Mandatory assignment – JavaScript – part 1

IMPORTANT:

- All the requirements are mandatory to implement in order to pass.
- You are required to use the knowledge that have been thought at the lecture, from the syllabus book, and further resources that were suggested at the lecture. Your assignment will be evaluated against these resources. Each task has references to some of these resources.
- More information regarding for example failing marks and plagiarism are given in Blackboard, on the “Mandatory assignments” page.

General considerations

Create **ONE** web page with HTML, CSS, and JS similar in complexity with the examples given to you on the pages 80 – 83 and 141 – 143 of the syllabus book “JavaScript & Jquery. Interactive front-end development.”

NOTE for the IIKG1002 students: continue to work with the web page that you have created for the first assignment by adding JavaScript to it and eventually extra HTML/CSS needed to implement the JavaScript part.

You are free to choose **the topic, the content, and the styling** for your page.

- The topic should be different than the ones given as examples (e.g., hotels, custom signs) in the syllabus book. The topic could be instead related, for example, to your free time activities and interests.
- The styling should be at the level of the example given in the book. However, do not copy the styling from the book, but create your own, to fit the topic you chose.

Delivery

All the files for your website should be included in **one** folder named:
IIKG1002_mandatoryAssignment2_StudentName /
IDG1011_mandatoryAssignment1_StudentName

NOTE for the IIKG1002 students: copy over the files from the first assignment to continue to work with these.

This folder should be packaged as **a .zip file** and delivered in Blackboard.

This folder should contain:

- ☐ a JavaScript file: *script.js*
- ☐ a *styles.css* file with the CSS code
- ☐ an *index.html* file with the HTML code

- ☐ a *scriptDesign.pdf* file containing the design of your script
- ☐ a folder with images

I. Defining a goal and designing the script

In a separate file – this can be created in any program you like, e.g., Microsoft Word, Google Sketchboard:

a) define the goal of your script,
b) break the goal in a series of tasks (in a ordered list) that have to be performed step-by-step to achieve the goal,
c) sketch out the tasks in a flowchart

Sources: Ch. 1/a in the book "JavaScript & Jquery. Interactive front-end development." and lecture notes from week 5 (p. 16 – 21)
Delivery: A single .pdf file containing the flowchart drawing, the goal, and tasks.

II. JavaScript implementation

a) Use an external script file. Do the same for the CSS file.

   Resources:
   For a reminder on how to link JavaScript files, see pp. 47 – 49 and 51 in the syllabus book "JavaScript & Jquery. Interactive front-end development."

   as well as Lecture notes from week 5, pp. 47 – 50

b) The path to the file should be a relative one (not absolute); this concerns also the paths to images and the CSS file.

c) Use the good practices taught in the class, and the ones in this guide: https://developer.mozilla.org/en-US/docs/MDN/Writing_guidelines/Writing_style_guide/Code_style_guide/JavaScript

   NOTE: for writing comments follow the guides in the task **II. Comments** and NOT the ones in the guide.

Your script must contain the following:

1 Objects

Examples of objects that you could create:

- For a page about movies, your can model the movies as objects.
- For a page about poems, you can model each poem as an object.
- For a page about visiting a city, you can model each activity/attraction as an object.

Resource: Lecture notes from week 8, pp. 2 – 11, and the resources recommended there, p. 11.

a) Create at least **three objects** using a class.

   The class should initialize at least **three properties** in the constructor function and include at least **one additional method** (besides the constructor).

b) The objects should contain at lest **one property** that has a more complex value, such as an array or another object.
   Resources: pp. 118 – 119, Duckett book on JavaScript, or Lecture notes from week 7, p. 13.

c) Display information about your objects to the user interface.

   Resources: see the examples from pp. 110 – 111, Duckett book on JavaScript.

2 Built-in objects

Resources:
For a reminder of the theory related to built-in objects:

→ Lecture notes from week 8,

→ pp. 120 – 139 in the Duckett book on JavaScript.

TO DO:

a) Use **one** property **or** method of the *window object* to add functionality/ information that is relevant for your web page. You are free to choose the property/method.

   Resources: See p. 124 for a list of properties/methods you can choose from

   Example of usage: Use the *window.location* to display to the user the URL for your page, for sharing purposes.

b) Use **one** property **or** method of the *document object* to add functionality/ information that is relevant for your web page. You are free to choose the property/method.

   Resources: See p. 126 for a list of properties/methods you can choose from

   Example of usage: Use the *document.lastModified* to display to the user the date on which your page was last modified.

c) Use **one** property **or** method of the *string object* on at least one of the string values that you have in your web page.

   Resources: See p. 128 for a list of properties/methods you can choose from

Example of usage: Use the method *toUpperCase* to display the names of the products in uppercase characters.

d) Use **one** method/property of the *number object* **OR** the *Math object.*

Resources: See p. 132 / p. 134 for a list of methods/properties you can choose from

Example of usage: Use the method *Math.random()* to randomly choose from an array which product to display as feature product.

e) Use **one** method of the *Date object* to get/set the date and time for the Date object that you have created.

Resources: See p. 137 for a list of methods you can choose from

Example of usage: Use the method *getFullYear()* to display the date your web page was created, as you can see in the example on the p. 138.

II. Comments

The comments must

a) explain in detail your implementation,
b) explain your reasoning around the choices you make in your implementation, and
c) show your understanding of the theory that you have learned from the syllabus and lectures, as applied to your specific implementation, by citing this in your comments
d) include references to the lecture notes, syllabus books, and Mozilla Developer (include page numbers)
e) refer to the task number the respective code is implementing,
e.g., `// Implementation of the task II.1.a`

Sources: See an example of commented code on Blackboard under “Mandatory assignments” / Writing comments in your code: *comments_example.js*
as well as the Lecture notes from week 6, pp. 10 – 11