Lab Exercise 11

Dr. Sarvar Abdullaev
s.abdullaev@inha.uz

April 16, 2020

The purpose of this lab is to learn basics of JavaScript by building a simple calculator. We'll focus on the JavaScript you need to write—how to think about building the calculator, how to write the code, and eventually, how to clean up your code.

By the end of the article, you should get a calculator that functions exactly like an iPhone calculator (without the +/- and percentage functionalities).

1 Clone Project and Review HTML

Complete steps below in corresponding order:

1. Clone the project and open `index.html` in browser. The calculator consist of two parts: the display and the keys.

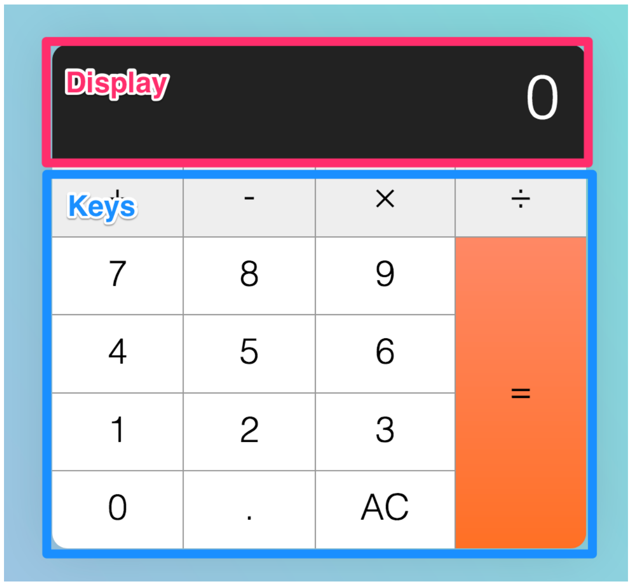


Figure 1: Calculator

2. Open `index.html` in Sublime Text. To help us identify operator, decimal, clear, and equal keys, we're going to supply a `data-action` attribute that describes what they do. Just ensure that this attribute is written for each action button.

```
<div class="calculator__keys">
  <button class="key--operator" data-action="add">+</button>
  <button class="key--operator" data-action="subtract">-</button
  <button class="key--operator" data-action="multiply">&times;</button>
  <button class="key--operator" data-action="divide">\</button
  <button>7</button>
  <button>8</button>
  <button>9</button>
  <button>4</button>
  <button>5</button>
  <button>6</button>
  <button>1</button>
  <button>2</button>
  <button>3</button>
  <button>0</button>
  <button data-action="decimal">.</button>
  <button data-action="clear">AC</button>
  <button class="key--equal" data-action="calculate">=</button>
</div>
```

3. Create `app.js` file in the same folder and reference it in `index.html`

2 Handling Button Clicks

Complete steps below in corresponding order:

1. We will write all of our Javascript code inside `app.js`. Inside `app.js` handle `window.onload` event with an anonymous function.

   ```
   window.onload=function(){


   }
   ```

   All of our javascript code will be written inside this anonymous function handler.

2. Let's start implementing buttons of the calculator. Five things can happen when a person gets hold of a calculator. They can hit:

   (a) a number key (0–9)

   (b) an operator key (+, -, ×, ÷)

   (c) the decimal key

   (d) the equals key

   (e) the clear key

   The first steps to building this calculator are to be able to (1) listen for all keypresses and (2) determine the type of key that is pressed. In this case, we can use an *event delegation pattern* to listen, since keys are all children of `.calculator__keys`. In this pattern, we use `addEventListener('event_name', handler)` to the parent object. If the child object is clicked, parent can handle the click event of its child. Event handler can accept a special event argument which contains information about the sender of the event.

   ```
   let calculator = document.querySelector('.calculator')
   let keys = calculator.querySelector('.calculator__keys')
   keys.addEventListener('click', function(e) {
    if (e.target.matches('button')) {
      // Do something
    }
   })
   ```

3. Next, we can use the `data-action` attribute to determine the type of key that is clicked.

   ```
   let key = e.target
   let action = key.dataset.action
   ```

4. If the key does not have a `data-action` attribute, it must be a number key.

```
if (!action) {
  console.log('number key!')
}
```

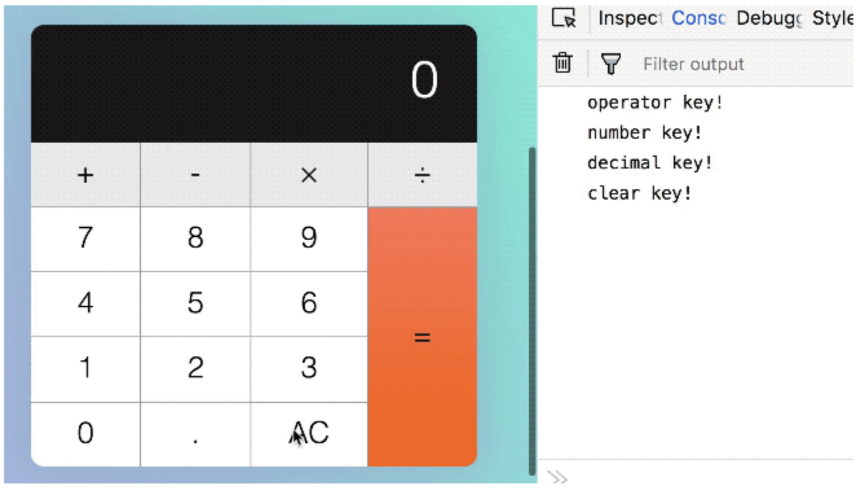
5. If the key has a `data-action` that is either *add*, *subtract*, *multiply* or *divide*, we know the key is an operator.

```
if (action === 'add' ||
  action === 'subtract' ||
  action === 'multiply' ||
  action === 'divide') {  console.log('operator key!') }
```

6. If the key's |data-action—is *decimal*, we know the user clicked on the decimal key. Following the same thought process, if the key's |data-action—is *clear*, we know the user clicked on the clear (the one that says AC) key. If the key's |data-action—is calculate, we know the user clicked on the equals key.

```
if (action === 'decimal') {
  console.log('decimal key!')
}
if (action === 'clear') {
  console.log('clear key!')
}
if (action === 'calculate') {
  console.log('equal key!')
}
```

At this point, you should get a console.log response from every calculator key.



```
operator key!
number key!
decimal key!
clear key!
```

Figure 2: Calculator with button handlers

3 Implementing Buttons

Let's consider what the average person would do when they pick up a calculator. This "what the average person would do" is called the happy path.

Let's call our average person Mary.

When Mary picks up a calculator, she could hit any of these keys:

1. a number key (0–9)
2. an operator key (+, -, ×, ÷)

3. the decimal key
4. the equal key
5. the clear key

It can be overwhelming to consider five types of keys at once, so let's take it step by step.

3.1 When user hits a number key

1. At this point, if the calculator shows 0 (the default number), the target number should replace zero. If the calculator shows a non-zero number, the target number should be appended to the displayed number.
2. Here, we need to know two things:
   (a) The number of the key that was clicked
   (b) The current displayed number

   We can get these two values through the `textContent` property of the clicked key and `.calculator__display` , respectively. Note we can use `const` keyword if variable that we are declaring is a constant.

```
const display = document.querySelector('.calculator__display')
keys.addEventListener('click', e => {
  if (e.target.matches('button')) {
    const key = e.target
    const action = key.dataset.action
    const keyContent = key.textContent
    const displayedNum = display.textContent
    // ...
  }
})
```

3. If the calculator shows 0, we want to replace the calculator's display with the clicked key. We can do so by replacing the display's `textContent` property.

```
if (!action) {
  if (displayedNum === '0') {
    display.textContent = keyContent
  }
}
```

4. If the calculator shows a non-zero number, we want to append the clicked key to the displayed number. To append a number, we concatenate a string.

```
if (!action) {
  if (displayedNum === '0') {
    display.textContent = keyContent
  } else {
    display.textContent = displayedNum + keyContent
  }
}
```

5. At this point, Mary may click either of these keys: A decimal key OR An operator key. Let's say Mary hits the decimal key.