THE UNIVERSITY OF WESTERN AUSTRALIA

School of Computer Science and Software Engineering

CITS3403 Agile Web Development — Lab04

Exercise 4: Javascript and DOM

This lab continues to build core javascript skills.

You may find it useful to refer to an on-line JavaScript reference and tutorials: such as

- MDN Web Docs on JavaScript
- the W3Schools JavaScript Tutorial.

W3Schools

Work through the W3Schools Javascript tutorial, focussing on the basic set, then the functions set, and then the DOM.

Creating a Text-based Calculator

Create a new web page, called `calculator.html` , from your template page.

1. Add JavaScript to pop up a dialogue box that asks the user to enter a formula that they wish to calculate. Write code that *parses* this formula, performs the calculation, and pops up an alert with the answer. You may assume for now that the formula contains at most three operations.
   You will need to use the regular expression methods to parse, or break down, the string into its components, and type conversion to extract the numbers. When doing the parsing, remember to respect the *precedence* of the arithmetic operators. For example, if the user enters:
   ```
   15 + 4 x 3
   ```
   it should give the answer 27, not 57.
   You should not assume any particular convention for whitespace. So for example
   ```
   15+4 x  3
   ```
   should give the same result.

2. Repeat the above task for arbitrary length formulae using an array as a *Stack* data structure (that is, using the push and pop methods). You will need three stacks: one (the input stack) to hold the parts of the input as it is parsed, one (the preorder stack) to hold the components of the formula in preorder format, and one (the answer stack) to accumulate the answer. (You may assume for now only natural numbers and no parentheses are used in the formula.)
   The algorithm proceeds as follows:

   - **Step 1**
     Pop the top item off the input stack. If it is a number push it onto the preorder stack. Otherwise find the first occurrence of the lowest precedence operator in the string, slice the string and push its two arguments onto the input stack (second argument first), and push the operator onto the preorder stack. Continue until the input stack is empty.

     Example:
     ```
     input stack         preorder stack

     1x2+3x4+5           empty
     3x4+5, 1x2          +
     3x4+5, 2, 1         +, x
     3x4+5, 2            +, x, 1
     3x4+5               +, x, 1, 2
     5, 3x4              +, x, 1, 2, +
     ...
     empty               +, x, 1, 2, +, x, 3, 4, 5
     ```
   - **Step 2**
     Pop the top item off the preorder stack. If it is a number, push it onto the answer stack. If it is a binary operator, pop two items off the answer stack, perform the operation, and push the answer onto the answer stack. Continue until the preorder stack is empty.

UNIT COORDINATOR

Dr Tim French
Phone: 64882794

LAB FACILITATOR

Mr Michael Stewart, Mr Tom Smoker

CONSULTATION TIME

Where: CSSE rm 2.14
Time: Wednesday 11-12pm
No appointment needed.

NEWS:

- [27 Feb 2019] This unit currently has 105 CITS3403 students and 29 CITS5505 students enrolled.
- [27 Feb 2019] Labs start in week 2. As Monday marsch 4th is a public holiday the first lab will be in Wednesday. Students in the monday lab can do the labs the following week.

Example:

```
preorder stack                answer stack

+, x, 1, 2, +, x, 3, 4, 5     empty
+, x, 1, 2, +, x, 3, 4        5
+, x, 1, 2, +, x, 3           5, 4
+, x, 1, 2, +, x              5, 4, 3
+, x, 1, 2, +                 5, 12
+, x, 1, 2                    17
...
empty                         19
```

Write code that prints the stacks out in the browser window at each step, as per the examples above, and then prints (in bold) the final answer.

3. *Challenge:* Extend your code so that it also correctly treats parentheses and negative numbers.
4.

Generating Pages Dynamically using the DOM

The aim of this exercise is to create the components of the page by directly constructing the DOM tree.

1. Read The DOM and JavaScript.

2. In the lectures a number of properties and methods that are useful for traversing the DOM tree were discussed, such as `previousSibling`, `nextSibling`, `firstChild`, `lastChild` and `parentNode`. A number of methods for manipulating the tree were also discussed, such as `insertBefore`, `appendChild`, `removeChild` and `replaceChild`. Some other useful properties and methods include the `body` property and the methods `createElement` and `createTextNode`.
   An on-line reference for these (and other) properties and methods can be found in the Gecko DOM Reference at the Mozilla Developer Center. Look up each of the above methods in the Gecko DOM Reference. Note that some of the methods belong to `document` and some belong to `element`.

3. Create a template page, with all the components that you had previously (a banner, a menu, and a footer), with JavaScript.

4. Assume you are writing a system that allows users to prototype web pages using your template and print them off. Previously you created pages by manually copying the template to a new file and adding the content (and title) with an editor. This of course requires access to the filesystem so is not an option for end users.
   Write methods that pop up dialogue boxes asking the user to enter the page title, the heading (of the content section of the page), a paragraph of text, and the URL of a picture. The page should then be displayed with the correct title, and the heading, paragraph of text, and image in the content section of the page.

Optional for fun!Using the Canvas

There is a good online tutorial for using the canvas from W3Schools. For the second exercise of this lab work through the tutorial, and make sure that you understand all features used.