Advanced JavaScript

Lecture 4

Client side VS Server Side

- Step1 : Sending Request
- Step 2 : Application processes the request.
- Step 3 : Receiving response HTML , CSS and JavaScript.
- Step 4 : Browser render the response.

How web works ?

- To request a page from the server, the user normally just type in the URL of the page he wants to open. Or he can click a link or a button to redirect him to the desired page.
- The browser will actually send a HTTP request to the server asking it to return the desired page.
- When the browser receives the response from the server, it will render it to the user and close the connection with the server.
- When the user initiates a new request to the server, the browser will send a new request to the server and the whole page will reload to render the new response.

What is HTTP ?

- HTTP : is a protocol on which the web is based on. Is standard for : Hyper Text Transfer Protocol.

- Types of HTTP requests:
  - GET request.
  - POST request.
  - PUT request.
  - DELETE request.

AJAX

- Ajax stands for Asynchronous Javascript And XML.
- Ajax is used to send HTTP requests asynchronously to the server.
- It can be used to fetch or send data from the server and then display the response in a section in the page.

Synchronous and Asynchronous

- Synchronous operations block instructions until the task is completed, while asynchronous operations can execute without blocking other operations.
- Making synchronous calls to resources can lead to long response times locking up the UI until the resource responds

XMLHttpRequest - syntax

- Create XMLHttpRequest Object: var xhr = new XMLHttpRequest();
- Initialize it xhr.open(method, URL, boolean) ;
  - method – HTTP-method. Usually "GET" or "POST".
  - URL – the URL to request, Usually a string.
  - async – if explicitly set to false, then the request is synchronous,
- Send it out: xhr.send([body]);
- Please note that open call, contrary to its name, does not open the connection. It only configures the request, but the network activity only starts with the call of send.

XMLHttpRequest - syntax cont.

- send method opens the connection and sends the request to server. The optional body parameter contains the request body.
- Some request methods like GET do not have a body. And some of them like POST use body to send the data to the server.

XMLHttpRequest - Listeners

- These three events are the most widely used:
  - load – when the request is complete (even if HTTP status is like 400 or 500), and the response is fully downloaded.
  - error – when the request couldn't be made, e.g. network down or invalid URL.
  - progress – triggers periodically while the response is being downloaded, reports how much has been downloaded.

XMLHttpRequest - Response

- Once the server has responded, we can receive the result in the following xhr properties:
  - Status - HTTP status code (a number): 200, 404, 403 and so on, can be 0 in case of a non-HTTP failure.
  - statusText - HTTP status message (a string): usually OK for 200, Not Found for 404, Forbidden for 403 and so on.
  - Response - (old scripts may use responseText)

XMLHttpRequest - URL search parameters

- To add parameters to URL, like `?name=value`, and ensure the proper `encoding`,
  - `var url = new URL('https://google.com/search');`
  - `url.searchParams.set('q', 'test me!');` // the parameter `'q'` is `encoded`
  - `xhr.open('GET', url);` // https://google.com/search?q=test+me%21

XMLHttpRequest - Ready states

- XMLHttpRequest changes between states as it progresses. The current state is accessible as xhr.readyState.
  - UNSENT = 0; // initial state
  - OPENED = 1; // open called
  - HEADERS_RECEIVED = 2; // response headers received
  - LOADING = 3; // response is loading (a data packed is received)
  - DONE = 4; // request complete

XMLHttpRequest - Ready states track

1. We can track xhr states using readystatechange event.

```
xhr.onreadystatechange = function() {
  if (xhr.readyState == 3) {
    // loading
  }
  if (xhr.readyState == 4) {
    // request finished
  }};
```

XMLHttpRequest - ReadyStatesChange

NOTE:

You can find readystatechange listeners in really old code, it's there for historical reasons, as there was a time when there were no load and other events. Nowadays, load/error/progress handlers deprecate it.

XMLHttpRequest - Aborting request

We can terminate the request at any time. The call to xhr.abort() does that:

```
xhr.abort();          // terminate the request
```

That triggers abort event, and xhr.status becomes 0.

XMLHttpRequest - HTTP-headers

XMLHttpRequest allows both to send custom headers and read headers from the response.

There are 3 methods for HTTP-headers:

- setRequestHeader(name, value)
- getResponseHeader(name)
- getAllResponseHeaders()

Lab

- Exercise 1
  - Part 1

    - Suppose we have to do list App.
    - **list** the todos tasks received from the server without any sort.
    - If the task status is complete , set the card background with green otherwise set it yellow.

Lab

- Part 2
  - Create drop down list that can filter the task based on :
    - Task status: with values completed or in Progress .
      - If user select completed you should list only the tasks with the flag completed is true.
      - If user select in progress you should list only the tasks with the flag completed is false.
    - User ID :
      - For example is user select 1 : list only tasks that userId= 1 have.

Lab

- Part 3
  - Create Reset filters button , if user clicked on it , you should reset all filters and show all tasks again without any sort.

NOTE:

Todos list url: https://jsonplaceholder.typicode.com/todos

Method: GET