

CA5213 – Full Stack Laboratory

Lab worksheet #1 - Simple exercise on JavaScript

Dr. M. Deivamani

(Due: 10:59am Tuesday, April 11, 2023)

(No Grading)

The objective of this assignment is to give you a chance to set up and get comfortable with your local JavaScript development environment and some hands-on experience.

Web browsers

The web browser, as the client, is essential to client-side web development. If you use several different browsers on a regular basis, focus on the one you use most often when developing web pages when answering the following questions.

1. What is your preferred browser (provide the name, platform, and version number)?
2. Why is this your preferred browser?
3. What things do you like most about this browser?
4. What things do you like least about this browser?
5. What, if any, extensions/plugin-ins/add-ons do you have installed for this browser and what functions do they provide?

Browser environment

Although it is standardized via EcmaScript, JavaScript as implemented in browsers can sometimes display differences.

1. Which browser are you using to view (or print) this page? Provide the version number and platform.

Data types

JavaScript has six primitive data types (including the new symbol data type introduced in EcmaScript 6) and one composite data type. You can use the `typeof` operator to determine the data type of any expression. Below are several values and the results of passing each as an operand of the `typeof` operator.

Examine the contents of the list closely and use it when answering the questions that follow.

- a. `typeof -.3428E-7` is "number"
- b. `typeof 0xBADFEED` is "number"
- c. `typeof BADFEED` is "undefined"
- d. `typeof "BADFEED"` is "string"
- e. `typeof '0xBADFEED'` is "string"
- f. `typeof `0xBADFEED`` is "string"
- g. `typeof true` is "boolean"
- h. `typeof True` is "undefined"
- i. `typeof TRUE` is "undefined"
- j. `typeof window` is "object"
- k. `typeof Window` is "function"
- l. `typeof WINDOW` is "undefined"

1. Explain the results for (a).
2. Explain the results for (b).
3. Why does (c) produce a different result from (b) and (d)?
4. What do the results of (d), (e) and (f) tell you about string literals in JavaScript?
5. What do the results of (g) through (o) tell you about capitalization in JavaScript?
6. Explain the difference in results for (j), (k), and (l).

Practical problems

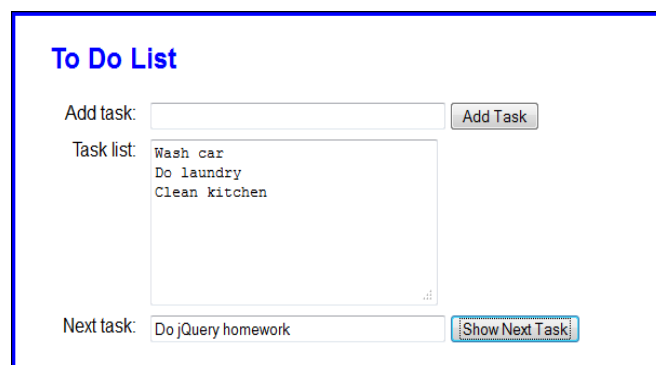
1. Write and test a function named **isPhone** that will take a single parameter and return true if that argument represents a phone number and false otherwise. This function should check that the entire value passed as an argument is a phone number with nothing preceding or following it. It should accept phone numbers in the following formats:
 - XXXXXXXXXX ,
 - XXX-XXX-XXXX ,
 - XXX XXX XXXX ,
 - (XXX) XXX-XXXX , and
 - (XXX)XXX-XXXX
2. Write and test a function named **containsPhone** that will take a single parameter and return true if that argument contains a phone number and false otherwise. This function should ignore anything that may precede or follow the phone number in the passed string value. It should accept phone numbers in the same formats as the previous step.

3. Write and test a function named **extractColorNumber** that will take a single parameter and return the first valid CSS color specifier it finds in that value as a string converted and normalized to the format **#HHHHHHH** (with any letters converted to uppercase). It should identify and extract CSS color specifiers in the following formats:

- **#HHHHHHH**,
- **#HHH**,
- **rgb(D, D, D)**, and
- **rgb(P%, P%, P%)**

where H is any valid hexadecimal digit, D is any decimal value between 0 and 255 (inclusive), P is any decimal value between 0 and 100 (inclusive), and the single spaces after the commas are optional. If the passed parameter contains no color specifier at all, or if the first thing that might be interpreted as a color specifier within the string is not a valid color specifier, this function should return a value of null.

4. Create a simple calculator which performs three more calculations other than the basic arithmetic calculations. You must use an external JavaScript file. Name your files **calculator.html** and **calculator.js**.
5. Create a **"To-do list application"** as shown below. Through this application you should be able to add the task. Once you add the task to be get updated in the task list and you click on the next task button the task must be removed from the list and to be displayed in the next task. You must use an external JavaScript file. Name your files **todolist.html** and **task.js**.



The screenshot shows a web application titled "To Do List" in blue text. Below the title, there is a section labeled "Add task:" followed by a text input field and an "Add Task" button. Below this is a section labeled "Task list:" followed by a text area containing the tasks "Wash car", "Do laundry", and "Clean kitchen". At the bottom, there is a section labeled "Next task:" followed by a text input field containing "Do jQuery homework" and a "Show Next Task" button.