How to use web services with JavaScript

Course Topic:	**Course Language:**	**Course Instructor:**	**Session:**
Setup, asynchronous loading	English	Vanessa Al Daham	1/4

Let's get to know each other!

- Name

- Place of work and current position

- Hobbies and interests?

- Why did you choose this training?

- What do you except as an outcome?

JavaSript

- Client-Side scripting language
- Runs on the browser
- Can manipulate the webpage

- But, often we rely on external data that we

  cannot manage (out of scope).

Softwares to be used

WampServer

Sublime text 3

Where to create our project ?

➢ Go to your Local disk, open Wamp folder.

➢ Go to "WWW" folder and create your projects folder there.
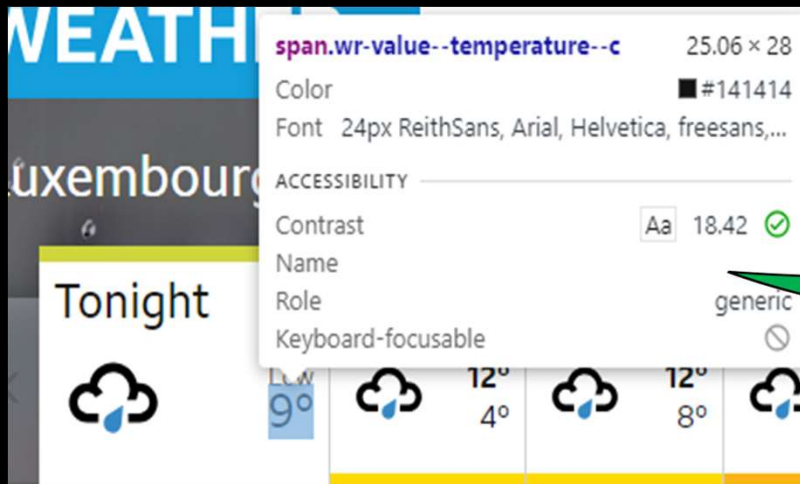
Let's Take an example!

- You are creating a webpage where you'ld like to show the outdoor temperature in degrees Celsius currently in Luxembourg

- This information already exists, and many websites maintain these information such as:
  Luxembourg - BBC Weather

- Let's see how to Make It Happen

Let's Make It Happen!

1. Find a website that has the needed information (meteolux.lu)
2. Find the information within the website. Inspect it for an id or class attribute

Let's Make It Happen!

3. Using JavaScript, try to fetch that data using an **asynchronous** data exchange.

How to Make It Happen?

Let's dig into it.

Asynchronous data exchange(AJAX)

➢ Fetching data asynchronously has the advantage that the website does not need to be refreshed and can be used during the fetching process. Specific parts of the website can be reloaded instead of the whole document.

➢ JavaScript has a native method but we will be using the **jQuery** framework since it makes it much easier.
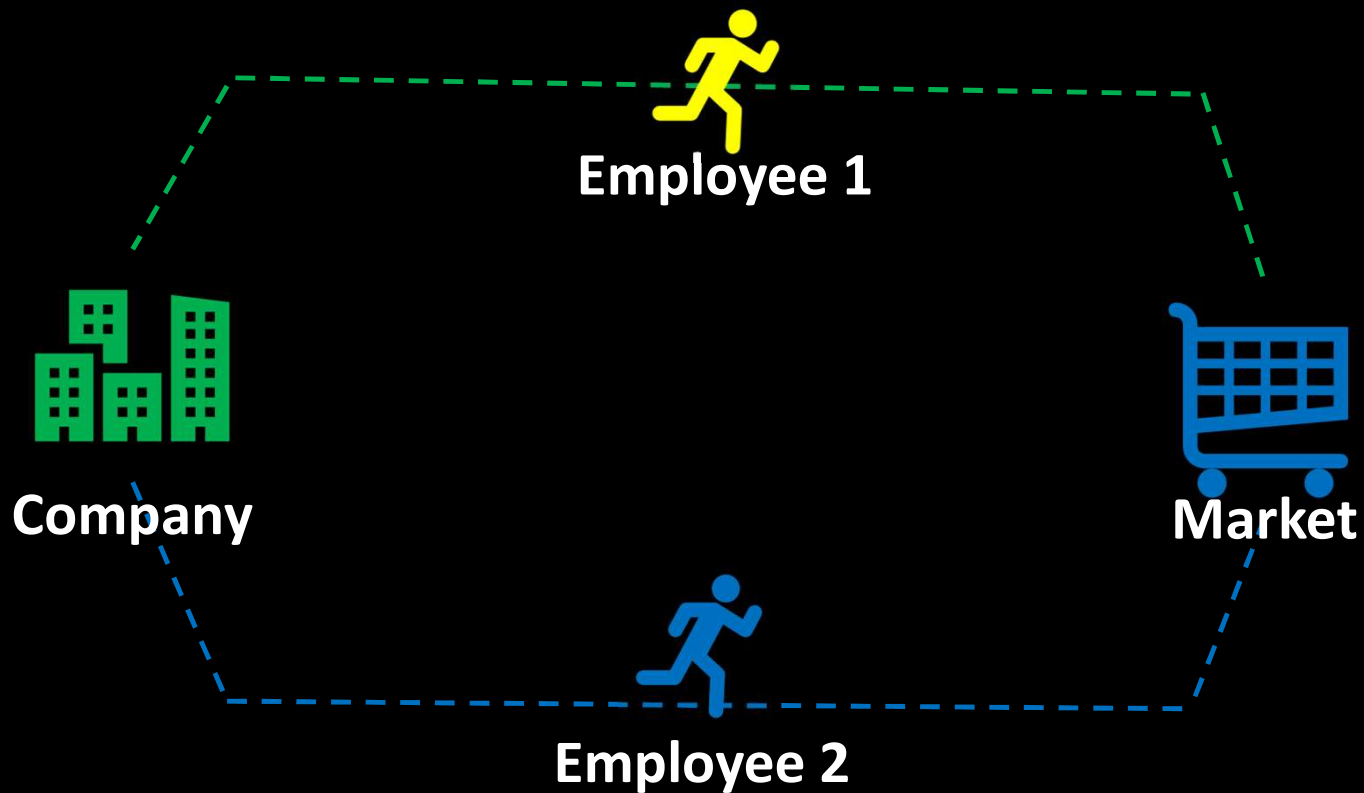
JQuery

➢ We can download jQuery and include it manually.

➢ Or, by embedding a CDN link such as: https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js

Let's get back to our task

Let's have a look on the following code :

```
1   <html>
2   <head>
3       <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js">
4
5       </script>
6
7       <script>
8           $(document).ready(init);
9               function init(){
10                  $("#divOutput").load("https://www.bbc.com/weather/2960316.temperature:first");
11              }
12
13      </script>
14  </head>
15
16  <body>
17      <div id="divOutput"></div>
18  </body>
19
20  </html>
21
```

Weather Webpage Problem

The previous code show the following error:


```
Console   Issues   +

top ▼   ⊘   Filter   Default levels ▼   💬 12   ⚙

❌ Access to XMLHttpRequest at 'https://www.bbc.com/weather/ weather.html:1
   2960316.temperature:first' from origin 'http://localhost' has been blocked
   by CORS policy: No 'Access-Control-Allow-Origin' header is present on the
   requested resource.

❌ ▶ GET https://www.bbc.com/weather/2960316.temperatur  jquery.min.js:2
   e:first net::ERR_FAILED 404 (Not Found)

>
```

Let's dig into the error.

CORS-Problem

Weather Information – CORS Problem :

- **Cross-Origin Resource Sharing**.

- Client restricted to fetch information from websites with different origin (security measure).

- To prevent JavaScript to run in the background and send user data to some servers (to the hackers.)

- Clients are hard to be trace.

- **Only servers are capable** of fetching information from different origins.

CORS-Solution

If only servers are capable of fetching information; Then let the server do so.

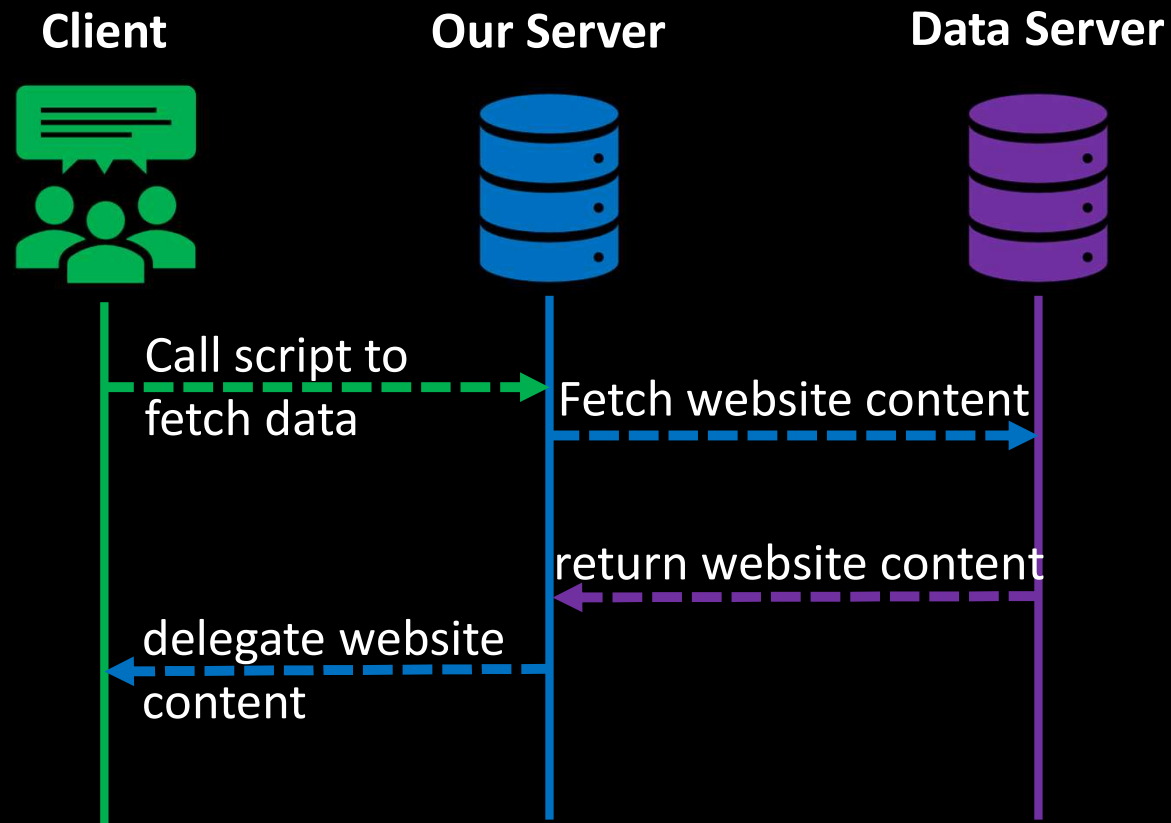**Let's Create our server using PHP (server language):**

```php
<?php
    echo file_get_contents("https://www.bbc.com/weather/2960316");
?>
```

CORS-Solution

Let's Create update our JS code :

```html
1   <html>
2   <head>
3       <script
4       src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js">
5       </script>
6
7       <script>
8           $(document).ready(init);
9               function init(){
10                  $("#divOutput").load("weather.php .wr-value--temperature--c:first");
11              }
12
13      </script>
14  </head>
15
16  <body>
17      <div id="divOutput"></div>
18  </body>
19
20  </html>
```

Worksheet-Session 1

- ❑ Solve exercises 1-5
- ❑ You can Find the givens on GitHub :

https://github.com/VanessaDH/JSWS/blob/1139c2f00b4432f20fac38443f5eb732520fab59/DLH_JSWS_Worksheet_1.pdf

Callbacks

➤ Sometimes fetching the data might take some time.

➤ Giving the user a visual feedback is

recommended so that he/she knows when

the data retrieval is finished.

Callbacks-First Approach

```javascript
// 1) show BEFORE load

$("#divLoading").html("<img src='loading.gif'>");

// 2) data retrieval

$("#divOutput").load("ex6.php .ktzuAh");

// 3) remove AFTER load

$("#divLoading").html("");
```

First Approach-Problem

- ➢ JavaScript runs one command after another and it does not wait until it finishes

- ➢ The loading gif is being displayed, then the data is being fetched and before the data arrives, JavaScript executes the last line, i.e. to remove the loading spinner

- ➢ Solution: We need to tell JavaScript to wait for the response of the server!

Callbacks-Better Approach

```
// 1) show BEFORE load
$("#divLoading").html("<img src='loading.gif'>");
// 2) data retrieval
$("#divOutput").load("ex6.php .ktzuAh", loadingDidFinish);

function loadingDidFinish( ){
        // 3) remove AFTER load
        $("#divLoading").html("");
}
```

Callbacks-Better Approach

- ➤ A second parameter can be specified in the load method

- ➤ This parameter is called Callback and

  represents the function that will be called

  automatically when the load method

  finishes, i.e. retrieves the data from the

  server.

Continue Worksheet-Session 1

☐ **Solve exercises 6.**