



JavaScript Fundamentals

Week 2 - JSON – AJAX



Welcome

Manuel Cubuca

Technical Trainer

manuelc@justit.co.uk



What is JSON

JSON stands for JavaScript Object Notation

THE MOST POPULAR DATA FORMAT ON THE WEB

The JSON syntax is very similar to JavaScript object notation but it is not an **object**.

- JSON format is just plain text often used to send data from a server to a web page
- JSON is a format for sending, receiving and storing data
- JSON is language independent, code to read and generate JSON data can be written in any programming language



JSON data types

- ✓ a string
- ✓ a number
- ✓ JSON object
- ✓ an array
- ✓ a boolean
- ✓ null

JSON values cannot be one of the following data types:

- a function
- a date
- Undefined

[See valid data types](#) and also

[Exception for function and date](#)



JSON build-in functions

JSON.parse()

Used to convert any JSON received from the server into JavaScript objects.

JSON.stringify()

Used to convert any JavaScript object into JSON format, and send JSON to the server.



JSON resources

JSON FORMATER AND VALIDATOR

JSON FORMATTER CHROME ADD-ON

W3SCHOOLS JSON INTRODUCTION

JSON FILES ON GITHUB

Now let's practice



What is AJAX

AJAX stands for **A**synchronous **J**ava**S**cript **A**nd **X**ML

Ajax is simply a Web Development Technic, used to **SEND** and **RETRIEVE** data in the background

Without having to refresh the web page

The data is often sent in a format called JavaScript Object Notation (**JSON**).



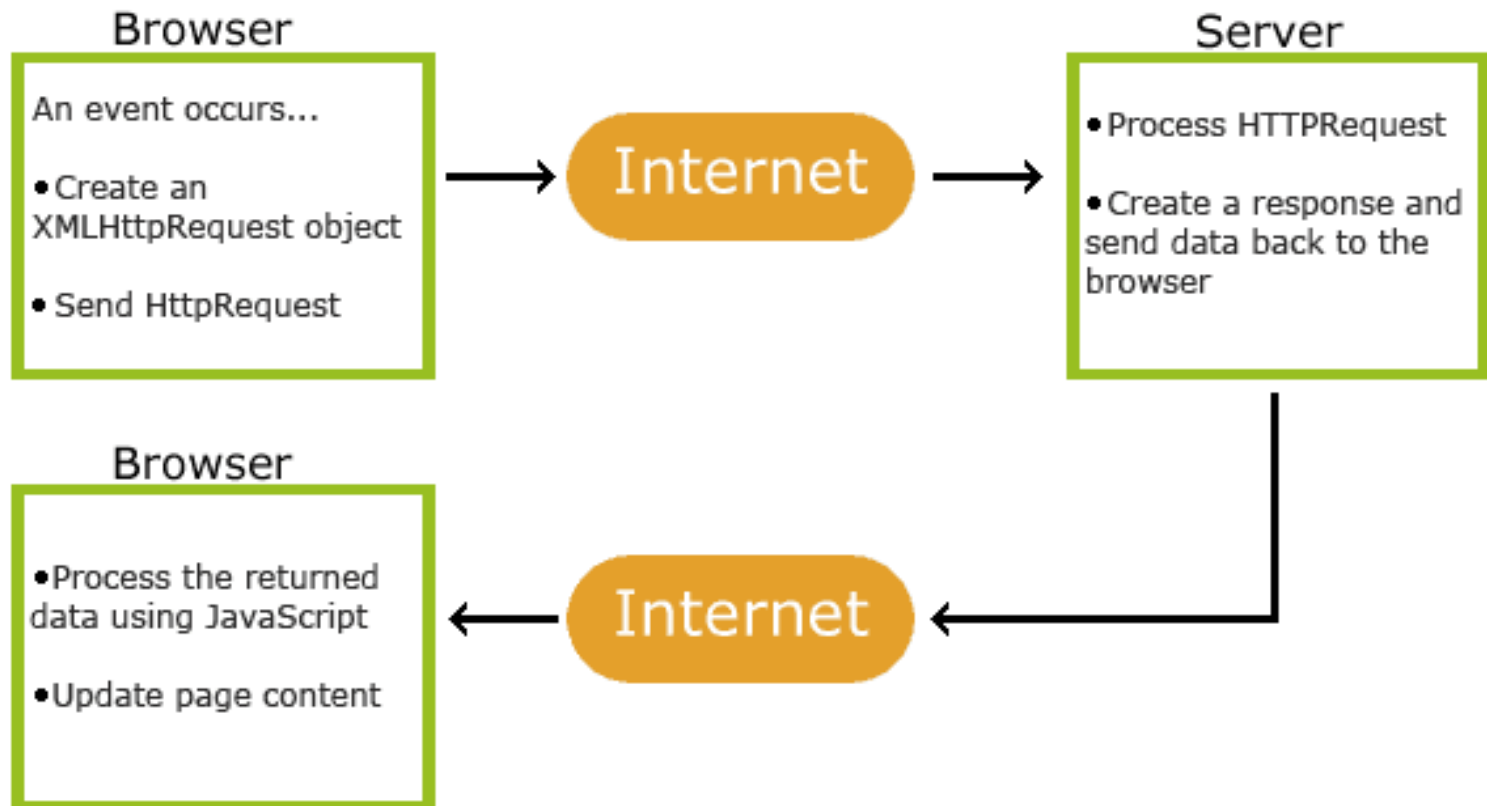
What is AJAX cont.

AJAX stands for **A**synchronous **J**ava**S**cript **A**nd **X**ML

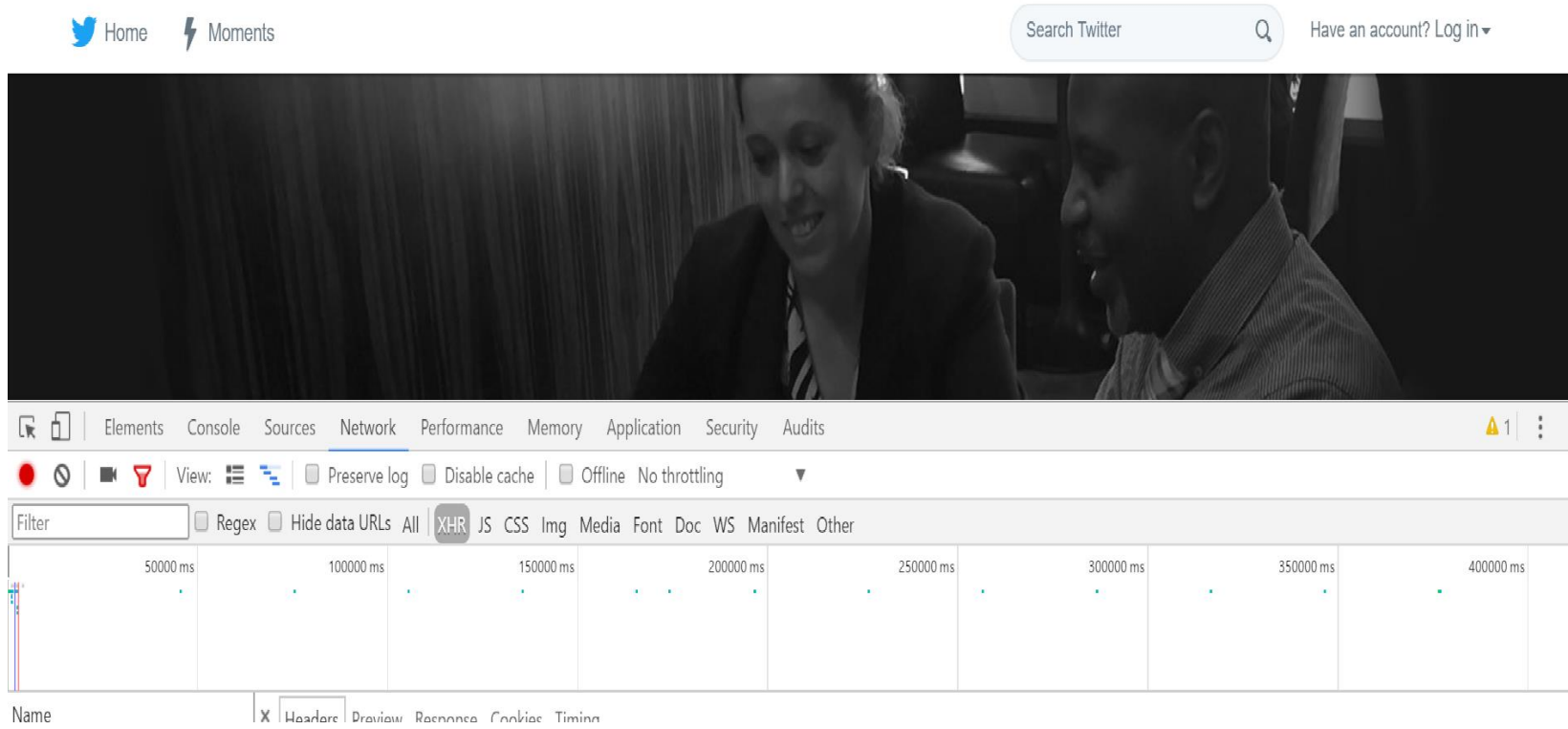
- AJAX is not a programming language
- AJAX uses a browser build-in **XMLHttpRequest** object (to request data from a server)
- AJAX uses an **asynchronous** processing model, improving and speeding up the user experience



How AJAX Works



How AJAX Works cont.



The AJAX Request

- 1 `var xhr = new XMLHttpRequest(); // browser build-in object`
- 2 `xhr.open("GET", "data/myData.json", true); // POST (send)`
- 3 `xhr.send();`

1. Creates a new **instance** of the XMLHttpRequest object and stores the object in a variable
2. The XMLHttpRequest object's **open()** method prepares the request. The 3 parameters are: the HTTP method, the URL of the page and the Boolean to indicate asynchronous
3. Uses the **send()** method, is the one that sends the prepared request to the server. Information can be passed to the server in the parentheses, you may also see `xhr.open(null)`



The AJAX Response

```
1 xhr.onload = function() { // could use onreadystatechange  
  
2   if (this.readyState == 4 && this.status == 200) {  
  
       // Code to process the results from the server  
   }  
};
```

1. When the browser has received and loaded a response from the server, that will trigger the anonymous function
2. The function checks the **readyState** and the **status** property of the object, to make sure the server's response was OK, also the data has arrived and available



The parse() & stringify() Methods

```
1 xhr.onload = function() { // could use onreadystatechange  
2   if (this.readyState == 4 && this.status == 200) {  
3       var myData = JSON.parse(this.responseText);  
    }  
};
```

3. **JSON.parse()** processes a string containing JSON data. It converts the JSON data into a JavaScript objects ready for the browser use. **JSON.stringify()** converts a JavaScript object into a string formatted using JSON, allows to send JavaScript objects from the browser to another application



AJAX Code Example

```
function loadMyData() {  
  
    var myRequest = new XMLHttpRequest();  
  
    myRequest.open("GET",  
    "https://raw.githubusercontent.com/biatoSalo/JSONANDAJAX/master/JSONANDAJAX/expensiveLuxuryCars.json", true);  
  
    myRequest.onload = function() {  
        if (myRequest.readyState == 4 && myRequest.status == 200) {  
            var myData = JSON.parse(myRequest.responseText);  
        }  
    };  
    myRequest.onerror = function() {  
    }  
    myRequest.send();  
}
```



XMLHttpRequest Methods

Method	Description
<code>new XMLHttpRequest()</code>	Creates a new XMLHttpRequest object
<code>abort()</code>	Cancels the current request
<code>getAllResponseHeaders()</code>	Returns header information
<code>getResponseHeader()</code>	Returns specific header information
<code>open(<i>method</i>, <i>url</i>, <i>async</i>, <i>user</i>, <i>psw</i>)</code>	Specifies the request <i>method</i> : the request type GET or POST <i>url</i> : the file location <i>async</i> : true (asynchronous) or false (synchronous) <i>user</i> : optional user name <i>psw</i> : optional password
<code>send()</code>	Sends the request to the server Used for GET requests
<code>send(<i>string</i>)</code>	Sends the request to the server. Used for POST requests
<code>setRequestHeader()</code>	Adds a label/value pair to the header to be sent



XMLHttpRequest Properties

Property	Description
onreadystatechange	Defines a function to be called when the readyState property changes
readyState	Holds the status of the XMLHttpRequest. 0: request not initialized 1: server connection established 2: request received 3: processing request 4: request finished and response is ready
responseText	Returns the response data as a string
responseXML	Returns the response data as XML data
status	Returns the status-number of a request 200: "OK" 403: "Forbidden" 404: "Not Found" For a complete list go to the Http Messages Reference
statusText	Returns the status-text (e.g. "OK" or "Not Found")



AJAX resources

W3SCHOOLS AJAX INTRODUCTION

MOZILLA DEVELOPER NETWORK - AJAX

Use the JSON practice example to practice AJAX

Now hands-on coding

