



# Title Lorem Ipsum

Sit Dolor Amet

# Title Lorem Ipsum

01

LOREM IPSUM  
DOLOR SIT AMET,  
CONSECTETUER  
ADIPISCING ELIT.  
MAECENAS

02

NUNC VIVERRA  
IMPERDIET ENIM.  
FUSCE EST. VIVAMUS  
A TELLUS.

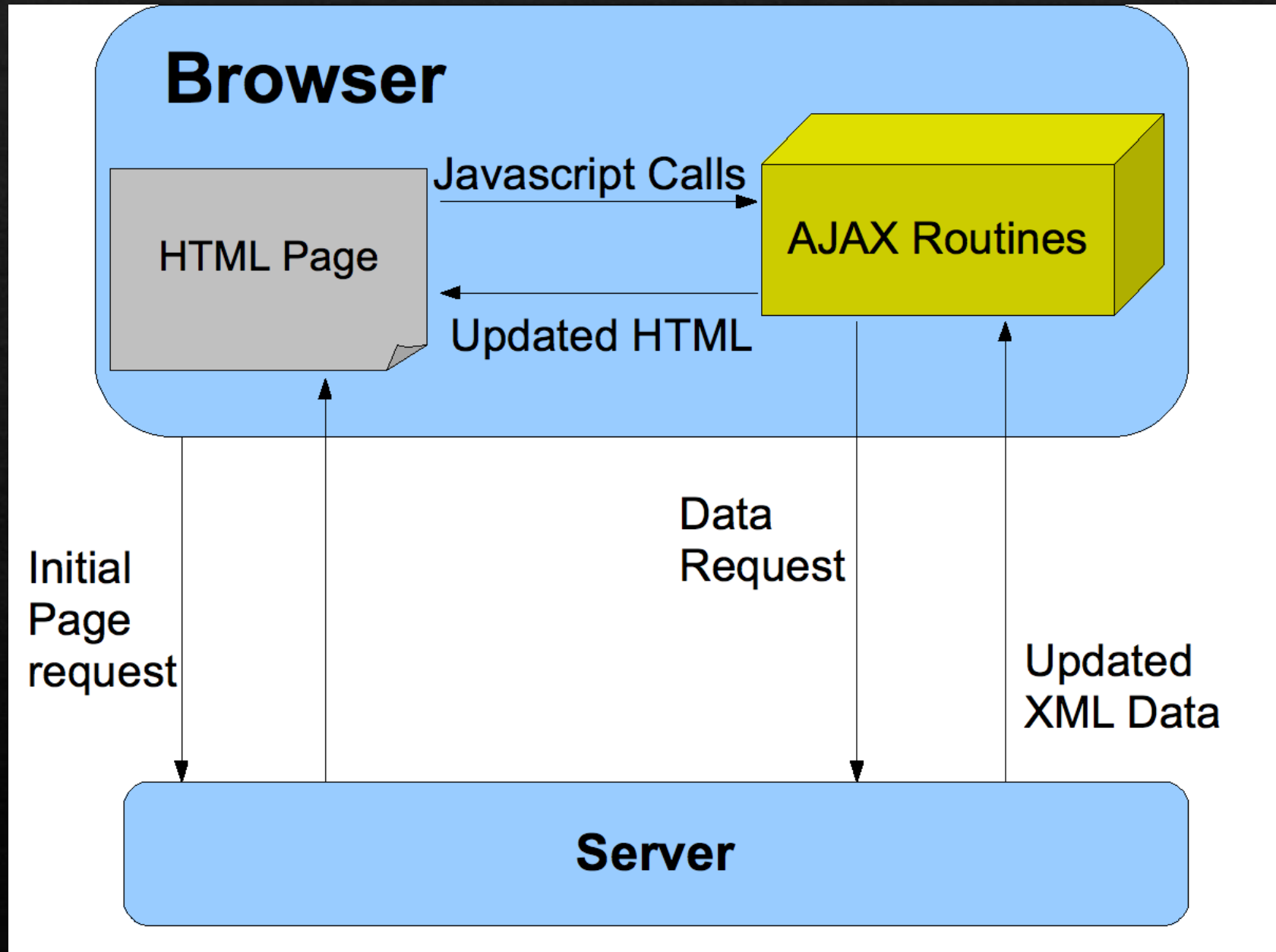
03

PELLENTEQUE  
HABITANT MORBI  
TRISTIQUE  
SENECTUS ET  
NETUS ET  
MALESUADA FAMES.

# AJAX –

## Chapter 8

### Sample Files





view-source:D:/!Courses/322\_257/!BOOK/Chapter08/c08/data-html.html

# data-html.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>JavaScript & jQuery - Chapter 8: Ajax & JSON - Loading HTML
6     <link rel="stylesheet" href="css/c08.css" />
7   </head>
8   <body>
9
10     <header><h1>THE MAKER BUS</h1></header>
11     <h2>The bus stops here.</h2>
12     <section id="content"></section>
13
14     <script src="js/data-html.js"></script>
15
16   </body>
17 </html>
```



← → ↻ ⓘ File | D:/!Courses/322\_257/!BOOK/Chapter08/c08/js/data-html.js

## data-html.js

```
// NOTE: If you run this file locally
// You will not get a server status and the example will fail
// Comment out lines 9 and 11 if you are working locally

var xhr = new XMLHttpRequest(); // Create XMLHttpRequest object

xhr.onload = function() { // When response has loaded
    // The following conditional check will not work locally - only on a server
    if(xhr.status === 200) { // If server status was ok
        document.getElementById('content').innerHTML = xhr.responseText; // Update
    }
};

xhr.open('GET', 'data/data.html', true); // Prepare the request
xhr.send(null); // Send the request
```





THE MAKER BUS

Local file (no server)

---

The bus stops here.



THE MAKER BUS

hosted file (web server)

The bus stops here.



San Francisco, CA  
May 1



Austin, TX  
May 15



New York, NY  
May 30

```
xhr.open('GET', 'data/data.html', true); // Prepare the request xhr.send(null);  
xhr.open('GET', 'data/data.html', true); // Prepare the request xhr.send(null);  
xhr.open('GET', 'data/data.html', true); // Prepare the request xhr.send(null);
```

# The Difference? Local File Access

**html file (data/data.html)**

Data-html.html could not access another folder and pass info back and forth using Finder or File Explorer.

AJAX is a client/server request. So data-html.html is loaded, and the AJAX request is called through the JavaScript file:

```
xhr.open('GET', 'data/data.html', true);
```

```
xhr.send(null);
```

```
← → ↻ ⓘ view-source:D:/!Courses/322_257/!BOOK/Chapter08/c08/data/data.html  
1 <div class="event">  
2     
3   <p><b>San Francisco, CA</b><br>  
4   May 1</p>  
5 </div>  
6 <div class="event">  
7     
8   <p><b>Austin, TX</b><br>  
9   May 15</p>  
10 </div>  
11 <div class="event">  
12     
13   <p><b>New York, NY</b><br>  
14   May 30</p>  
15 </div>
```



```
xhr.open('GET', 'data/data.html', true); // Prepare the request xhr.send(null);  
xhr.open('GET', 'data/data.html', true); // Prepare the request xhr.send(null);  
xhr.open('GET', 'data/data.html', true); // Prepare the request xhr.send(null);
```

# The Difference? Local File Access

**XML file (data/data.xml)**

Same file (data.html.html) but it  
points instead to an XML file

(Extensible Markup Language)

```
xhr.open('GET', 'data/data.xml', true);  
  
xhr.send(null);
```

← → ↻ 🌐 file:///D:/!Courses/322\_257/!BOOK/Chapter08/c08/data/data.xml

```
1 <?xml version="1.0" encoding="utf-8" ?>  
2 <events>  
3   <event>  
4     <location>San Francisco, CA</location>  
5     <date>May 1</date>  
6     <map>img/map-ca.png</map>  
7   </event>  
8   <event>  
9     <location>Austin, TX</location>  
10    <date>May 15</date>  
11    <map>img/map-tx.png</map>  
12  </event>  
13  <event>  
14    <location>New York, NY</location>  
15    <date>May 30</date>  
16    <map>img/map-ny.png</map>  
17  </event>  
18 </events>
```

# XML Tags Describe the Data they Contain

← → ↻ 🌐 file:///D:/!Courses/322\_257/!BOOK/Chapter08/c08/data/data.xml

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <events>
3   <event>
4     <location>San Francisco, CA</location>
5     <date>May 1</date>
6     <map>img/map-ca.png</map>
7   </event>
8   <event>
9     <location>Austin, TX</location>
10    <date>May 15</date>
11    <map>img/map-tx.png</map>
12  </event>
13  <event>
14    <location>New York, NY</location>
15    <date>May 30</date>
16    <map>img/map-ny.png</map>
17  </event>
18 </events>
```

```
xhr.open('GET', 'data/data.html', true); // Prepare the request xhr.send(null);  
xhr.open('GET', 'data/data.html', true); // Prepare the request xhr.send(null);  
xhr.open('GET', 'data/data.html', true); // Prepare the request xhr.send(null);
```

# The Difference? Local File Access

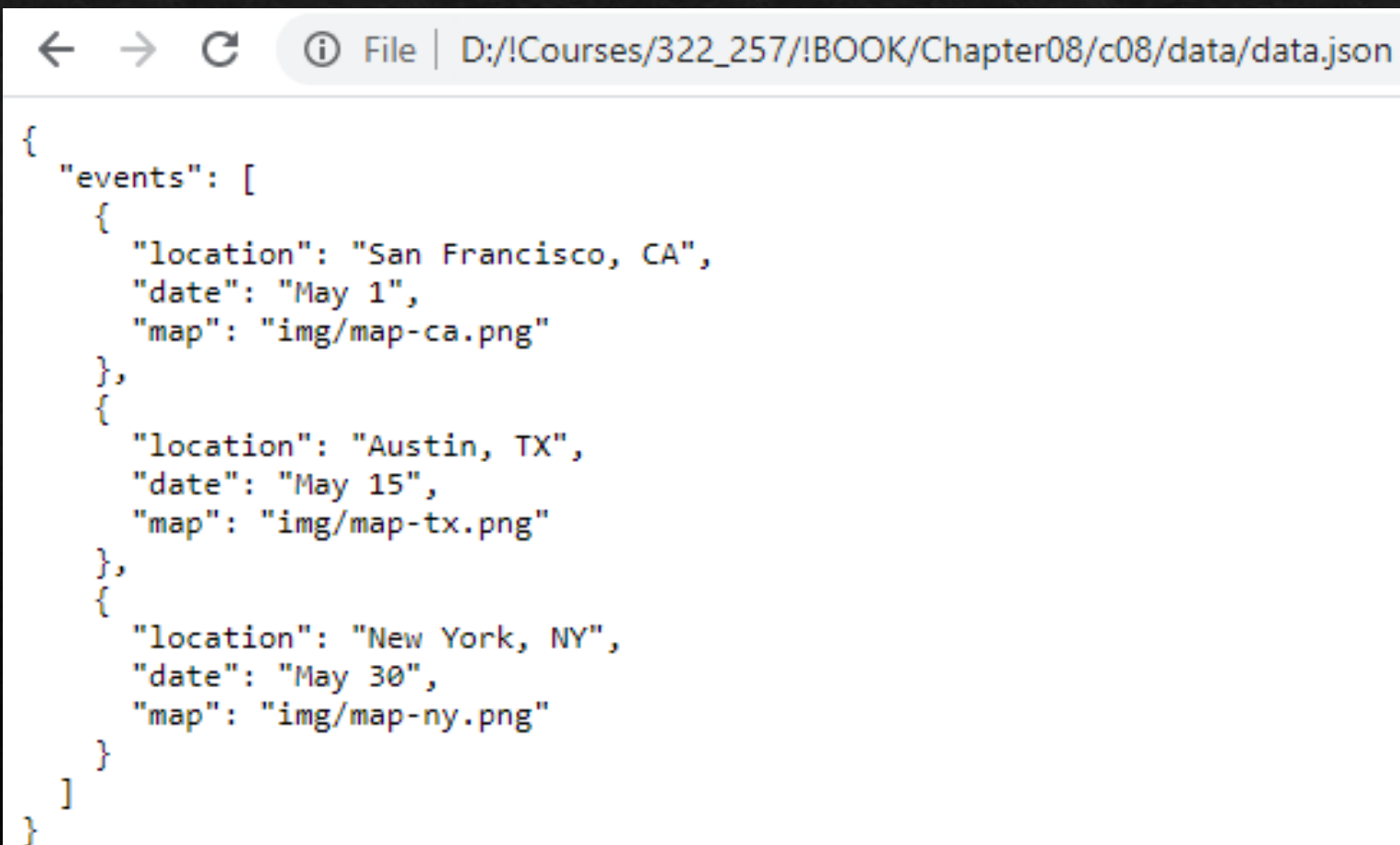
**XML file (data/data.xml)**

Same file (data.html.html) but it  
points instead to an JSON file

(JavaScript Object Notation)

```
xhr.open('GET', 'data/data.json', true);
```

```
xhr.send(null);
```

A screenshot of a web browser window. The address bar shows the file path: D:/!Courses/322\_257/!BOOK/Chapter08/c08/data/data.json. The browser displays the content of the JSON file, which is a JavaScript Object Notation (JSON) object. The object has a single key "events" which points to an array of three objects. Each object in the array represents an event with three properties: "location", "date", and "map". The first event is for San Francisco, CA, on May 1, with a map image at img/map-ca.png. The second event is for Austin, TX, on May 15, with a map image at img/map-tx.png. The third event is for New York, NY, on May 30, with a map image at img/map-ny.png.

```
{  
  "events": [  
    {  
      "location": "San Francisco, CA",  
      "date": "May 1",  
      "map": "img/map-ca.png"  
    },  
    {  
      "location": "Austin, TX",  
      "date": "May 15",  
      "map": "img/map-tx.png"  
    },  
    {  
      "location": "New York, NY",  
      "date": "May 30",  
      "map": "img/map-ny.png"  
    }  
  ]  
}
```

# Don't Be Deceived... It's Just Text (for now)

- ◆ Do pay attention to the structure
- ◆ Text organizes String into Data set, using Key:Value Pairs
- ◆ Key – on the left are repeated data elements that are matched with unique values, ie
- ◆ “location” : “San Francisco, CA”,  
“days”, 21
- ◆ Left side is Always in Dbl Quotes
- ◆ Right side in quotes if value is String
- ◆ Quotes NOT require if VALUE is num's

```
← → ↻ ⓘ File | D:/!Courses/322_257/!BOOK/Chapter08/c08/data/data.json

{
  "events": [
    {
      "location": "San Francisco, CA",
      "date": "May 1",
      "map": "img/map-ca.png"
    },
    {
      "location": "Austin, TX",
      "date": "May 15",
      "map": "img/map-tx.png"
    },
    {
      "location": "New York, NY",
      "date": "May 30",
      "map": "img/map-ny.png"
    }
  ]
}
```