JS HTTP Requests

JS & HTTP Requests

- Why?
  - Performance
  - User Experience

- XMLHttpRequest
  - AJAX
    - Basic Concept
    - yes is is all about asynchronous calls

- JS Libraries

XMLHttpRequest

- For simplicity the “old” onstatechange pattern will be used

```
http.onreadystatechange = function() {//Call a function when the state changes.
```

- The “newer” addEventListener pattern will be presented later
  - Links for event listener
    - https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/load_event

Yes, there is a “newer” way -> fetch

We will discuss this later

```
<!DOCTYPE html>

<html>

<body>

<p id="demo">Fetch ralph.txt and place it here </p>

<script>

fetch ("ralph.txt") .then(x => x.text()).then(y => document.getElementById("demo").innerHTML = y);

</script> </body> </html>
```

Basic Design

1. Creating request object
2. Setting parameters of request object & submitting it
3. Synchronous or **asynchronous** handling of response

Documentation

- Docs
    - https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/open
- Links for event listener
    - https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/load_event

Example -> make aysn. call to node post

Node Code

```
app.post('/sayHello', (req,res) => {
       var name = req.body.name;
       var answer = "hello " + name;
        res.send(answer);
  });
```

```
function sayHello()
{
   var http = new XMLHttpRequest();
   var url = '/sayHello';
   var params = 'name=' + document.getElementById("input").value;

   http.open('POST', url, true);

   http.setRequestHeader('Content-type',
'application/x-www-form-urlencoded');
http.onreadystatechange = function()
   {
       if(http.readyState == 4 && http.status == 200)
       {

     document.getElementById("answer").value = http.responseText;
       }
   };

   http.send(params);
};
```

- request.readyState
- https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/readyState
-
    - 0 -> unsent
    - 1 -> opened
    - 2 -> headers_received
    - 3 -> loading
    - 4 -> done

JSON?

- JavaScriptObjectNotation
- JSON.parse
  - String -> Object
- JSON.stringify
  - Object -> String