University of Haute-Alsace
ENSISA

Web Programming

TD1: JavaScript Calculator

TD Objectives

Discover JavaScript and its mechanisms to capture events and dynamically modify the content of a web page.

Code functionalities in an environment that is already partially defined, focusing on logic rather than design.

TD Description

A skeleton HTML/CSS file is provided in the archive `squelette_calculatrice.zip`, available on Moodle.

The user interface is made up of buttons that can be clicked, but currently, they do not function.

Instructions

Study the architecture of the calculator's web page, including the various CSS and JavaScript imports. Normally, a quick overview of the structure will be given at the beginning of the session.

Read the assignment carefully; it contains a lot of useful information, and every word is significant!

Use the documentation provided. When working with a new library, developers spend considerable time understanding its functionality. **RTFM (Read The Fine Manual)!**

Write clean code with meaningful variable and function/method names. Add useful comments to your code so you can understand it even after a few days.

Follow coding style guides. It is recommended to use guidelines from Mozilla [1] or the Google JavaScript Style Guide [2].

Simple is beautiful: Avoid overly complex solutions; simplicity is more effective and less error-prone.

Think before coding. Take some time to sketch out or write down your ideas. This reflection time will save you a lot of debugging effort later.

---

1. The Basics

You have an empty skeleton called **tp1.js** in which you will put your JavaScript code. For web development, there are many IDEs, I let you use your favorite, if you don't have one, you can use the following:

Visual Studio Code [3]

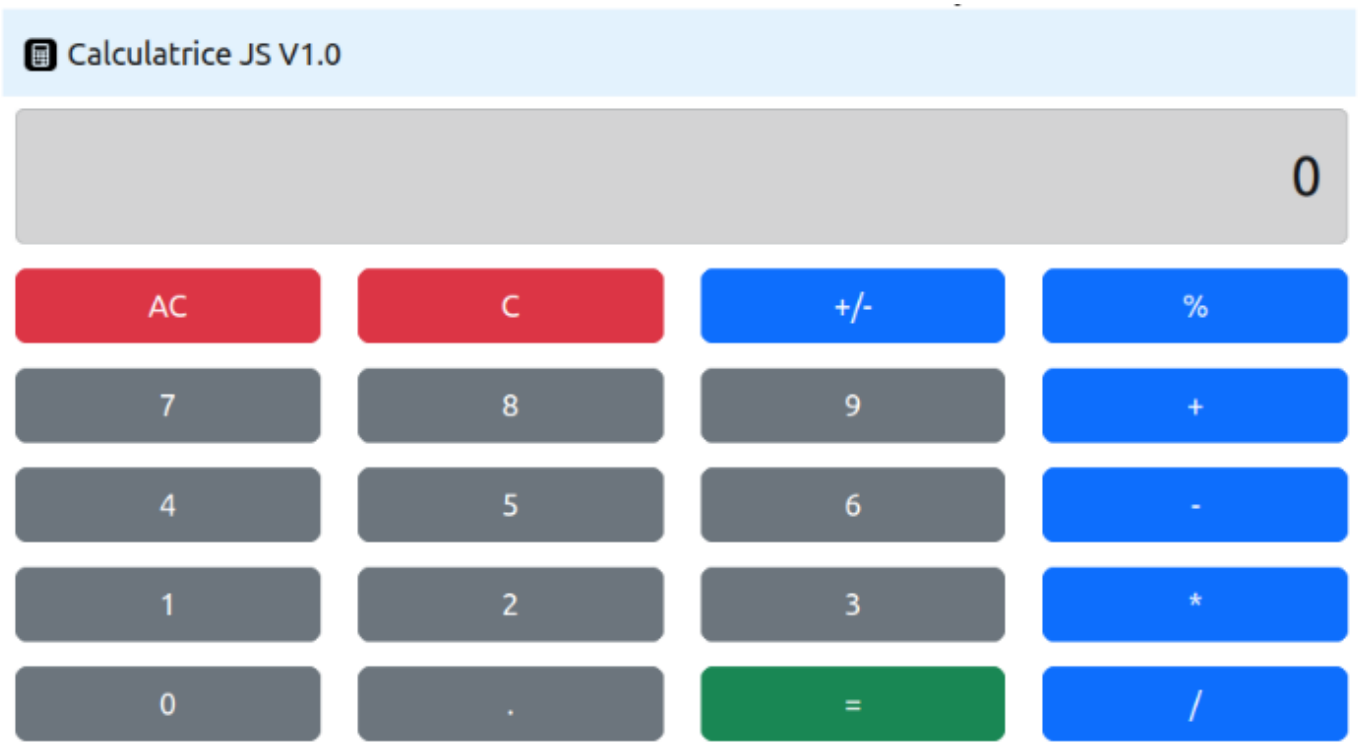
Brackets [4]

WebStorm [5] (likely too advanced for this course)

[1] Mozilla : `https://developer.mozilla.org/en-US/docs/MDN/Writing_guidelines/Writing_style_guide/Code_style_guide/JavaScript`
[2] Google JavaScript Styleguide : https://google.github.io/styleguide/jsguide.html
[3] https ://code.visualstudio.com/
[4] https ://brackets.io/
[5] https ://www.jetbrains.com/fr-fr/webstorm/

2. Let's manage the numbers

Right now, if you open *index.html* , you get this nice Bootstrap interface.



You can click on the different buttons but absolutely nothing happens. We will therefore retrieve in Javascript the click event made by the user on the buttons and then process them to modify the display and perform the different calculations.

2.1 Let's get the clicks on the numbers

To add a Listener to an object on the web page, you must use the Javascript *addEventListener* [6] method which takes as an argument the type of click and the function to call as a *callback*. You must also specify the object to *listen* to, in this case, a button. To do this, you can use the *getElementById*[7] function.

Create the 10 listeners for the 10 buttons containing the numbers and make them call a write function that you will create and which contains an alert[8] displaying a message of your choice.

2.2 Let's treat each number differently

You can see that for now, whatever the button clicked, the result is the same. Indeed, the callback call does not allow you to specify parameters and therefore to differentiate the processing. To discriminate the different numbers, we will replace the call to write with an anonymous function [9] that will call write with a parameter that will be the number of the button clicked. You will display this number with alert.

2.3 Let's display the numbers in the calculator display bar

Now we want to display the entered numbers in the calculator display bar. To do this, we will need to modify the HTML content of the display object. To do this, use the innerHTML[10] property of the HTML elements in the write function.

[6] https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener
[7] https://developer.mozilla.org/fr/docs/Web/API/Document/getElementById
[8] https://developer.mozilla.org/fr/docs/Web/API/Window/alert
[9] https://developer.mozilla.org/en-US/docs/Glossary/IIFE
[10] https://developer.mozilla.org/en-US/docs/Web/API/Element/innerHTML

2.4 Strange display

You notice that it works, but you are adding your digits after the initial 0. Find a mechanism to detect when you are in this initial situation and thus remove the 0 before adding your clicked digits. Hint: use a boolean variable [11].

2.5 And the point in all this?

Add the management of the "." button so that when we click a point is added to the display. The management of the point adds two difficulties:

— There can only be one point in a number. You can use the indexOf [12] function to check that there is not already a point in the display.

— When we are on the display of the initial 0, we must not replace it with "." but with "0.".

2.6 Clear the display

Add handling for the C button that clears the display and resets it. Create a dedicated function **clearDisplay**.

3 Let's Manage the Operations

We will now manage the different operations, focusing first on -, *, and /, leaving + aside for now.

The typical usage pattern is to enter an operator after typing a number and trigger the calculation when choosing a second operator or pressing the = key. This will require storing information in memory, notably the current number and the last chosen operation. When a calculation is performed, its result is displayed in the calculator's display bar.

For implementation, I recommend creating two functions:

1. **manageOperator**: Called when operator buttons are clicked.
2. **compute**: Performs the calculations when necessary.

3.1 Managing Subtraction, Multiplication, and Division

Implement subtraction, multiplication, and division.

3.2 Managing Addition

Implement addition. Is there an issue? What is it? How can it be resolved?

3.3 Managing Unary Operations

Now implement the unary operations:

1. +/-: Changes the sign of the displayed number (which can either be a number entered by the user or the result of a previous calculation).

2. %: Converts the displayed number into a percentage.

3.4 Implement the AC Button

Implement the AC button, which fully resets the state of the calculator.

4 Bonus

Already finished? In addition to the buttons, now manage the user's ability to enter numbers and operators by typing keyboard keys. Hint: KeyboardEvent [13] is your friend

[11] `https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Statements/var`

[12] `https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/indexOf`

[13] `https://developer.mozilla.org/en-US/docs/Web/API/KeyboardEvent/key`