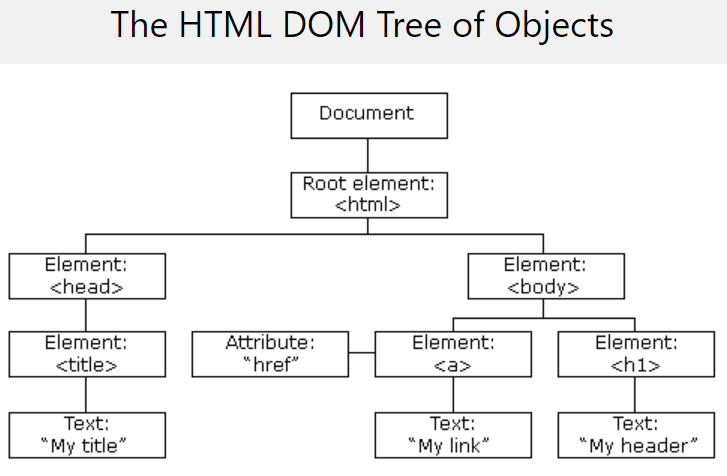
## JavaScript, DOM, JSON, AJAX and JPA

General part

* **Explain about the Document Object Model, and why it’s extremely relevant for modern Web-development**

[**https://www.w3schools.com/js/js\_htmldom.asp**](https://www.w3schools.com/js/js_htmldom.asp)

* + When a web page is loaded, the browser creates a Document Object Model (DOM) of the page
  + With the object model, JavaScript can access and change all the elements and attributes of an HTML document also the CSS.
  + With the object model, JavaScript can remove and add new HTML elements and attributes and create dynamic HTML.
  + With the object model, JavaScript react to all existing HTML event in the page
  + With the object model, JavaScript can create ned HTML events in the page



* **Explain how JavaScript fit’s into modern Web Development**

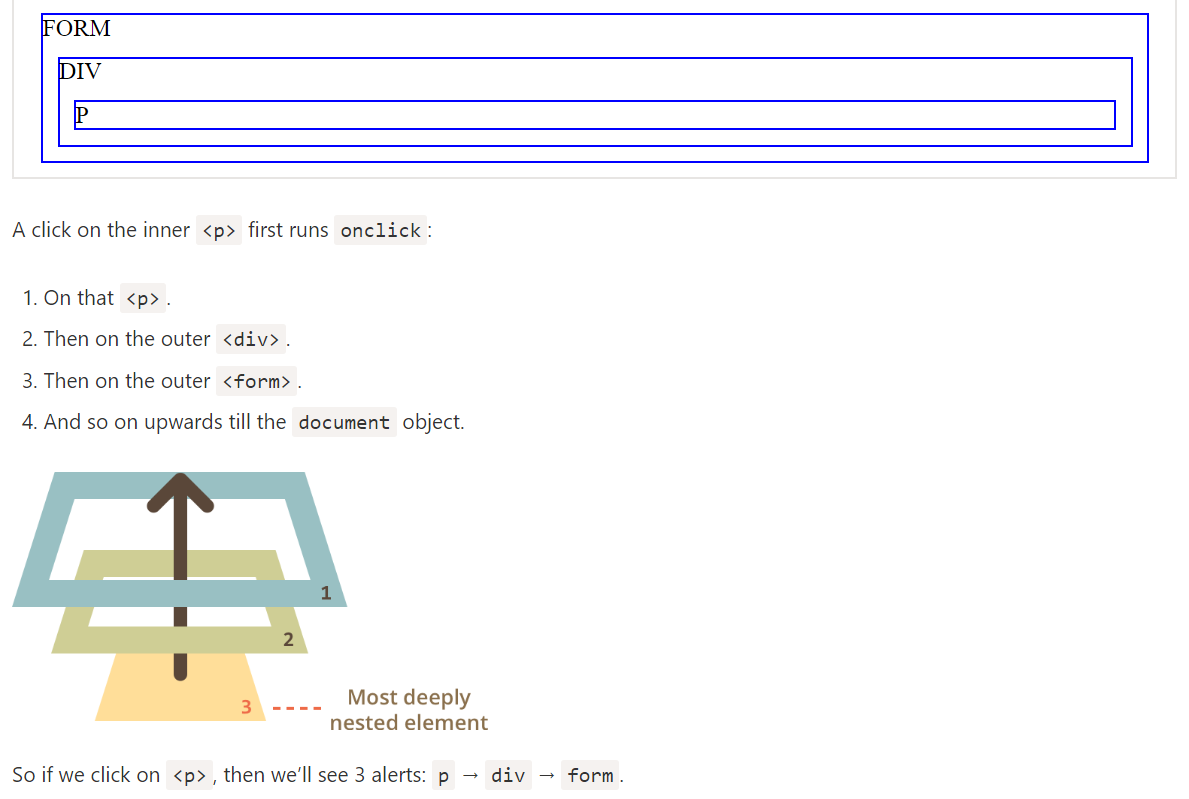
[**https://en.wikipedia.org/wiki/JavaScript**](https://en.wikipedia.org/wiki/JavaScript)

* + With todays tempo of changing it is necessary easy to get access to the different elements and attributes
  + The amount of data is enormous and need flexibilities to handle and work with
  + JavaScript (JS) er et dynamisk programmeringssprog. Det er mest almindeligt anvendt som en del af webbrowsere, hvis implementeringer tillader klientside scripts at kommunikere med brugeren, kontrollere browseren, kommunikere asynkront, og ændre indholdet af HTML dokumenter. JS anvendes også ved server-side programmering (med Node.js), spiludvikling og udvikling af desktop- og mobile applikationer.
  + Et JavaScript-bibliotek udmærker sig ved at gøre udviklingen af web applikationer lettere, ved at give et simplere og mere kraftfuldt API at arbejde med, i forhold til Document Object Model(DOM) og browser specifikke API'er. Et af de nyeste og mest brugte biblioteker er jQuery, men andre populære er Dojo og YUI, se henvisning nedenfor.
* **Explain (using an example of your own choice) about JavaScript events, and Event Bubbling**

[**https://javascript.info/bubbling-and-capturing**](https://javascript.info/bubbling-and-capturing)

[**https://www.w3schools.com/js/js\_htmldom\_events.asp**](https://www.w3schools.com/js/js_htmldom_events.asp)

* + Events are things that happen to HTML elements
  + JavaScript can react on these events (onchange, onclick, onmouseover, onmouseout, onkeydown, onload)
  + Event handlers can handle and verify user inputs actions and browser actions.
  + HTML event attributes can execute JavaScript-code directly and call JavaScript functions
  + Event bubbling runs the handler on an element, then on its parent, then all the way up

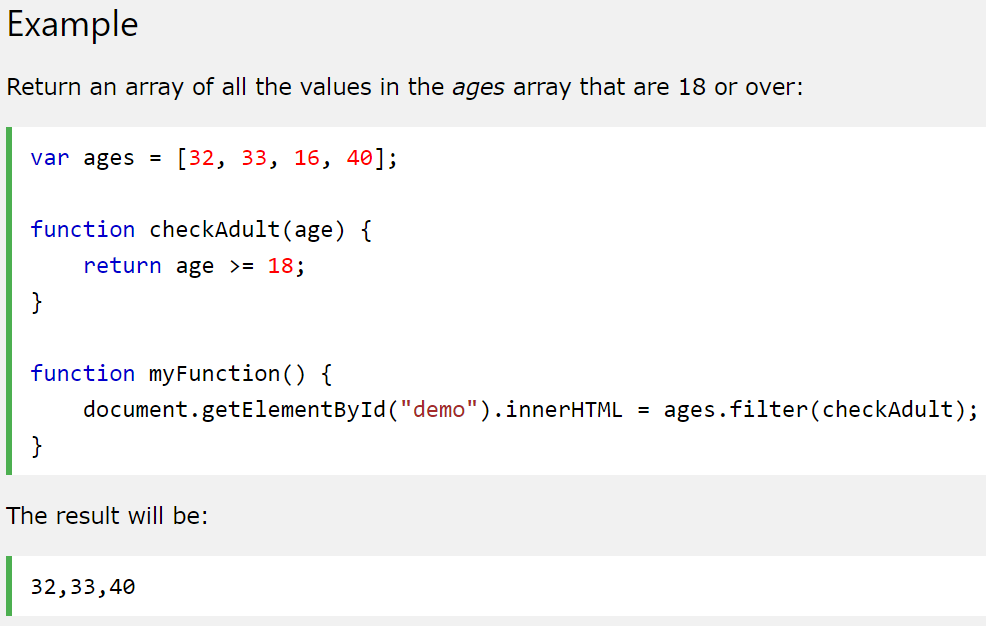


* **Explain (in words) the purpose of the JavaScript-arrays filter and map methods (also, provide a few examples)**

**Filter:** [**https://www.w3schools.com/jsref/jsref\_filter.asp**](https://www.w3schools.com/jsref/jsref_filter.asp)

**Map:** [**https://www.w3schools.com/jsref/jsref\_map.asp**](https://www.w3schools.com/jsref/jsref_map.asp)

* + Filter(): creates an array filled with all array elements that pass the test
  + Does not execute the function for array elements without values
  + Does not change the original array



* + The map() method creates a new array with the results of calling a function for every array element.
    - Laver et callback for hvert af elementerne i arrayet fx array-length er 10, foretages der 10 calbacks.
  + The map() method calls the provided function once for each element in an array, in order.
  + Note: map() does not execute the function for array elements without values.
  + Note: map() does not change the original array.
* **Explain the topics AJAX and how it has changed the way modern web-applications are created**

[**https://www.w3schools.com/js/js\_ajax\_intro.asp**](https://www.w3schools.com/js/js_ajax_intro.asp)

* + Asynchronous JavaScript and XML
  + Set of web technologies
  + Send and receive data asynchronously
  + Does not interfere with current web page
  + JSON has replaced XML for the most parts
* **Elaborate on how JSON or XML supports communication between subsystems, even when the subsystems are implemented on diﬀerent platforms.**

[**https://www.w3schools.com/js/js\_json\_xml.asp**](https://www.w3schools.com/js/js_json_xml.asp)

* + JSON and XML is basically just plain text.
  + It is not platform specific and therefore any system that receives this format can work with it
  + The strength is that the implementation on the server side is irrelevant when what is being communicated from client-server is JSON/XML. The webservice can have been made in Java, C#, C++, Python and so on and still function as long as we use JSON or XML when parsing data from client-server.

CA or Semester Project

For a real exam exercise, this will be a small part where you are expected to talk, in about 5 minutes, about the semester project or one of the semester CA’s (related to the topic for this question).

Practical part start (use this [index.html file](https://github.com/Cphdat3sem2017f/StartcodeExercises/blob/master/JS/index.html) as your start template)

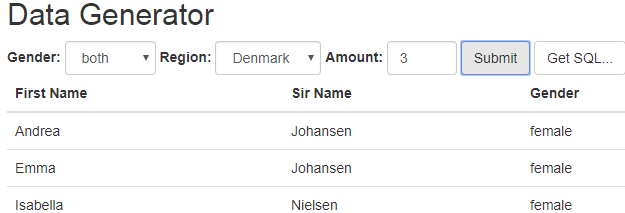
Design a simple SPA using plain JavaScript, and fetch(for server requests). The application should allow users to create test data representing a person. Initially these data must be presented in a table, but the page must include a “Get SQL ..” button to convert the test data into a valid SQL INSERT Script, presented in the TextArea given in the start code.

**Getting started:**  Create a web-project (in whatever way you prefer) and include the index.html file given above in the project. “Run” the file in a browser to see the provided layout for the exercise. Observe script-tag, in the bottom. Create this file (app.js) and place all your JavaScript in this file.

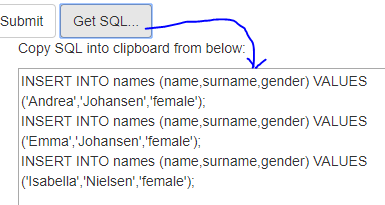
**1)** You must use this public [REST API](https://github.com/thm/uinames/) to get data for the exercise:

Copy the link below into your browser and test. To get a feeling of how to use the API, try (as a minimum) to: remove *region*, change *gender* to *male*, remove *gender*, change *amount* to 600;

<http://uinames.com/api/?amount=25&region=denmark&gender=female>

**2)** Add the necessary code to (using fetch, the API given above, and DOM-manipulation) render a table as sketched in this figure, when the “Submit” button is pressed. 

**2a)** Add a way to present errors for the users (try and request more than 500 names). Hint: Check if the response code is >= 400.

**3)** If not already done, find a way (the simplest and acceptable for this exercise, is a global variable, but feel free to come up with better alternatives) to store the data fetched above, so we can reuse it in it’s original form.

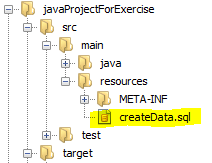
**4)** Add the necessary code to convert (when Get SQL is pressed) the data into valid MySQL syntax that will insert data, into a table with matching columns as sketched in this figure. Insert the generated SQL into the TextArea provided with the startcode.

*Hints: You can insert the SQL into the TextArea like this: document.getElementById("sql").value = sql;*

*If you have stored the data as requested in 3, and know how to use* ***map*** *and* ***join****, this should be relatively simple ;-)*

**5)** Using the generated SQL-script

This steps assumes step 1-4 is complete (if not see hints at the bottom). Now lets create a JPA-application with an Entity class, which rows are populated from the script you get from step-4.

Create a new plain Java Maven project. In this project create an Entity Class with properties matching those inserted into the script created in step4.

**6)**

In this project, create a file createData.sql in the exact location as sketched in this figure:

Paste the Script, generated via the feature implemented in step-4, into this script.

Add this line to your persistence.xml file (set the file to drop-and-create):

<property name="javax.persistence.sql-load-script-source" value="createData.sql"/>

Create the schema, for example by executing: Persistence.generateSchema("YOUR-PU-NAME", null);

Verify that data is inserted into the table, either by executing a select all, up against the database or, better, by creating a JPQL-Query that will fetch all data

*Hint: If you had problems with step-4, you can Create the script manually, with just a few names, and use that script for these steps. When/if this works, go back to step-4, and use this working script as template for what you have to create.*