T5 - Web Development Seminar

T-WEB-500

Day 07

Javascript

2.1

The firsts 7 tasks are provided as HTML pages.
You have to interact with the HTML code given to solve the exercises.
These very HTML files will be used to correct your exercises, so make sure that your JavaScript files do not require to edit the HTML code.

No edition of HTML code is necessary, in fact you must not add them in your delivery.

The use of external libraries or any code that you may have copy/pasted from the Internet is forbidden.

Starting from task 08, the JQuery library is authorized.

Task 01

Turn in: `script/task01.js`
Used resource: `task01.html`

Make sure that with each click in the white box, a counter displays the number of clicks in this block.

Task 02

Turn in: `script/task02.js`
Used resource: `task02.html`

Make a dialog box appear, prompting "What's your name ?" when one clicks on the white block.
If no name is filled, the box must keep displaying until a name is filled. Once a name is filled, make a confirmation box appear, displaying "Are you sure that **name** is your name ?"
If the name is confirmed, an alert box with the content "Hello **name** !" must be displayed. That content will also be displayed in the white box.

Task 03

Turn in: `script/task03.js`
Used resource: `task03.html`

Display in the white box the last 42 characters entered from the keyboard on this page.

Task 04

Turn in: `script/task04.js`
Used resource: `task04.html`

The + and - buttons must allows respectively to increase or decrease the page's font size.

The dropdown menu must allows to change the background color of the page.

Task 05

Turn in: `script/task05.js`
Used resource: `task05.html`

Draw a white triangle inside the canvas with a 1px border and the following coordinates: {6, 6}, {14,10}, {6, 14}.

As you click on the play button below the music which can be found in the following URL must play: music.

Finally, make the control buttons for Pause, Stop and Mute work as expected.

Task 06

Turn in: `script/task06.js`
Used resource: `task06.html`

Make sure you can drag and drop the black square inside the white box.
Then, display in the second white box the relative position of the black square, replacing the "?" respectively by their x and y coordinates.

Task 07

Turn in: `script/task07.js`
Used resource: `task07.html`

When the link in the white box is clicked a cookie “acceptsCookies” will be created. It must expire in 1 day. The cookie will have the value `true`.

If the cookie is defined and if its value is true, then the message in the white box must not be displayed. However, you will need to make a second white box appear with a button “Delete the cookie”. When this button is clicked, the second white box and the cookie must be deleted, and the first message must reappear.

From now on, we will focus on the basics of jQuery.
For each exercise, you are asked to create a ".js" file and the appropriate `index.html` to show that your script works.

In the following exercises, the instructions must be executed after the page is fully loaded.

If you feel you are doing a bit of magic, fear not!!! That's because you are doing some magic!
Remember, you are the master of all things, a master among masters.
The DOM must bow before you.

PS: Try not to get yourself hurt

Task 08

Turn in: `task08/index.html`, `task08/selector.js`

This JS file must contain a function that selects all elements of hyperlink type that do not have the target="_blank" attribute and makes them semi-transparent with 50% opacity.

Task 09

Turn in: `task09/index.html`, `task09/event.js`

This JS file must contain a function that assigns a "click" event on the first "button" element of the page. The action of the event must make all paragraphs of the page disappear.

Task 10

Turn in: `task10/index.html`, `task10/append.js`

Add a text input with the id “listItem” in your page and a button.
Add a function which will be called every time a click happens on this button with the input’s content as parameter.
This function must add a div after this element, that contains the value of the element passed as parameter.

Task 11

Turn in: `task11/index.html`, `task11/blue.js`

This JS file must contain a function that, when hovering over a paragraph, adds the “blue” class to it.
In addition, you should be able to include or remove a paragraph from the “highlighted” class with a single mouse click.

Task 12

Turn in: `task12/index.html`, `task12/script.js`, `task13/style.css`

Create a form with two input : a text input and a dropdown and a button to submit the form. When the form is submitted, add the text to a bulleted list displayed under the form.
Define a few CSS classes:
* a **note** class, putting the border of the element in blue,
* an **email** class, putting the border of the element in green,
* a **todo_** class, putting the border of the element in red.
The dropdown of the form should contains the three options: note, email and todo.
When an element is added to the bulleted list it must have the correct css class assigned to it based on the option selected in the dropdown.

TASK 13

Turn in: `task13/index.html`, `task13/script.js`, `task13/style.css`

This task is a direct follow-up of the preceding one.

Implement a form that will be used to perform searches amongst the created items. The form should have a dropdown, a "search" button and a "reset" button

As of now you only need to handle a search by types (you will implement a search by words in the following exercise) using a dropdown input with the 3 options.

Searching for "email", only the elements being "email" must stay displayed (the same goes for the other options)
The others must not be deleted and must be shown again when the search is resetted.

TASK 14

Turn in: `task14/index.html`, `task14/script.js`, `task14/style.css`

This task is a direct follow-up of the preceding one.

Upgrade your search feature.
It should now be possible to search by a word or a part of word contained in the elements.
For example, searching for "rick" when there are a "patrick@something.com" email and a "Send a mail to Rick" todo, both should be displayed.

Add the possibility to combine your search with words and types, so in the previous example, one could choose to search only todos, which would only display "Send a mail to Rick".

Task 15

Turn in: `task15/index.html`, `task15/script.js`, `task15/style.css`

Make it possible to add tags to an element of the list.
The tags must also be displayed.
It should be possible to add multiple tags to an element, even after its creation.
It should now be possible to search for “tags”, but also for multiple types of elements at the same time (“email” and “todo” for example).
If you’ve got this far, congratulations! You’ve now got youself a basic item list with a search engine all done in HTML, CSS and Javascript (with JQuery).