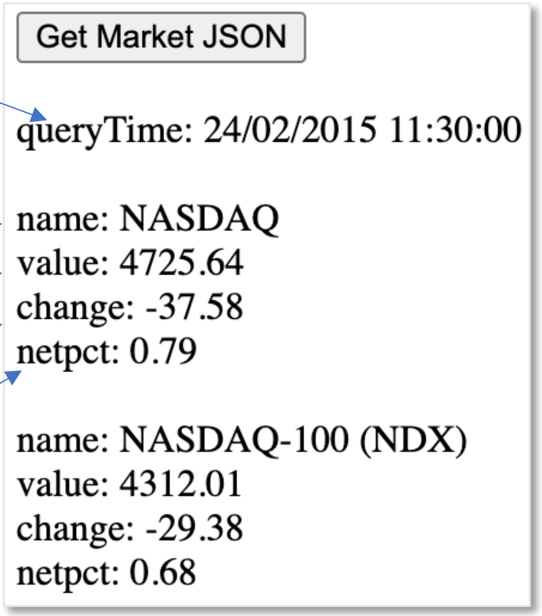
AJAX/JSON Example: Stock Market

This is the content of the JSON file `market.json`.

```
{
  "queryTime": "24/02/2015 11:30:00",
  "stockList": [
    {
    "name": "NASDAQ",
    "value": 4725.64,
    "change": -37.58,
    "netpct": 0.79
    },
      {
       "name": "NASDAQ-100 (NDX)",
       "value": 4312.01,
       "change": -29.38,
       "netpct": 0.68
       },
       ....
       {
       "name": "Russell 2000",
       "value": 1113.13,
       "change": -8.62,
       "netpct": 0.77
       }
    ]
}
```

Version 0 - plain display

```
// display the market object on the page
function displayMarket(marketObj){
  // construct HTML code to display market information
  var html = "";
  html += "queryTime: " + marketObj.queryTime;
  html += "<br /><br />";
  for(var i=0; i < marketObj.stockList.length; i++){
   var stockObj = marketObj.stockList[i];
   html += "name: " + stockObj.name;
   html += "<br />";
   html += "value: " + stockObj.value;
   html += "<br />";
   html += "change: " + stockObj.change;
   html += "<br />";
   html += "netpct: " + stockObj.netpct;
   html += "<br /><br />";
 }
 // show the constructed HTML code in the display div
 var displayDiv = document.getElementById("display");
 displayDiv.innerHTML = html;
}
```



```
queryTime: 24/02/2015 11:30:00

name: NASDAQ
value: 4725.64
change: -37.58
netpct: 0.79

name: NASDAQ-100 (NDX)
value: 4312.01
change: -29.38
netpct: 0.68
```

```
marketObj {
 queryTime: "24/02/2015 11:30:00", stockList: [
   {
   name: "NASDAQ",
   value: 4725.64,
   change: -37.58,
   netpct: 0.79
   },
   {
   name: "NASDAQ-100 (NDX)",
   value: 4312.01,
   change: -29.38,
   netpct: 0.68
   },
   ....
   {
   name: "Russell 2000",
   value: 1113.13,
   change: -8.62,
   netpct: 0.77
   }
 ]
}
```

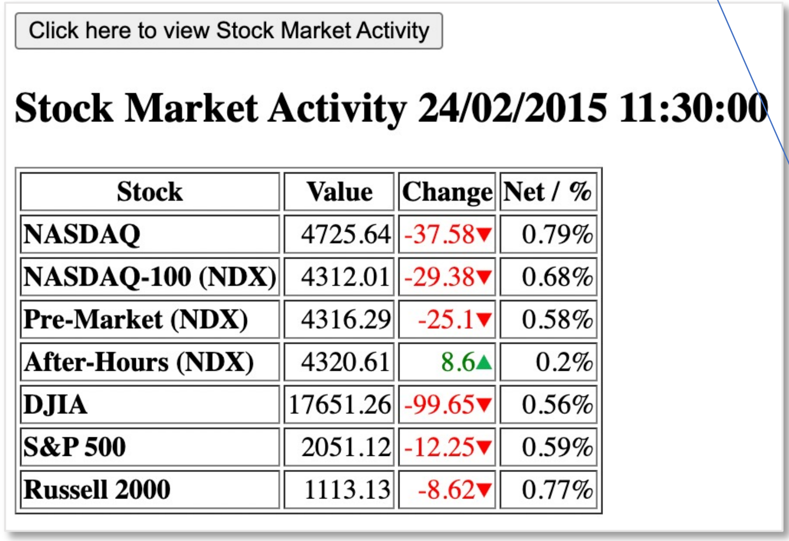
Version 1 - table display

```
// display the market object on the page
function displayMarket(marketObj){
 …
}
```

We need to construct the following HTML code to display the stock market information

```
<h2>Stock Market Activity 24/02/2015 11:30:00</h2>
<table border='1'>
 <tr> <th>Stock</th> <th>Value</th> <th>Change</th>
 <th>Net / %</th> </tr>
 <tr>
   <td><b> NASDAQ</b></td>
   <td align='right'> 4725.64</td>
   <td style='color:red' align='right'>
   -37.58
   <img src='stockDown.png' />
   </td>
   <td align='right'> 0.79%</td>
 </tr>
 <tr>
   <td><b> After Hours (NDX)</b></td>
   <td align='right'> 4320.61</td>
   <td style='color:green' align='right'>
   8.6
   <img src='stockUp.png' />
   </td>
   <td align='right'> 0.2%</td>
 </tr>
</table>
```

```
marketObj {
 queryTime: "24/02/2015 11:30:00", stockList: [
  {
   name: "NASDAQ",
   value: 4725.64,
   change: -37.58,
   netpct: 0.79
  },
  {
   name: "NASDAQ-100 (NDX)",
   value: 4312.01,
   change: -29.38,
   netpct: 0.68
  },
  ....
 ]
}
```

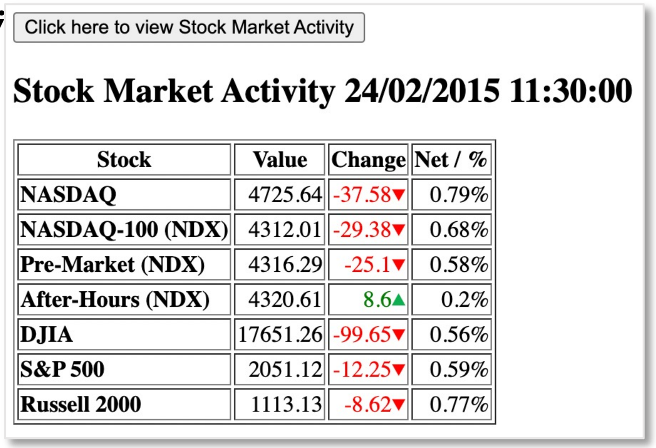


Stock Market Activity 24/02/2015 11:30:00

Stock	Value	Change	Net / %
NASDAQ	4725.64	-37.58▼	0.79%
NASDAQ-100 (NDX)	4312.01	-29.38▼	0.68%
Pre-Market (NDX)	4316.29	-25.1▼	0.58%
After-Hours (NDX)	4320.61	8.6▲	0.2%
DJIA	17651.26	-99.65▼	0.56%
S&P 500	2051.12	-12.25▼	0.59%
Russell 2000	1113.13	-8.62▼	0.77%

Version 1 - table display

```
// display the market object on the page
function displayMarket(marketObj){
 // construct HTML code to display market information
 var html = "<h2>Stock Market Activity " + marketObj.queryTime + "</h2>";
 html += "<table border='1'>";
 html += "<tr><th>Stock</th><th>Value</th><th>Change</th><th>Net / %</th></tr>";
 for(var i=0; i < marketObj.stockList.length; i++){
   var stockObj = marketObj.stockList[i];
   html += "<tr>";
   html += "<td><b>" + stockObj.name + "</b></td>";
   html += "<td align='right'>" + stockObj.value + "</td>";
   if(stockObj.change < 0){
     html += "<td style='color:red' align='right'>";
     html += stockObj.change;
     html += "<img src='stockDown.png' />";
     html += "</td>";
   }else{
     html += "<td style='color:green' align='right'>";
     html += stockObj.change;
     html += "<img src='stockUp.png' />";
     html += "</td>";
   }
   html += "<td align='right'>" + stockObj.netpct + "%</td>";
   html += "</tr>";
 }
 html += "</table>";
 // show the constructed HTML code in the display div
 var displayDiv = document.getElementById("display");
 displayDiv.innerHTML = html;
}
```



Click here to view Stock Market Activity

Stock Market Activity 24/02/2015 11:30:00

Stock	Value	Change	Net / %
NASDAQ	4725.64	-37.58▼	0.79%
NASDAQ-100 (NDX)	4312.01	-29.38▼	0.68%
Pre-Market (NDX)	4316.29	-25.1▼	0.58%
After-Hours (NDX)	4320.61	8.6▲	0.2%
DJIA	17651.26	-99.65▼	0.56%
S&P 500	2051.12	-12.25▼	0.59%
Russell 2000	1113.13	-8.62▼	0.77%

```
marketObj {
 queryTime: "24/02/2015 11:30:00", stockList: [
   {
   name: "NASDAQ",
   value: 4725.64,
   change: -37.58,
   netpct: 0.79
   },
   {
   name: "NASDAQ-100 (NDX)",
   value: 4312.01,
   change: -29.38,
   netpct: 0.68
   },
   ....
   {
   name: "Russell 2000",
   value: 1113.13,
   change: -8.62,
   netpct: 0.77
   }
 ]
}
```

References

- http://www.w3schools.com/json
- Robert W. Sebesta, Programming the World Wide Web, Pearson.