T - Web Development Seminar

T-WEB-x00

Day 09

jQuery

1.1

{EPITECH.}

Day 09

group size:	1
repository name:	web_seminar_day09_$ACADEMICYEAR
repository rights:	ramassage-tek
language:	javascript

- Your repository must contain the totality of your source files, but no useless files (binary, temp files, obj files,...).

During this first part, we will focus on the basics of jQuery.
For each exercise, you are asked to create a ".js" file.

In the following exercises, the instructions must be executed after the page is fully loaded.
On each exercise you MUST provide the appropriate *index.html* file.

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

EXERCISE 01 — 1PT

Turn in: ex_01/index.html, ex_01/selector.js

At first there was something and then nothing

This JS file must contain a function that selects all paragraphs on the page and makes them disappear.

EXERCISE 02 — 1PT

Turn in: ex_02/index.html, ex_02/selector.js

Try again

This JS file must contain a function that selects an element with the id "test" and makes it disappear.

EXERCISE 03 — 1PT

Turn in: ex_03/index.html, ex_03/selector.js

Third time's the charm

This JS file must contain a function that selects an element with class "test" and makes it disappear.

EXERCISE 04 — 1PT

Turn in: ex_04/index.html, ex_04/selector.js

Half and half

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.

EXERCISE 05 — 2PTS

Turn in: ex_05/index.html, ex_05/selector.js

He was the first of his kind and then he was not

This JS file must contain a function that selects the first element of the first unordered list and makes it disappear.

EXERCISE 06 — 1PT

Turn in: ex_06/index.html, ex_06/event.js

Make them aware of your power

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.

EXERCISE 07 — 2PTS

Turn in: ex_07/index.html, ex_07/event.js

Make them all pale and vanish as you pronounce the words

This JS file must contain a function that assigns an "on" event to all paragraphs of the page.
3 actions must be assigned to this event:

1. when the mouse enters a paragraph, the background color of the paragraph must become light gray,
   2.when the mouse leaves the paragraph, the background color must revert to white,
   3.when you click on the paragraph, its text must disappear.

EXERCISE 08 — 1PT

Turn in: ex_08/index.html, ex_08/event.js

As you wave your hand it's here, then it's not, then again it's here

This JS file must contain a function that assigns a "click" event on the "button" element of the page.
The action of the event must make the element with id "menu" appear or disappear.

EXERCISE 9 — 2PTS

Turn in: ex_09/index.html, ex_09/animate.js

One shall levitate as you change the scenery

This JS file must contain a function that assigns a "click" event on the element with class "platypus."
The action of the event must move the element with class "platypus" 150 pixels to the right, 200 pixels to the bottom and set the background color to green.

Exercise 10 — 2pts

Turn in: ex_10/index.html, ex_10/callback.js

Amber alert

This JS file must contain a function that selects the element with class "test" in the page and makes it disappear.
In addition, an alert must be sent after the disappearance of the element with the message "The paragraph is now hidden."

Exercise 11 — 2pts

Turn in: ex_11/index.html, ex_11/append.js

Fill the void

Add an 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 you just created as parameter.
This function must add a div after this element, that contains the value of the element passed as parameter.

Exercise 12 — 1pt

Turn in: ex_12/index.html, ex_12/insert.js

In-between

This JS file must contain a function that inserts the sentence "Wow, I precede the image!" before the first image and "Hey, I come in last" right after.

EXERCISE 13 — 1PT

Turn in: ex_13/index.html, ex_13/removed.js

Outcasts

This JS file must contain a function that removes from the page all paragraphs with classes "test" and "platypus."

You must call on a single function.

EXERCISE 14 — 2PTS

Turn in: ex_14/main.js

I appoint you "blue class"!

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.

EXERCISE 15 — 3PTS

Turn in: ex_15/

Create an input allowing someone to enter a text.
When the text is considered valid, add it to a bulleted list displayed under the input.

EXERCISE 16 — 3PTS

Turn in: ex_16/

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.

It should now be possible to precise one of this 3 types (note, email, todo) when validating the input.

You're free on the way you implement this and you can add more style to your CSS.
However if an input of type "email" is created, the element displayed with the content must possess the class CSS "email".
It must work the same way for the 2 other types.

You may need to add some different check to validate the inputs depending on the type.

EXERCISE 17 — 3PTS

Turn in: ex_17/

Implement a search features among the created elements.

As of now you only need to handle a search by types (you will implement a search by words in the following exercise).

Searching for "email", only the elements being "email" must stay displayed.
The others must not be deleted and must be shown again when the search is over.

Once again, you're free concerning the implementation

.

Exercise 18 — 4pts

Turn in: ex_18/

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".

Once again, there should be no element destroyed, and it should be possible to switch from a search to another without the need to reset everything.
For example, if I selected "todo" and "rick" in my search but now decided I want all the types I should not need to erase everything and re-do my selection.

Exercise 19 — 2pts

Turn in: ex_19/

Let's now upgrade your input.
Make it possible to add tags to an element.
The tags must also be displayed.
It should be possible to add multiple tags to an element, even after its creation.

Exercise 20 — 5pts

Turn in: ex_20/

Let's upgrade our search engine a little more.
It should now be possible to search for "tags", but also for two types of elements at the same time ("email" and "todo" for example).

Moreover, add the possibility to search for multiples tags at the same time.
It should also be possible to precise if the tags must all be included or not.

For example, if I have

- a "todo" element with the following tags: "work", "tired", "dailyLife".
- an "email" element with the following tags: "work", "boss".
- a "note" element with the followings tags: "boss", "money".

If I search for tags "work", the "todo" and "email" must be displayed.
If I now search for "work" and "boss" (both being included), only the "email" must be displayed.
If I search for "dailyLife" or "money" (one or the other), the "todo" and "note" must be displayed.

We do not ask you to handle cases like: ("work" && "boss") || ("boss" && "money"). Neither all the twisted cases you may imagine.
However if you do handle those cases, you may be awarded bonus points.

Bonus

Here is a little idea for an interesting bonus.
Add a bit of PHP for the backend, and make it possible to save all the elements created with all the fields in a database by calling PHP functions.
It would now be possible to leave the page and come again to see all the elements previously saved be displayed again.