POZNAŃ UNIVERSITY OF TECHNOLOGY

FACULTY OF CONTROL, ROBOTICS AND ELECTRICAL ENGINEERING

INSTITUTE OF ROBOTICS AND MACHINE INTELLIGENCE

DIVISION OF CONTROL AND INDUSTRIAL ELECTRONICS

WEB APPLICATION jQUERY AND CSS

MOBILE AND EMBEDDED APPLICATIONS FOR INTERNET OF THINGS

TEACHING MATERIALS FOR LABORATORY

DOMINIK ŁUCZAK, PH.D.; ADRIAN WÓJCIK, M.SC.

DOMINIK.LUCZAK@PUT.POZNAN.PL
ADRIAN.WOJCIK@PUT.POZNAN.PL

I. Goal

Knowledge

The aim of the course is to familiarize yourself with:

- role of JavaScript libraries in web applications on the example of jQuery,
- syntax and available functions of jQuery library,
- AJAX technique in the context of jQuery,
- basics of CSS syntax,
- using CSS libraries on the example of Bootstrap.

Skills

The aim of the course is to acquire skills in:

- creating web applications using the AJAX technique,
- creating web applications using the jQuery library,
- defining the style of the web application user interface using CSS.

Social competences

The aim of the course is to develop proper attitudes:

- proper management of web application code base,
- strengthening understanding and awareness of the importance of non-technical aspects and effects of the engineer's activities, and the related responsibility for the decisions taken,
- choosing the right technology and programming tools for the given problem,
- testing developed IT system.

II. Laboratory report

Complete laboratory tasks as per the instructor's presentation. Work alone or in a team of two. **Keep safety rules while working!** Prepare laboratory report documenting and proving the proper execution of tasks. Editorial requirements and a report template are available on the *eKursy* platform. The report is graded in two categories: tasks execution and editorial requirements. Tasks are graded as completed (1 point) or uncompleted (0 points). Compliance with the editorial requirements is graded as a percentage. The report should be sent as a *homework* to the *eKursy* platform by Sunday, May 23, 2021 by 23:59.

III. Prepare to course

a) Know the safety rules

All information on the laboratory's safety instructions are provided in the laboratory and on Division website [1]. All inaccuracies and questions should be clarified with the instructor. It is required to be familiar with and apply to the regulations.

Attend the class prepared. Knowledge from all previous topics is mandatory.

b) Introduction to jQuery

jQuery is a JavaScript library that primarily simplifies the use of the HTML Document Object Model (DOM). Important jQuery functionalities are also: event handling, animation and style support, and a set of methods implementing the AJAX technique [2] . It is free *open source* software using the MIT License (X11 License) [3]. The jQuery library is used by almost 3/4 of websites (as of April 2020) [4].

jQuery has a simple syntax in which three basic elements can be distinguished:

- the character '$' (dollar sign) denotes the alias of the jQuery library (it can be replaced with the word `jQuery`, however this notation is not typically used in practice),
- a selector in the form `("selector")` where the attribute *selector* is the name of the HTML element, the ID of the HTML element (with the symbol '#' at the beginning) or the class of the HTML element (with the symbol '.' at the beginning).
- action (function) name after the dot.

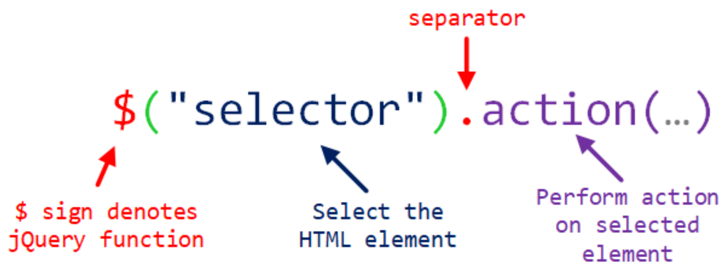
separator
$("selector").action(…)
$ sign denotes jQuery function
Select the HTML element
Perform action on selected element

Fig. 1. The structure of the jQuery syntax.

Fig. 1 shows the structure of the jQuery syntax. Detailed API documentation with use case examples is available on the project website [5].

c) Asynchronous JavaScript and XML

AJAX (Asynchronous JavaScript and XML) is *technique* (not stand-alone *technology*) for creating web applications using HTML documents represented by DOM, JavaScript script language and CSS style sheets. Thanks to the properly implemented AJAX model, web applications are able to make quick, incremental updates in the user interface without the need to reload the entire web page. As a result, the application seems faster and gives a better user experience [6]. AJAX technique, despite the name, is increasingly giving up the XML data format in favor of the JSON data format.

A typical scenario for using the AJAX technique is as follows: [7]:

1. Event takes place on the website (e.g. a button has been pressed).
2. XMLHttpRequest object is created by JavaScript.
3. XMLHttpRequest object sends a request to a web server (e.g. an IoT server).
4. Server application is processing the request.
5. Server sends the response back to the website.
6. Response is read and decoded by JavaScript.
7. Proper action (usually interacting with the HTML document) is performed by JavaScript.

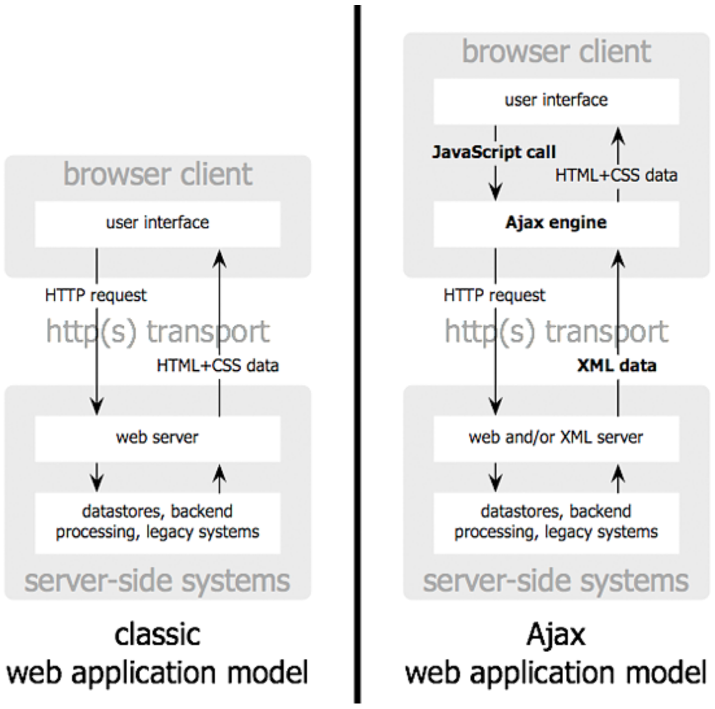

Fig. 2. Web application model: classic vs. AJAX [8].

Fig. 2 presents a comparison of the classic web application model with the AJAX model. This technique is often used in cooperation with the REST architecture, which in turn allows for obtaining clearly defined and legible client-server communication.

D) Introduction to CSS

Cascading Style Sheets (CSS) is a language that describes *style* of an HTML document, i.e. how HTML elements are displayed. The name „cascading” comes from the priority scheme specified in CSS to determine which style rule applies if more than one rule matches a particular element. Separation of content (HTML) and formatting (CSS) allows document with the same markers structure to be presented in different styles for different rendering methods, including alternative formatting if e.g. the content is accessed on a mobile device [9][10]. CSS has different *levels* and *profiles*. Each level of CSS builds on an earlier level, usually adding new features; they are currently marked as CSS 1, CSS 2, CSS 3 and CSS 4.

Styles defined using CSS can be placed directly in an HTML document using the `<style>` tags or by using an external file (also the web repository) using the `<link>` tag (Listing 1).

Listing 1. An example of the header of an HTML document with CSS.

```html
<!-- Style definiton in HTML document -->
<style>
        h1 { color: blue; }
</style>

<!-- Style definiton in file/repository -->
<link href="path/to/file.css" rel="stylesheet" type="text/css">
```

Fig. 3 shows the structure of CSS syntax. As with the jQuery library selectors in CSS, we can define the style of an HTML element by its name, ID (with the symbol '#' at the beginning) or class (with the symbol '.' At the beginning). On the W3C consortium website can be found language documentation along with a list of supported atributes of HTML elements [11].

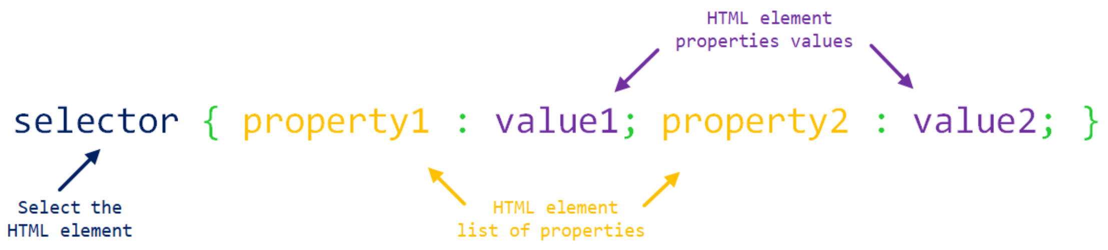

Fig. 3. CSS syntax structure.

E) CSS library example: Bootstrap

One of the reasons for the huge popularity of CSS technology is the availability of many libraries facilitating creation of aesthetic and ergonomic interfaces of web applications with little effort from the developer. An example of a popular, free and open CSS library is Bootstrap [12]. This library contains CSS-based design templates and (optional) JavaScript for typography, forms, buttons, navigation and other user interface elements. Fig. 4 presents an example of a simple textitdata grabber application developed using one of Bootstrap template.

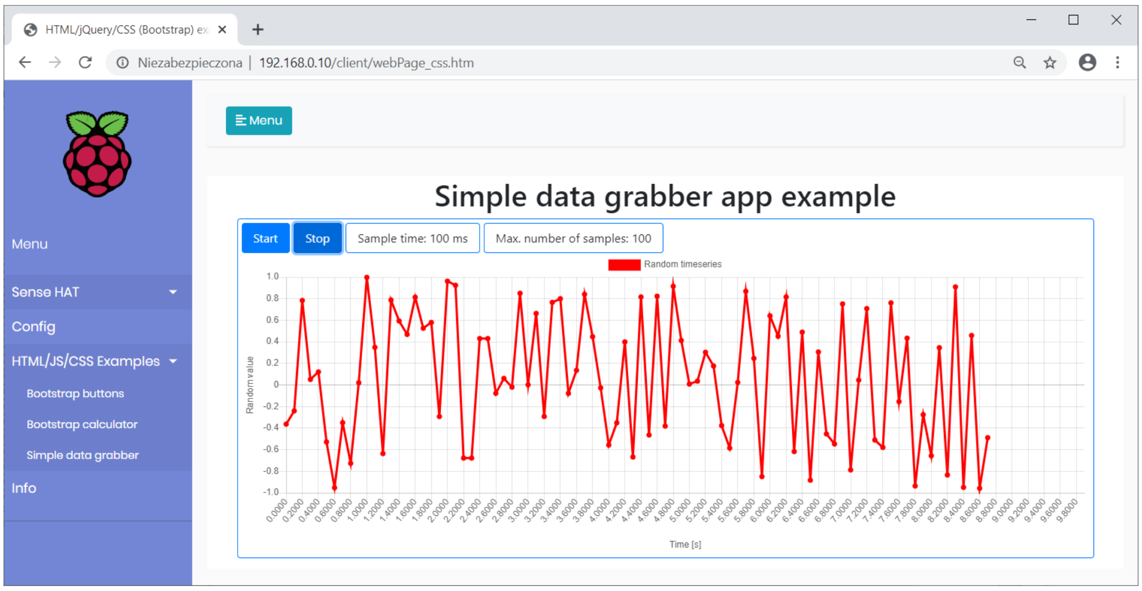

Fig. 4. An example of a web application with Bootstrap.

F) Examples

Sample web apps - HTML documents with JavaScript scripts - are available in repository github.com/adrianwojcikpp/AMiWdIP-L. Examples can run by opening HTML documents with web browser. However, for security reasons, the default configuration of modern web browsers follow *Same-Origin Policy* [13]. Therefore, apps that perform web requests (*web clients*) should be accessed via the same domain / IP address as server resources. In context of laboratory tasks, refer to all available examples. The *FIR Example* application is an example of a client *synchronously* requesting measurement data from the server and performing their digital filtering. This is also an example of using Chart.js to handle charts, as well as an example of using the jQuery `get` method with the keywords `async` and `await`.

IV. Scenario for the class

a) Teaching resources

Hardware	• computer,
Software	• web browser (eg. Firefox, Chrome),
	• text editor (eg. Notepad++),
	• web server (eg. XAMPP, Lighttpd),

b) Tasks

Familiarize yourself with examples of mobile applications available in GitHub repository. Create a web application using the AJAX technique. Apply CSS to format the style / theme of your documents.

1. Create REST client applications for a web browser (Firefox, Chrome, etc.) for embedded system measurement data visualization.
   (a) The interface should contain timeseries plots of *at least two* measurement quantities, e.g. angular orientation (RPY) or temperature, pressure and humidity from the Sense Hat add-on. Put on the chart all the necessary information for the correct and unambiguous interpretation of the data. Use previously prepared graphic interface prototype designs.
   (b) The application should *cyclically* request data from server and display responses in the form of timeseries plot. The sample time should be defined by the user.
   (c) Run and test app on a selected web browser. Make sure the network communication is working properly. You can use a physical or virtual embedded device for testing, or a server *mock* in the form of a local CGI script generating random synthetic measurement data.
2. Expand client application with module (web page) to control user output device (e.g. LED display).
   (a) The interface should allow setting the state of all available user outputs (e.g. the LED matrix from the Sense Hat add-on). The interface should clearly communicate whether the current user settings correspond to the output states (i.e. whether the user has applied the last changes). Use previously prepared graphic interface prototype designs.
   (b) The application should send a request to the server containing control commands. Use an adequate HTTP method.
   (c) Run and test app on a selected web browser. Make sure the network communication is working properly. You can use a physical or virtual embedded device for testing, or a server *mock* in the form of a local CGI script saving control commands to text file.
3. Expand your application with a module (web page) to configure and save user settings.
   (a) The configuration page should contain basic application settings, such as port number, server API version (convenient for the developer), requests sample time, maximum number of saved samples, etc.
   (b) Configuration information should be saved on **server** as a JSON text file.
   (c) After reloading web page, saved user's settings should be read from the server.

References

1. *Regulations, health and safety instructions* [online]. [N.d.] [visited on 2019-09-30]. Available from: http://zsep.cie.put.poznan.pl/materialy-dydaktyczne/MD/Regulations-health-and-safety-instructions/.
2. JS.FOUNDATION, JS Foundation-. *jQuery* [online]. [N.d.] [visited on 2020-04-27]. Available from: https://jquery.com/. Library Catalog: jquery.com.
3. *jQuery JavaScript Library* [online]. [N.d.] [visited on 2020-04-27]. Available from: https://github.com/jquery/jquery/blob/master/LICENSE.txt. Library Catalog: github.com.
4. *Usage Statistics and Market Share of jQuery for Websites, April 2020* [online]. [N.d.] [visited on 2020-04-27]. Available from: https://w3techs.com/technologies/details/js-jquery.
5. JS.FOUNDATION, JS Foundation-. *jQuery API Documentation* [online]. [N.d.] [visited on 2020-04-27]. Available from: https://api.jquery.com/. Library Catalog: api.jquery.com.
6. *Asynchronous JavaScript and XML* [online]. [N.d.] [visited on 2020-04-28]. Available from: https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX. Library Catalog: developer.mozilla.org.
7. *AJAX Introduction* [online]. [N.d.] [visited on 2020-04-28]. Available from: https://www.w3schools.com/xml/ajax_intro.asp.
8. *Jacek Kobusiński - AJAX w przykładach* [online]. [N.d.] [visited on 2020-04-28]. Available from: http://www.cs.put.poznan.pl/jkobusinski/ajax.html.
9. *Cascading Style Sheets* [online]. [N.d.] [visited on 2020-04-19]. Available from: https://www.w3.org/Style/CSS/.
10. *Learn to style HTML using CSS* [online]. [N.d.] [visited on 2020-04-19]. Available from: https://developer.mozilla.org/en-US/docs/Learn/CSS. Library Catalog: developer.mozilla.org.
11. *CSS Snapshot 2018* [online]. [N.d.] [visited on 2020-04-19]. Available from: https://www.w3.org/TR/CSS/#intro.
12. THORNTON, Jacob; OTTO, Mark. *Bootstrap* [online]. [N.d.] [visited on 2020-04-28]. Available from: https://getbootstrap.com/. Library Catalog: getbootstrap.com.
13. *Same-origin policy - Web security | MDN* [online] [visited on 2021-05-06]. Available from: https://developer.mozilla.org/pl/docs/Web/Security/Same-origin_policy.