Seminar 4

Internet Applications, ID1354

Jacob Kimblad, jacobki@kth.se

1019-01-13

1 Introduction

The final seminar gives an introduction to client-side scripting using Javascript. It aims to make the author more familiar with interacting with the Document Object Model (DOM) through the use of the Javascript library JQuery and AJAX. The idea is to sen requests through Javascript to update information on the server and on the client such as ther user session and recipe comments. The requirements for the assignment are summarised in the list below.

- AJAX is used for login and logout.
- AJAX is used for reading, writing and deleting comments.
- No HTML code, but only data, is sent as a response to an AJAX request.
- Recipe comments are stored on the client side (for example in the DOM).

2 Literature Study

Since this is the first encounter with javascript in the course the Mozilla JavaScript reference *JavaScript reference* [1] is a good resource to have at hand. Since the jQuery library will be used its API is also necessary to keep at hand. The jQuery documentation is very extensive and well written and can be found at js.foundation [2]. The jQuery documentation also includes the documentation for AJAX, which specifically can be found at js.foundation [3].

3 Method

No new tools were needed during this seminar other than the javascript library jQuery which is fetched on the client side from googles Content Delivery Network (CDN). Other than that Codeigniter is still used for the backend and the LAMP stack consists of Arch Linux, Apache, MariaDB and PHP. The texteditor used is Vim and DBeaver is used for database management. Firefox will be used as the browser during development and Chromium will also be used during testing to check for cross-browser compability.

4 Result

All of the JavaScript was written into a single file that is served to the user along each page loaded by the user visiting the website. The Javascript file is located under the assets folder and can be directly found at [4]. The file consists of five event handlers and one function. The first event handler [5] is tied to the page viewed by the user and its callback-function is called once the DOM is ready to be used through executing JavaScript code. All of the four other event handlers are located within the callback of this event handler as they are tied to elements on the page. The single function [6] is also located within the event handler for the document and is used to display the comments on each of the recipe pages. The function is also called [7] within the event handler for the document, so once the DOM is ready to be interacted with it is made sure that all of the comments are loaded onto the page through the DOM.

To show the comments an AJAX request is first made to the server to receive all the comments in the database as well as the username of the currently logged in user. Each comment on the page consists of two list item HTML elements, one showing the author of the comment and one showing the acutal comment. To add these elements we append them [8] through jQuery to an existing div with an assigned id, this is done for each comment and they are in such a way saved in the DOM. When adding a comment to the DOM there is also a check if the author of the comment is the same as the logged in user (passed through the response to the AJAX request) and if so is the case a delete button is added underneath that specific comment using the DOM.

Each delete button is also tied to an event handler [9] that sends a GET request using AJAX once pressed. The GET request calls the delete function in the comments controller on the server with one argument passed in the URL. The argument is the id of the comment tied to its row in the database such that the model can know which row to delete. The server also makes sure that the comment actually belongs to the currently logged in user. Since the delete buttons are loaded after the rest of the page through an AJAX call their event handler cant be attached as soon as the DOM is ready as they wont exist yet. This is the reason for using the function .on() instead of .click() to attach an event handler to these buttons. After the call is completed the current page is reloaded using the location.reload() function in JavaScript.

The logout button in the navbar is also attached to an event handler [9] that makes a call to the logout function in the users controller using AJAX. Since all of the logic for the logout is dealt with on the server this function doesnt need to send or receive any further data, but simply call the provided URL using AJAX. After the call is succeded the location.reload() function is called in JavaScript to reload the page.

A third event handler [10] is attached to the login button located on the login page. This event handler submits a POST request to the function loginUser located in the users controller once the login button is pressed. It makes sure to load the information that has been inputed by the user in the username and password input boxes and send to the server. Once the request is succesfull the page is reloaded using JavaScript. Which page is served to the user during the reload is then decided by the server depending on if the login was succesfull or not.

The final event handler [11] is connected to the submit comment button. An AJAX request is then made to the function create in the comments controller on the server providing the comment inputted by the user in the textarea and what recipe page the comment was made on. Once the AJAX request has been made succesfully the page is reloaded using JavaScript to show the new comment.

5 Discussion

The requirements were to use JavaScript and AJAX to handle login and logout of the user as well as reading, writing and deleting comments. An additional requirement was that no HTML code, but only data was to be sent as a response to an AJAX request and the recipes comments should be stored on the client side. The results show how two event handlers tied to the login and logout buttons are used to make calls to functions in a controller on the server side to login and logout the user. Two additional handlers were used to add and delete comments by making calls to different functions in a different controller on the server. An additional AJAX call is also made each time the DOM of the page has been loaded to fetch all existing comments in the database from the server such that they can be added to the DOM using JavaScript and shown to the user and are in this way stored on the client side. As the comments from the database are sent to the client in the AJAX response using JSON representation no HTML is sent as a response to the AJAX. Thus all of the requirements have been reached.

6 Comments About the Course

The author spent around 20 hours on this seminar.

References

[1] *JavaScript reference*, sv-SE. [Online]. Available: https://developer.mozilla.org/sv-SE/docs/Web/JavaScript/Reference (visited on 01/13/2019).

[2] J. F.-. js.foundation, *jQuery API Documentation*, en-US. [Online]. Available: http://api.jquery.com/ (visited on 01/13/2019).

[3] ——, *Ajax — jQuery API Documentation*, en-US. [Online]. Available: https://api.jquery.com/category/ajax/ (visited on 01/13/2019).

[4] [Online]. Available: https://github.com/jkimblad/ID1354/blob/master/seminar4/app/assets/scripts/ajax_requests.js.

[5] [Online]. Available: https://github.com/jkimblad/ID1354/blob/b4d619aca4f51e3245871be122f2
seminar4/app/assets/scripts/ajax_requests.js#L1.

[6] [Online]. Available: https://github.com/jkimblad/ID1354/blob/b4d619aca4f51e3245871be122f2
seminar4/app/assets/scripts/ajax_requests.js#L77.

[7] [Online]. Available: https://github.com/jkimblad/ID1354/blob/b4d619aca4f51e3245871be122f2
seminar4/app/assets/scripts/ajax_requests.js#L5.

[8] [Online]. Available: https://github.com/jkimblad/ID1354/blob/master/seminar4/app/assets/scripts/ajax_requests.js.

[9] [Online]. Available: https://github.com/jkimblad/ID1354/blob/b4d619aca4f51e3245871be122f2
seminar4/app/assets/scripts/ajax_requests.js#L54.

[10] [Online]. Available: https://github.com/jkimblad/ID1354/blob/41edefcada72eee2d55b3942d078
seminar4/app/assets/scripts/ajax_requests.js#L40.

[11] [Online]. Available: https://github.com/jkimblad/ID1354/blob/41edefcada72eee2d55b3942d078
seminar4/app/assets/scripts/ajax_requests.js#L8.