VIA University College

Web Development 1

AJAX, XML and JSON

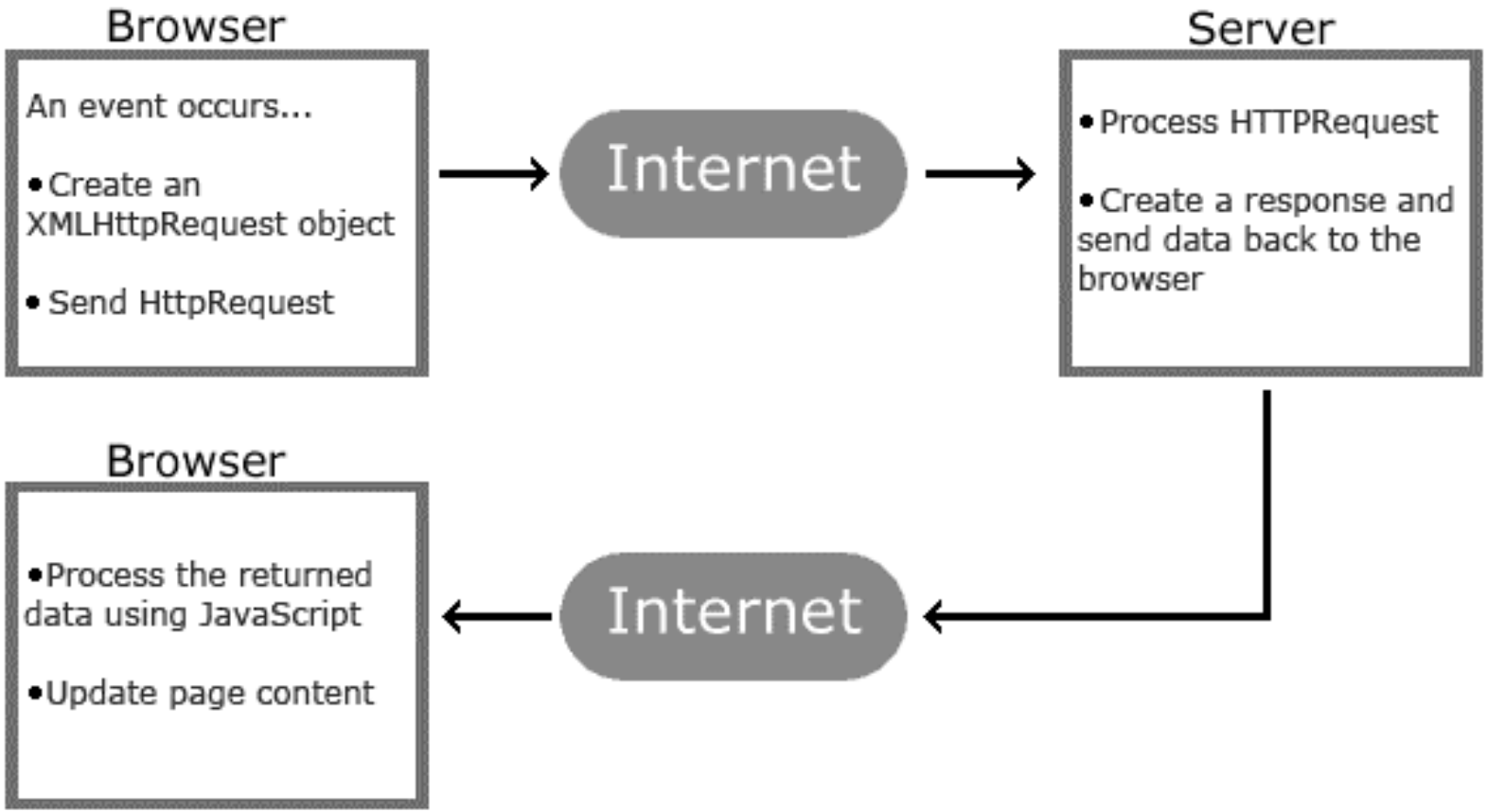
Agenda

- AJAX
- XML
- JSON

AJAX

- Not a programming language
  - A methodology/technique
  - "*AJAX is the art of exchanging data with a server, and updating parts of a web page - without reloading the whole page*"
- Stands for **A**synchronous **J**avaScript **A**nd **X**ML
- Used to access web servers from a web page
  - 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
- We want to get data from somewhere and show on our page
  - In SEP1, "somewhere" will be a local file

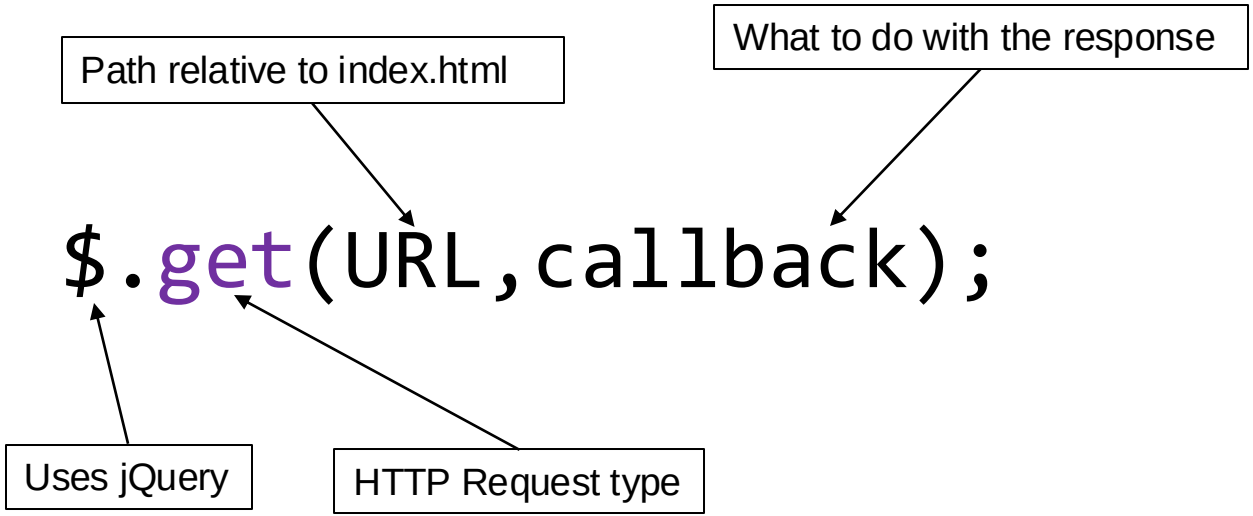
AJAX


Browser
An event occurs...
• Create an XMLHttpRequest object
• Send HttpRequest
Internet
Server
• Process HTTPRequest
• Create a response and send data back to the browser
Internet
Browser
• Process the returned data using JavaScript
• Update page content

AJAX

- `HttpRequest`
- Can be used to send, as name suggests, HTTP requests
  - GET
  - POST/PUT/DELETE/PATCH (ignored for now)
- Requests data from a specified resource
- Abstracted away in jQuery
  - `$.get`

AJAX

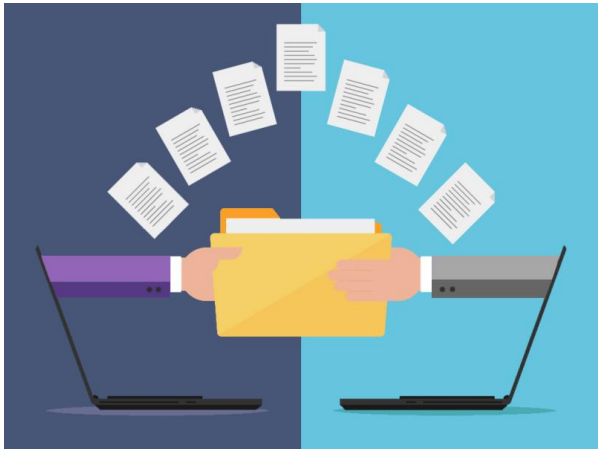

Path relative to index.html
What to do with the response
$.get(URL,callback);
Uses jQuery
HTTP Request type

AJAX

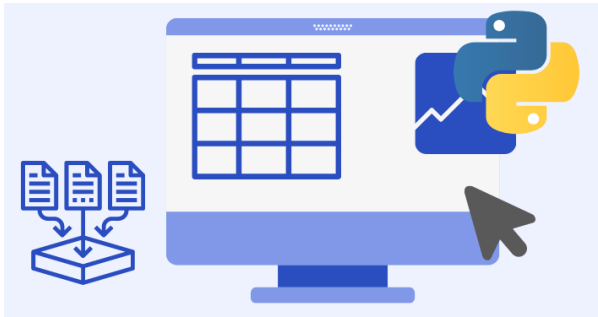
```
<p>
    When this page loads, we will retrieve some text, from some file.
    The content is placed below:
</p>
<p id="contentPlacement">
</p>
```

```
$.get("ExampleContent.txt", handleContent);
function handleContent(data, httpstatus){
    $("#contentPlacement").html(data);
}
```

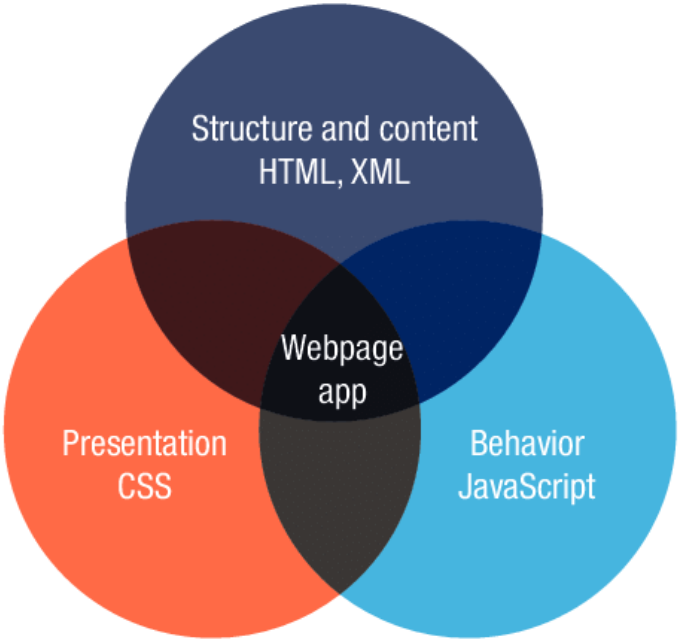
XML

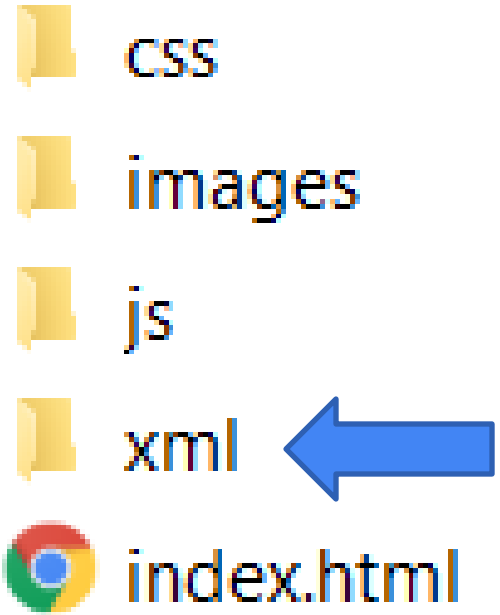


- Stands for e**X**tensible **M**arkup **L**anguage
- Designed to store and transport data
- XML in itself does not DO anything – it's just a text-format
  - Information wrapped in tags
  - Must be processed to be useful
- XML arranges data, HTML displays data



XML


Structure and content
HTML, XML
Webpage
app
Presentation
CSS
Behavior
JavaScript


css
images
js
xml
index.html

XML

```
<person>
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

XML

```xml
<person gender="female">
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

XML

```
<root>
  <person gender="female">
    <firstname>Anna</firstname>
    <lastname>Smith</lastname>
  </person>
  <person gender=”male">
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </person>
<root>
```

XML

```
<root>
  <person gender="female">
    <firstname>Anna</firstname>
    <lastname>Smith</lastname>
    <birthday>1/1-2000</birthday>
    <email>AnnaSmith@mail.com</email>
  </person>
  …
<root>
```

XML

```
<root>
  <person gender="female">
    <firstname>Anna</firstname>
    <lastname>Smith</lastname>
    <birthday>1/1-2000</birthday>
    <email>AnnaSmith@mail.com</email>
  </person>
  …
<root>
```

XML

```xml
<root>
  <person gender="female">
    <firstname>Anna</firstname>
    <lastname>Smith</lastname>
    <birthday>
      <day>1</day>
      <month>January</month>
      <year>2000</year>
    </birthday>
    <email>AnnaSmith@mail.com</email>
  </person>
  …
<root>
```

XML

```
public class Person
{
    private String firstname;
    private String lastname;
    private Birthday birthday;
    private String email;
}
```

```
<root>
  <person gender="female">
    <firstname>Anna</firstname>
    <lastname>Smith</lastname>
    <birthday>
      <day>1</day>
      <month>January</month>
      <year>2000</year>
    </birthday>
    <email>AnnaSmith@mail.com</email>
  </person>
  …
<root>
```

XML - rules

- Element names are case-sensitive
- Element names must start with a letter or underscore
- Element names cannot start with the letters xml (or XML, or Xml, etc)
- Element names can contain letters, digits, hyphens, underscores, and periods
- Element names cannot contain spaces

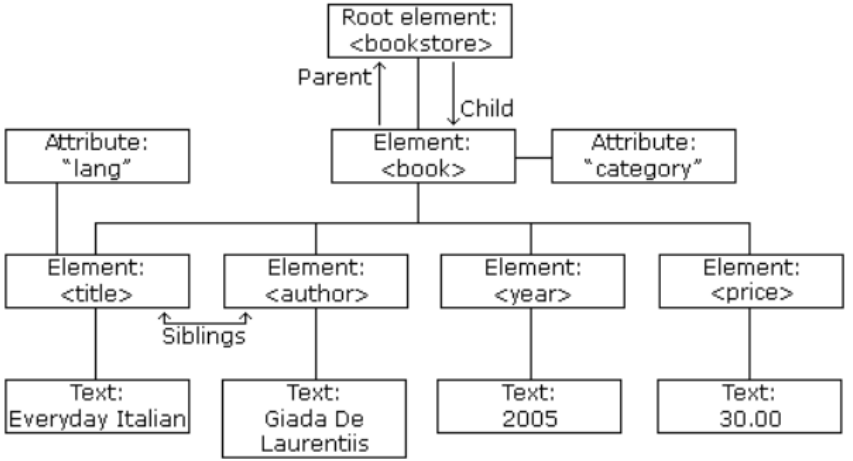
- Any name can be used, no words are reserved (except xml)
- See more here https://www.w3schools.com/xml/xml_elements.asp

XML - structure

XML code

```
<?xml version="1.0" encoding="UTF-8"?>
  <bookstore>
      <book category="cooking">
          <title lang="en">Everyday Italian</title>
          <author>Giada De Laurentiis</author>
          <year>2005</year>
          <price>30.00</price>
      </book>
…
…
  </bookstore>
```

XML tree structure



XML - Example

When this page loads, we will retrieve some text, from some file. The content is placed below:

Anna Smith John Doe

```
$.get("ExampleXml.xml", handleContent);

function handleContent(data, httpstatus){
    $("#contentPlacement").text($(data).text());
}
```

Converts `data` into a jQuery object

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <person gender="female">
    <firstname>Anna</firstname>
    <lastname>Smith</lastname>
  </person>
  <person gender="male">
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </person>
</root>
```

XML - parsing

When this page loads, we will retrieve some text, from some file. The content is placed below:

Anna Smith John Doe

- Text is just one long string
- Extract various elements
- jQuery to the rescue!
  - `.find`

XML - parsing

When this page loads, we will retrieve some text, from some file. The content is placed below:

First name: Anna

First name: John

```
$.get("ExampleXML.xml", handleContent);

function handleContent(data, httpstatus){
    let result = "";
    let persons = $(data).find("person");

    for(let person of persons)
    {
      result += "<p>";
      result += "<b>" + "First Name: " + "</b>";
      result += $(person).find("firstname" ).text();
      result += "</p>";
    }

    $("#contentPlacement").html(result);
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <person gender="female">
    <firstname>Anna</firstname>
    <lastname>Smith</lastname>
  </person>
  <person gender="male">
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </person>
</root>
```

JSON

- Stands for **J**ava**S**cript **O**bject **N**otation
- Also designed to store and transport data
- Less verbose than XML
- Ofter easier to work with

JSON

Java

```
public class Person
{
    private String firstname;
    private String lastname;
    private Birthday birthday;
    private String email;
}
```

XML

```
<root>
  <person gender="female">
    <firstname>Anna</firstname>
    <lastname>Smith</lastname>
    <birthday>
      <day>1</day>
      <month>January</month>
      <year>2000</year>
    </birthday>
    <email>AnnaSmith@mail.com</email>
  </person>
  …
<root>
```

JSON

```
{
  "firstname": "Anna",
  "lastname": "Smith",
  "birthday": {
    "day": "1",
    "month": "January",
     "year": "2000“
  },
  "email": "AnnaSmith@mail.com",
  "@attributes": {
    "gender": "female“
  }
}
```

JSON

- In XML, had to convert data to jQuery object
- With JSON, can convert to JavaScript object
- Easier to work with

JSON - parsing

When this page loads, we will retrieve some text, from some file. The content is placed below:

First name: Anna

First name: John

```
$.get("ExampleJSON.json", handleContent);

function handleContent(data, httpstatus){
    let result = "";

    for(let person of data)
    {
      result += "<p>";
      result += "<b>" + "First Name: " + "</b>";
      result += person.firstname;
      result += "</p>";
    }

    $("#contentPlacement").html(result);
}
```

```
[
{
  "firstname": "Anna",
  "lastname": "Smith",
  "birthday": {
    "day": "1",
    "month": "January",
     "year": "2000“
  },
  "email": "AnnaSmith@mail.com",
  "@attributes": {
    "gender": "female“
  }
},
…
]
```