Managing Data on the World-Wide-Web

Assignment 1

HTML, CSS & JavaScript

In this assignment you are required to build a homepage using HTML5, JavaScript and CSS. You should use CSS for defining the visual style of the homepage and you should not rely on styles of the HTML5 tags. Note that this assignment may require self-learning of HTML5, CSS and JavaScript. That is, the material that was shown in the lectures and the tutorials may not suffice for this assignment.

Assignment is 5% Takef. Submission date: 05/11/2015, 23:55

TA in change: Yoav Nahshon. Email: 236369cs@gmail.com

Each student should create his/her own homepage

(Note that differently from future assignments, this assignment is not in pairs)
Assignment Description

In this assignment you are required to create a homepage using your Technion account. It is recommended to use W3Schools tutorials to fill in the gaps on HTML5, CSS and JavaScript. Instructions on how to create a personal homepage on the T2, TX and WWW servers can be found in the following links: T2/TX accounts, CS accounts (if your email is @cs).

You are required to generate two copies of your homepage and present them using two different CSS style sheets. The HTML5 pages should comprise as little style instructions as possible. Style should be handled in a separate CSS document. To evaluate the quality of your page design, make sure the following tasks can be accomplished without changing the HTML5 document.

- If your homepage has several different lists, is it possible to change the style of one list without changing the style of the other?
- Is it easy and straightforward to change a common design property (e.g. the font style)?

Another design goal should be to avoid duplication. For instance, elements that represent similar entities should use a similar style.

The homepage should include (at least) the following content:

- Personal details: name, email.
- A list of 3 links to HTML5 tutorials, and 2 links to CSS3 tutorials.
- A link to another HTML5 page which has the exact same content as your homepage, yet, uses a different CSS (this is the copy, which by including this link, has a link to itself). By having these two copies, we would like to examine the separation between content & style. Grading will consider the separation between content and style and the style-difference between the two copies.
- Your planned partner for the rest of the course assignments – name, email and link to her/his homepage. In case you don't have a partner yet, please mention it.
- A graph representation of at least 3 facts about you that are related in some way.
  - For example: if you use a programming language A at your workplace at company B, that is located in city C, that its Meir is D, then a corresponding graph is:

  ![Graph Example]

  - You can use vis.js or other JavaScript visualization library to create the graph.
- Logging area for read events (explained below).
- Paragraph elements for testing (explained below).
Creating a Read Event

Background

Automatic detection of users' feedback is very important for evaluating and improving web sites. Specifically, detecting which parts of a web page were focused by the user are a common measure for satisfaction as shown in several studies (e.g., this paper won the "best student paper award" in SIGIR 2014). Orthodox methods for detecting user feedback, use mouse tracking (consider this paper for example).

In this assignment you will implement a basic method for logging mouse movements using HTML Events.

As described in W3Schools’ JavaScript tutorial, HTML events are "things" that happen to HTML elements. For example, the event onclick is triggered when the user clicks on an element (e.g. a button). One of the most powerful features of JavaScript is its events handlers which allow us, web developers, to add custom behavior to these events. Furthermore, JavaScript enables us to create our own events; these are called custom events.

In this exercise you will create and handle two new custom events, readenter and readleave, in order to implement the above-mentioned feature.

Requirements

You are required to create the two following custom events:

- readenter – the readenter event should be triggered, for an HTML element, when the mouse cursor is located on that element for a duration that is longer than that set by the user (see notes below). In that case, we say that this element is currently being read.
- readleave - the readleave event is triggered for an HTML element, if the element is being read and the mouse cursor moves out of the element. In that case, we say that this element has stopped being read.

Notes:

- Two readenter events should not be triggered consecutively; that is, a readleave event should always occur between two consecutive readenter events.
- For simplicity, the readenter and readleave events should only be triggered on paragraph elements (<p>). Note that the Firefox browser does not allow nested paragraphs, therefore you can ignore them in your implementation.
- For logging purposes (see below), each paragraph in your homepage should have a unique id.
- Add a way to set the duration of the time interval from the moment the mouse cursor enters the paragraph element to the moment the readenter event is triggered. The default should be 2 seconds, and the minimum should be 0 (i.e.
the event is triggered immediately as the mouse move onto the element). Suggestion: use a range slider (search for “range” in this link).

- To ease the assignment, we allow the readenter event not to occur as long as the mouse cursor is not moving.

Behavior

When the user reads an element (i.e., the readenter event has been triggered), a red border should appear around that element. When the user stops reading (i.e., the readleave event has been triggered), the red border surrounding that element should disappear, and a new log entry should be created (explained next).

Logging

Create an area in your homepage where log entries, created by readleave events, would be displayed. A log entry has the form:

    id: <id>, elapsed time: <elapsed reading time in seconds>

For example, the following line indicates that the paragraph with the id “p-01” was being read for 4.522 seconds:

    id: p-01, elapsed time: 4.522

Finally, add a button to clear the log.

Notes:

- The reading begins when the mouse cursor enters the element.
- Each new log entry should be appended to the end of the current list of entries.

Paragraphs for Testing

For a quick validation of your implementation, please add to your document the elements provided in the file "paragraphsForTesting.txt" under the assignment section in the course website.
Remarks

- Using external JavaScript Libraries - The goal of this task is for you to experience with JavaScript. Therefore, you are not allowed to use any JavaScript library in this task (with the exception of a visualization library, as mentioned above).
- For details on how to create custom events, you can check this short tutorial.
- Your work will be tested using Firefox (the latest version).
- Your HTML5 & CSS files should be valid HTML5 and CSS pages.
  - In order to verify the validity of your page, you may use an HTML validator add-on for Mozilla Firefox - more details in the miscellaneous page.
- Recommended text editors for web development are Brackets and Sublime Text.
- A student that already has a homepage in his/her Technion account is asked to create in this page a link titled "236369 Assignment 1" to the pages of the assignment.
- The homepage language can be either Hebrew or English.

Submission

Submission is via the webcourse electronic submission feature. Create a text file named hw1.txt with 3 lines (this text file will be parsed using a script - make sure there are no mistakes):

- Your email address at the Technion (@t2/cs/tx). We will use this email address to define user names for later use
- Your student number
- The URL of your homepage (if you already have a homepage write the URL of the page you created for the assignment)

Example: hw1.txt

Good Luck!