Javascript Project

progate

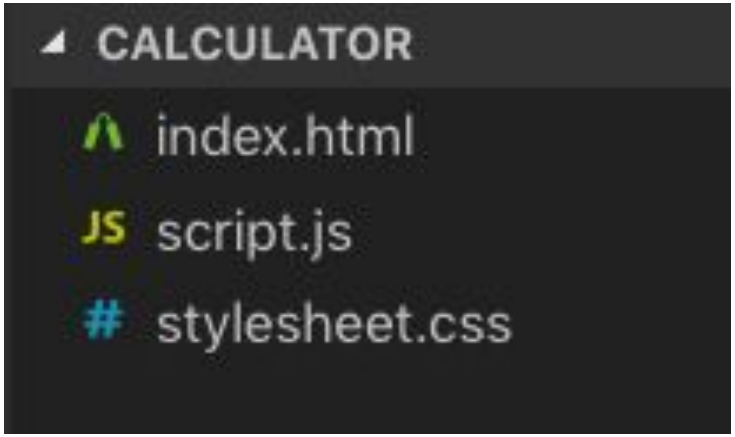
Apa yang akan kita buat

- Kita akan membuat aplikasi kalkulator berbasis web seperti dibawah ini:

Langkah 1:

Tambahkan html code ke index.html

- Buat folder dengan nama “kalkulator”.
- Kita membutuhkan 3 file. File html, file css, dan file JavaScript. Buat file-file ini di folder kalkulator.
- Buka folder tersebut dengan text editor.



Menambahkan struktur dasasr HTML Document

- Tambahkan struktur dasar HTML ke index.html.

- Deklarasi DOCTYPE, <html> tag, <head> tag dan <body> diperlukan.

  rf. https://progate.com/html/study/1/10#/30

Tambahkan konten ke dalam <head> Tag

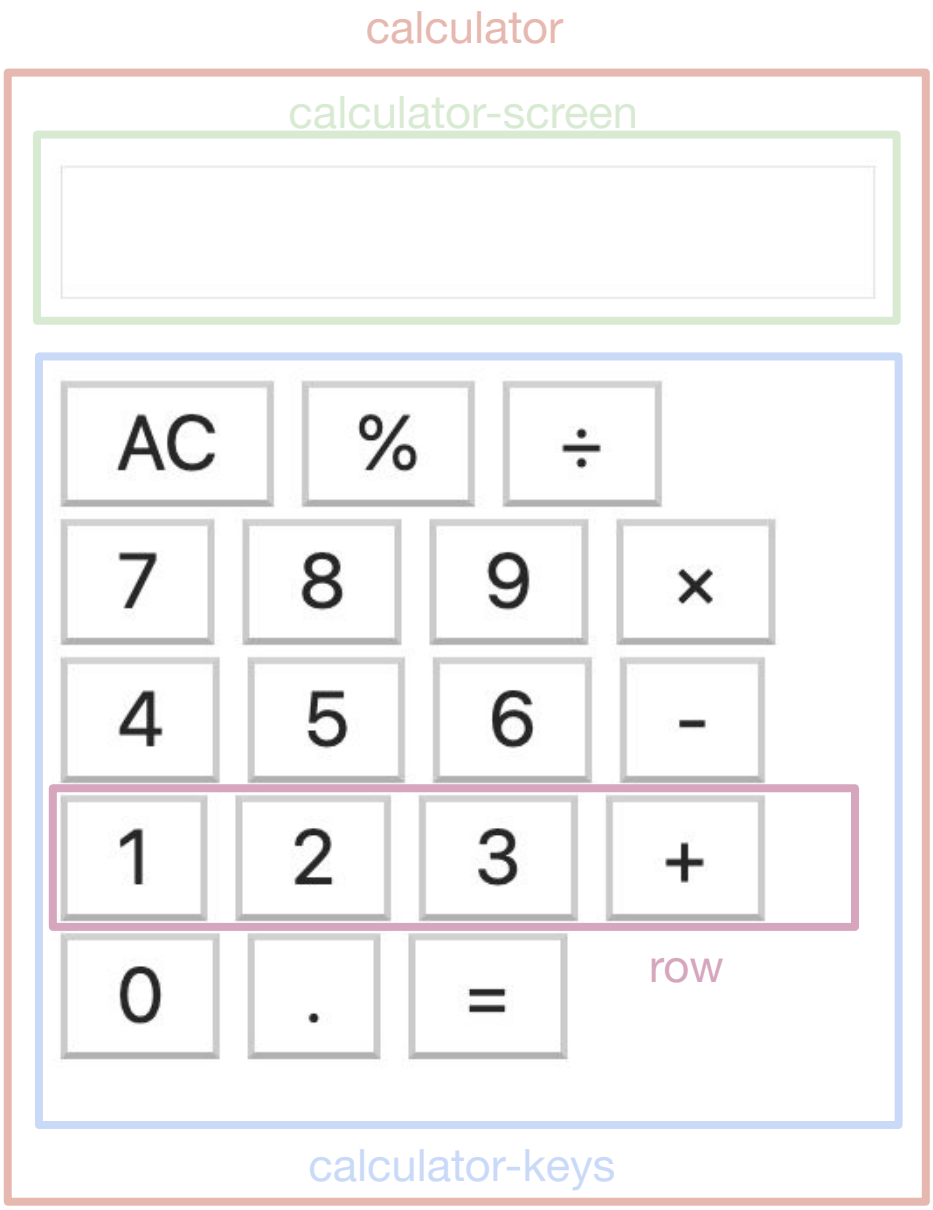
- Tambahkan konten ke dalam <head> tag. Character Encoding, Judul dan link ke file CSS diperlukan.

  rf. https://progate.com/html/study/1/11#/33

index.html

```
1    <!DOCTYPE>
2    <html>
3      <head>
4        <meta charset="utf-8">
5        <title>Calculator</title>
6        <link rel="stylesheet" href="stylesheet.css">
7      </head>
8      <body>
9      </body>
10   </html>
```

Layout Utama aplikasi Kalkulator



Tambahkan tags untuk Layout Utama

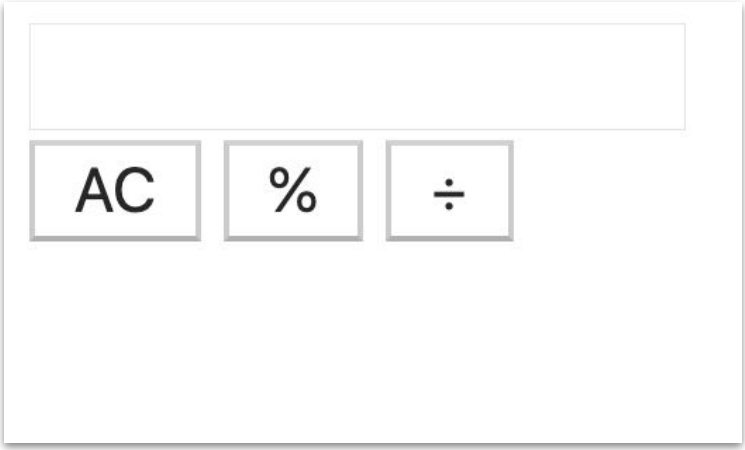
- Tambahkan tag dengan class “calculator”.

- Tambahkan tag <input> dengan class “calculator-screen” dan tag dengan “calculator-keys” di dalam class “calculator”.

```
<!DOCTYPE>
<html>
  <head>
    <meta charset="utf-8">
    <title>Calculator</title>
    <link rel="stylesheet" href="stylesheet.css">
  </head>
  <body>
    <div class="calculator">
      <input type="text" class="calculator-screen" />
      <div class="calculator-keys">
      </div>
    </div>
  </body>
</html>
```

Add HTML for Calculator Keys

- Tambahkan tag dengan class “row” dan 3 tag <button> untuk row pertama di dalam class “row”.




```
<div class="calculator-keys">
    <div class="row">
        <button class="all-clear">AC</button>
        <button class="percentage">%</button>
        <button class="operator">&divide;</button>
    </div>
</div>
```

Tambahkan HTML untuk baris-baris yang lain

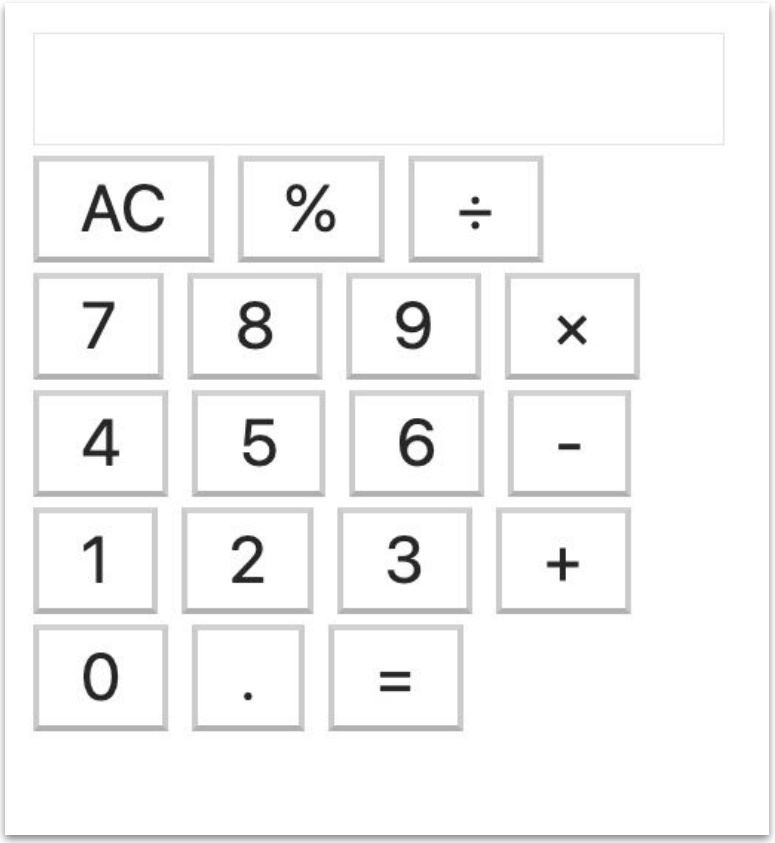
- Tambahkan 4 tag lagi dengan class “row” dan tag <button> di dalamnya.

- Tambahkan nama class “number”, “operator”, “decimal”, “equal-sign”, “all-clear”, “percentage”, dan “zero-btn” ke tag button.

```
<div class="calculator-keys">
    <div class="row">
        <button class="all-clear">AC</button>
        <button class="percentage">%</button>
        <button class="operator">&divide;</button>
    </div>
    <div class="row">
        <button class="number">7</button>
        <button class="number">8</button>
        <button class="number">9</button>
        <button class="operator">&times;</button>
    </div>
    <div class="row">
        <button class="number">4</button>
        <button class="number">5</button>
        <button class="number">6</button>
        <button class="operator">-</button>
    </div>
    <div class="row">
        <button class="number">1</button>
        <button class="number">2</button>
        <button class="number">3</button>
        <button class="operator">+</button>
    </div>
    <div class="row">
        <button class="number zero-btn">0</button>
        <button class="decimal">.</button>
        <button class="equal-sign">=</button>
    </div>
```

Periksa layoutnya

- Jika tampilan app kalkulator Anda seperti di gambar dibawah ini, berarti anda telah mengikuti instruksi dengan baik dan benar!



Tambah atribut nilai ke tag <input>

- Tambahkan atribut nilai dan berikan “0” ke pada atribut tersebut. Atribut nilai ini adalah nilai yang akan ditampilkan di form input.

- Tambahkan “disabled” ke tag input juga. Fungsi Atribut ini untuk melarang melakukan input angka menggunakan keyboard anda. Ini untuk membantu menjaga code anda ringkas dan mudah dimengerti.

```
<div class="calculator">
    <input type="text" class="calculator-screen" value="0" disabled />
    <div class="calculator-keys">
        <div class="row">
```

Tambahkan atribut nilai ke tag <button>

- Tambahkan atribut nilai ke semua tag button yang memiliki class number", "operator" dan "decimal". Ini tidak akan mempengaruhi di tampilan nanti, namun akan kita gunakan nanti di code javascript.

```
<div class="row">
    <button class="all-clear">AC</button>
    <button class="percentage">%</button>
    <button class="operator" value="/">&divide;</button>
</div>
<div class="row">
    <button class="number" value="7">7</button>
    <button class="number" value="8">8</button>
    <button class="number" value="9">9</button>
    <button class="operator" value="*">&times;</button>
</div>
<div class="row">
    <button class="number" value="4">4</button>
    <button class="number" value="5">5</button>
    <button class="number" value="6">6</button>
    <button class="operator" value="-">-</button>
</div>
<div class="row">
    <button class="number" value="1">1</button>
    <button class="number" value="2">2</button>
    <button class="number" value="3">3</button>
    <button class="operator" value="+">+</button>
</div>
<div class="row">
    <button class="number zero-btn" value="0">0</button>
    <button class="decimal" value=".">.</button>
    <button class="equal-sign">=</button>
</div>
```

Langkah 2:

Menambahkan code CSS ke stylesheet.css

Mensejajarkan tampilan Calculator

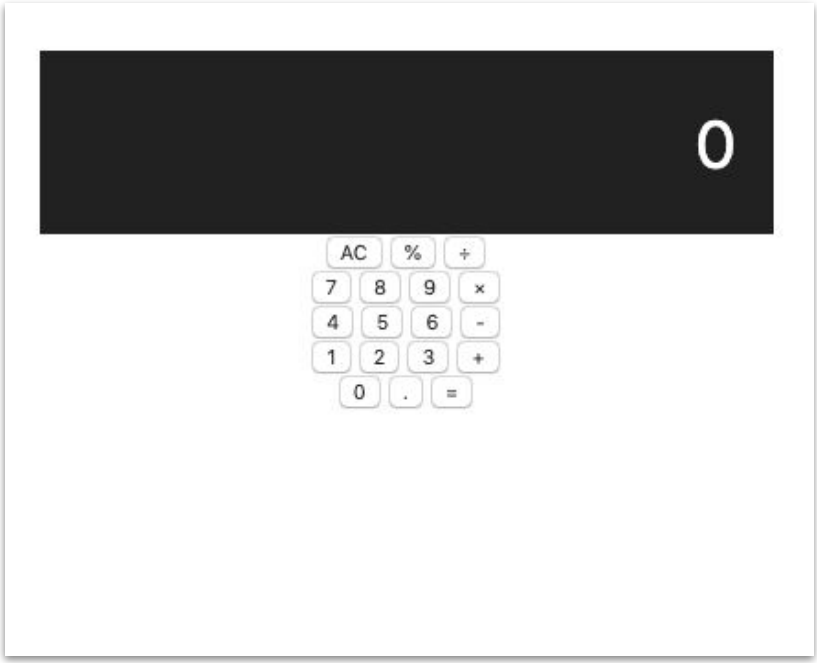
- Tambahkan code CSS ke class calculator untuk mensejajarkan ke tengah

```
# stylesheet.css ▸ ...
1    .calculator {
2      text-align: center;
3      margin: 0 auto;
4      padding-top: 200px;
5    }
6
```

Membuat Tampilan calculator-screen lebih Rapih

- Tambahkan `width: 400px;` ke class “calculator”.

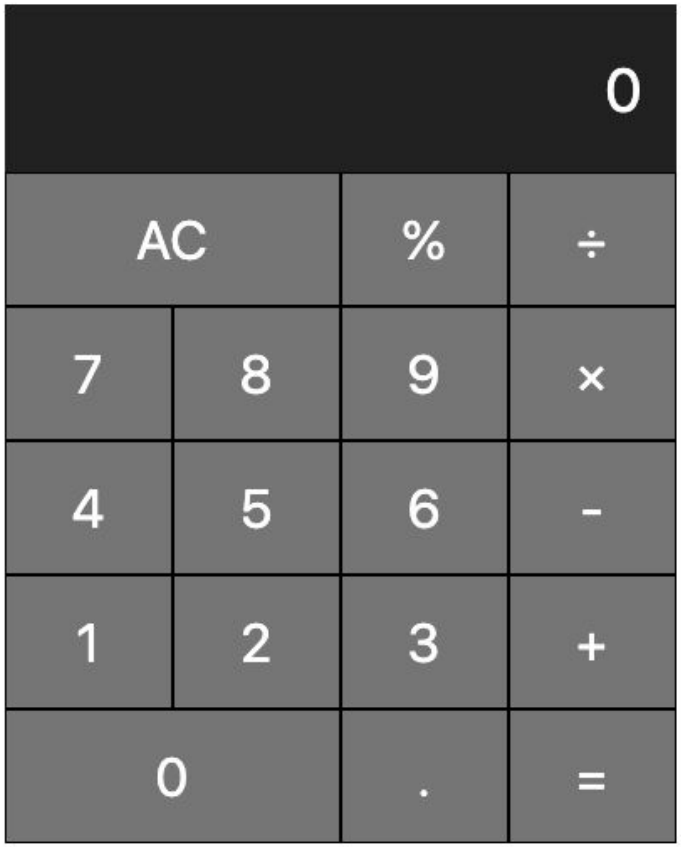
- Tambahkan beberapa settingan style untuk “calculator-screen” seperti contoh yang telah diberikan di gambar sebelah kanan.




```css
.calculator {
  text-align: center;
  margin: 0 auto;
  padding-top: 200px;
  width: 400px;
}

.calculator-screen {
  width: 100%;
  height: 100px;
  background-color: #252525;
  color: #fff;
  text-align: right;
  font-size: 36px;
  border: none;
  padding: 0 20px;
  box-sizing: border-box;
}
```

Modifikasi layout tombol calculator




```css
.calculator-keys{
  width: 100%;
}

.row{
  display: flex;
}

button{
  height: 80px;
  background-color: gray;
  border: 1px solid black;
  font-size: 32px;
  color: #fff;
  width: 25%;
  outline: none;
}

.all-clear, .zero-btn{
  width: 50%
}
```

Merubah warna ikon Operator dan sama-dengan menjadi Oranye

- Ketik ulang background-color untuk tombol dengan class “operator” dan class “equal-sign class”.


```
.operator, .equal-sign {
  background-color: orange;
}
```

Merubah Background Color Saat kursor mouse di atas tombol

- Rubahlah background-color tombol saat kursor mouse berada diatas (hover) tombol tersebut.


```
button:hover {
  background-color: darkgray;
}

.operator:hover, .equal-sign:hover {
  background-color: darkorange;
}
```

Langkah 3:

Membuat aplikasi dapat menerima input dari tombol-tombol dan menampilkannya di layar display aplikasi kalkulator.

Menyambungkan File HTML dengan File JavaScript

- Anda perlu menyambungkan File HTML dengan Fila JavaScript sebelum memulai menambahkan code Javascript (dapat dilakukan dengan menggunakan tag <script>).

- Anda dapat menambahkan tag <script> tag dimana saja didalam file HTML, namun lumrahnya di tempatkan di ujung tag body karena HTML akan di-load sebelum file Javascript.

```
    </div>
    <script type="text/javascript" src="script.js"></script>
</body>
```

Cara mengaktifkan data input dan tampilkan hasil

- Mari aktifkan data input dan menampilkan hasil input. Tahapannya sebagai berikut:

  Saat tombol di klik:

  1. Mendeteksi aksi klik tombol
  2. Mengambil angka tombol yang di klik.
  3. Jalankan function untuk menampilkan angka di layar display

Mendeteksi adanya klik sebuah tombol

- Anda perlu men-setting “click event” untuk mendeteksi klik sebuah tombol, dan ini dapat dilakukan dengan menggunakan “event listener”. (walau tidak dibahas di materi Progate, namun tidak terlalu sulit untuk dipelajari)

- Ada 2 langkah untuk men-setting click event:
  1. Mengambil element HTML dari code JavaScript
  2. Tambahkan click event ke element menggunakan “event listener”

Mengambil element HTML dari code Javascript

- Anda dapat mengambil element menggunakan `document.querySelectorAll(“.class name”)` atau `document.querySelector(“.class name”)`

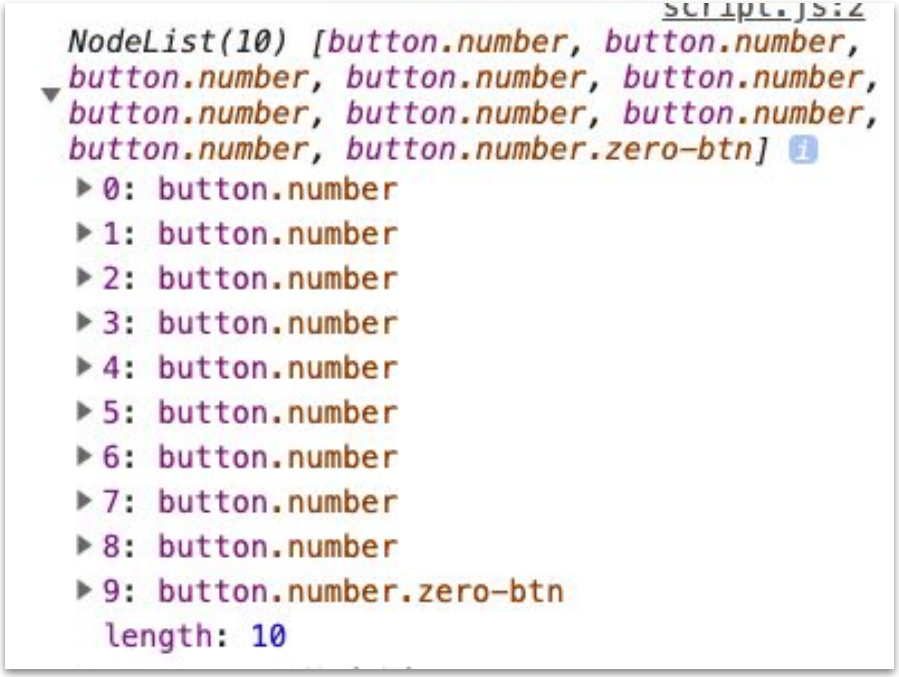
- `querySelectorAll` digunakan saat Anda ingin mendapatkan beberapa element, dan `querySelector` digunakan saat Anda ingin mendapatkan satu jenis element.

- Untuk latihan ini, kita akan menggunakan querySelectorAll karena ada banyak tombol yang memiliki class “numbera’ dan kita mau menampilkan semua.

Mengambil element HTML di code Javascript

- Mari kita ambil element-element <button> menggunakan class “number” dan print-kan didalam console.

```
JS script.js ▸ ...
1    const numbers = document.querySelectorAll(".number")
2    console.log(numbers)
```



Dapatkan masing-masing angka dari constant “numbers”

- Event listener harus ditambahkan ke setiap element, jadi kita harus mengambilnya satu persatu. Anda bisa menggunakan method forEach.

  rf. https://progate.com/es6/study/6/2#/7

```
const numbers = document.querySelectorAll(".number")

numbers.forEach((number) => {
  console.log(number)
})
```

```
                                                    script.js:4
<button class="number" value="7">7</button>
                                                    script.js:4
<button class="number" value="8">8</button>
                                                    script.js:4
<button class="number" value="9">9</button>
                                                    script.js:4
<button class="number" value="4">4</button>
                                                    script.js:4
<button class="number" value="5">5</button>
                                                    script.js:4
<button class="number" value="6">6</button>
                                                    script.js:4
<button class="number" value="1">1</button>
                                                    script.js:4
<button class="number" value="2">2</button>
                                                    script.js:4
<button class="number" value="3">3</button>
                                                    script.js:4
<button class="number zero-btn" value="0">0
</button>
```

Menambah event Click ke setiap element

- Mari kita tambahkan click event ke setiap element menggunakan addEventListener.

- Jika Anda menekan tombol dan teks “number is pressed” ditampilkan di console, berarti code Anda sudah benar.

```
const numbers = document.querySelectorAll(".number")

numbers.forEach((number) => {
    number.addEventListener("click", () => {
        console.log("number is pressed")
    })
})
```

Cara menampilkan angka saat menekan tombol

- Click events sudah ditetapkan ke setiap element <button>, sekarang saatnya untuk menampilkan angka-angka nya dari tombol-tombol tersebut.

- Anda perlu menambahkan argument `event` dan setelah itu Anda bisa mengakses angka menggunakan `event.target.value`. Nilai ini berasal dari atribut nilai yang Anda telah berikan ke tag <button> sebelumnya.

```
const numbers = document.querySelectorAll(".number")

numbers.forEach((number) => {
    number.addEventListener("click", (event) => {
        console.log(event.target.value)
    })
})
```

Definisikan Function untuk Memperbarui Layar Tampilan

- Sekarang, Anda perlu membuat agar layar dapat menampilkan angka yang benar saat mengklik suatu tombol.

- Anda dapat merubah angka yang ditampilkan di layar dengan cara memperbarui atribut nilai dari tag <input>. Mari kita tarik element di code JS dan definisikan function “updateScreen” untuk memperbarui nilai.

```
const calculatorScreen = document.querySelector('.calculator-screen')

const updateScreen = (number) => {
  calculatorScreen.value = number
}
```

Perbarui layar saat tombol angka di tekan

- Jalankan function “updateScreen” menggunakan `event.target.value` sebagai argument saat tombol di klik.

```
const calculatorScreen = document.querySelector('.calculator-screen')

const updateScreen = (number) => {
  calculatorScreen.value = number
}

const numbers = document.querySelectorAll(".number")

numbers.forEach((number) => {
  number.addEventListener("click", (event) => {
    updateScreen(event.target.value)
  })
})
```

Memahami apa yang barusan kita lakukan

- Mungkin terlihat rumit, namun apa yang baru saja Anda lakukan adalah mengambil element-element HTML dan memanipulasinya di code Javascript. Mari kita lihat lagi langkah-langkahnya:

1. Ambil element <button> dengan class “number”.
2. Tambahkan click event ke setiap element dan jalankan function “updateScreen” saat tombol diklik.
3. Perbarui atribut nilai dari element <input> didalam “calculator-screen” dengan menggunakan fucntion “updateScreen”.

Langkah 4:

Menyimpan angka-angka dan operator untuk melakukan kalkulasi.

Definisikan Variable untuk melakukan kalkulasi

- Yang Anda butuhkan untuk melakukan suatu kalkulasi adalah 2 angka dan sebuah operator.

- Definisikan 3 variable, yaitu “prevNumber”, “currentNumber” dan “calculationOperator” untuk menyimpan 2 angka dan and 1 operator. Nilai awal “currentNumber” seharusnya adalah 0.

```
let prevNumber = ''
let calculationOperator = ''
let currentNumber = '0'
```

Cara Variable bekerja

	prevNum	currentNum	operator	displayed
Press 3		3		3
Press +	3		+	3
Press 5	3	5	+	5
Press =		8		8

Memberikan the Number yang di klik ke variable currentNumber

- Mari kita definisikan function “inputNumber” dan jalankan saat angka di klik.

- Didalam function “inputNumber”, berikan angka yang di klik ke currentNumber. Jangan lupa untuk merubah argument “updateScreen” menjadi currentNumber.

```
const inputNumber = (number) => {
  currentNumber = number
}
```

```
numbers.forEach((number) => {
  number.addEventListener("click", (event) => {
    inputNumber(event.target.value)
    updateScreen(currentNumber)
  })
})
```

Cara mengaktifkan peng-input-an lebih dari 2 digit angka

- Anda bisa meng-input suatu angka dan merubah tampilan angka di layar, tapi belum bisa meng-input lebih dari 1 digit angka saat ini.

- Ini dapat diselesaikan menggunakan rangkaian string, karena suatu angka yang di simpan didalam variable “currentNumber” adalah sebuah string.

```
const inputNumber = (number) => {
  currentNumber += number
}
```

Permasalahan saat 0 di pencet terlebih dahulu

- Ada suatu masalah bahwa angka yang ditampilkan dapat diawali dengan angka 0. Mari kita perbaiki dengan menggunakan if statement.

```
const inputNumber = (number) => {
  if (currentNumber === '0') {
    currentNumber = number
  } else {
    currentNumber += number
  }
}
```

Menambah Click event ke operator tombol-tombol

- Mari kita lanjutkan ke bagian operator. Pertama, Anda harus melakukan hal yang sama seperti saat mengonfigurasi angka. Ambil element-element <button> menggunakan class “operator” dan tambahkan click event ke setiap tombol operator.

- Jangan lupa untuk memeriksa jika code anda tidak ada error di console.

```
const operators = document.querySelectorAll(".operator")

operators.forEach((operator) => {
  operator.addEventListener("click", (event) => {
    console.log(event.target.value)
  })
})
```

Definisikan function inputOperator

- Mari kita definisikan function “inputOperator”. Apa yang ingin kita lakukan dengan function tersebut:

1. Memberikan nilai yang tersimpan di currentNumber ke prevNumber.
2. Berikan operator ke calculationOperator sebagai suatu argument.
3. Kosongkan currentNumber.

```
const inputOperator = (operator) => {
  prevNumber = currentNumber
  calculationOperator = operator
  currentNumber = ''
}
```

Jalankan function inputOperator saat Operator di Klik

- Jalankan function “inputOperator” saat operator di klik.

- Kita tidak akan memanggil pdateScreen saat ini karena kita ingin menampilkan angka sebelumnya.

```
const operators = document.querySelectorAll(".operator")

operators.forEach((operator) => {
    operator.addEventListener("click", (event) => {
        inputOperator(event.target.value)
    })
})
```

Langkah 5:

Mengaktifkan fungsi kalkulasi ke aplikasi calculatornya.

Apa yang akan kita lakukan di Langkah ini

- Karena sekarang kita sudah memiliki 2 angka dan 1 operator yang tersimpan di dalam ke-tiga variable tersebut, mari kita coba apakah memungkinkan untuk melakukan fungsi kalkulasi di langkah yang ini.

- Definisikan function “calculate” dan jalanka saat tombol sama-dengan (=) di klik.

Tambahkan click event ke tombol sama-dengan (=)

- Tambahkan click event ke suatu element. Untuk kali ini, hanya ada 1 tombol untuk sama-dengan (=), jadi kita akan menggunakan querySelector, bukan querySelectorAll.

```
const equalSign = document.querySelector('.equal-sign')

equalSign.addEventListener('click', () => {
  console.log('equal button is pressed')
})
```

Definisikan function Calculation

- Mari kita definisikan function Calculate. Terdapat 4 pola dalam kalkulasi yaitu penambahan, pengurangan, perkalian, dan pembagian. Tambahkan 4 kasus ini tergantung terhadap nilai yang tersimpan di variable calculationOperator.

```
const calculate = () => {
  let result = ''
  switch(calculationOperator) {
    case "+":
      result = prevNumber + currentNumber
      break
    case "-":
      result = prevNumber - currentNumber
      break
    case "*":
      result = prevNumber * currentNumber
      break
    case "/":
      result = prevNumber / currentNumber
      break
    default:
      break
  }
}
```

Simpan hasil Kalkulasi ke currentNumber

- Hasil kalkukasi seharusnya ditampilkan di laya. Mari perbarui variable currentNumber dengan hasilnya tersebut.
- Selain itu, nilai dari calculationOperator seharusnya kosong.

```
const calculate = () => {
  let result = ''
  switch(calculationOperator) {
    case '+':
      result = prevNumber + currentNumber
      break
    case '-':
      result = prevNumber - currentNumber
      break
    case '*':
      result = prevNumber * currentNumber
      break
    case '/':
      result = prevNumber / currentNumber
      break
    default:
      return
  }
  currentNumber = result
  calculationOperator = ''
}
```

Jalankan Function Calculate saat tombol sama-dengna (=) di Klik

- Jalankan Function Calculate saat tombol sama-dengna (=) di Klik, dan perbarui layarnya.

```
const equalSign = document.querySelector('.equal-sign')

equalSign.addEventListener('click', () => {
  calculate()
  updateScreen(currentNumber)
})
```

Selesaikan suatu masalah di Penambahan

- Fungsi Penambahan tidak bekerja karena angka-angka tersimpan sebagai string. Untuk menghindari perangkaian string, saat ini Anda butuh mengganti angka-angka tersebut menjadi integer. Hal ini dapat dilakukan dengan menggunakan method parseInt.

```
const calculate = () =>{
  let result = ''
  switch(calculationOperator) {
    case '+':
      result = parseInt(prevNumber) + parseInt(currentNumber)
      break
```

Langkah ke 6:

Membuat tombol AC berjalan dengan lancar

Apa yang akan Anda lakukan di langkah ini

- Langkah berikutnya, kita akan memastikan tombol AC bekerja dengan baik.

- Apa yang akan kita lakukan adalah menambahkan click event ke suatu element, dan jalankan function suatu fungsi untuk menghapus isi layar tampilan saat tombol AC di klik.

Ambil Element Button dan tambahkan Click Event

- Mari kita ambil element <button> menggunakan class “all-clear” dan tambahkan click event,

```
const clearBtn = document.querySelector('.all-clear')

clearBtn.addEventListener('click', () => {
  console.log('AC button is pressed')
})
```

Definisikan dan jalankan Function clearAll

- Definisikan function“clearAll” dan tetapkan angka 0 ke currentNumber, lalu kosongkan 2 variable yang lain.

- Jalankan function saat tombol AC di klik dan perbarui layar tampilan setelah itu.

```
const clearAll = () => {
  prevNumber = ''
  calculationOperator = ''
  currentNumber = '0'
}
```

```
const clearBtn = document.querySelector('.all-clear')

clearBtn.addEventListener('click', () => {
  clearAll()
  updateScreen(currentNumber)
})
```

Langkah 7:

Membuat aplikasi dapat mengkalkulasi angka desimal.

Apa yang akan Anda lakukan di Langkah ini

- Kita akan memungkinkan untuk mengkalkulasi angka desimal.
- Cara melakukannya sama seperti sebelumnya.

Tambahkan Click Event ke element <button>

- Mari kita ambil element <button> yang memiliki class “decimal” class dan tambahkan click event ke element tersebut

```
const decimal = document.querySelector('.decimal')

decimal.addEventListener('click', (event) => {
  console.log(event.target.value)
})
```

Definisikan dan jalankan Function inputDecimal

- Definisikan function “inputDecimal” dan tambahkan titik desimal ke currentNumber.

- Jalankan function tersebut saat tombol titik desimal di klik dan perbarui layar tampilan setelah itu.

```
inputDecimal = (dot) => {
  currentNumber += dot
}
```

```
const decimal = document.querySelector('.decimal')

decimal.addEventListener('click', (event) => {
  inputDecimal(event.target.value)
  updateScreen(currentNumber)
})
```

Masalah dengan Fungsi Penambahan

- Mari hitung “1.2 + 1.3”. Hasil yang diharapkan adalah.5, tapi hasil yang ditampilkan adalah 2.

- Ini terjadi karena method `parseInt` method, dimana ia tidak bisa mengkalkulasi angka desimal.

- Bagaimana cara perbaikinya? Ganti method `parseInt` dengan method `parseFloat`.

```
case '+':
    result = parseFloat(prevNumber) + parseFloat(currentNumber)
    break
```

Mencegah peng-inputan titik desimal berulang kali

- Saat ini, aplikasi calculator Anda dapat meng-input titik desimal berkali-kali.

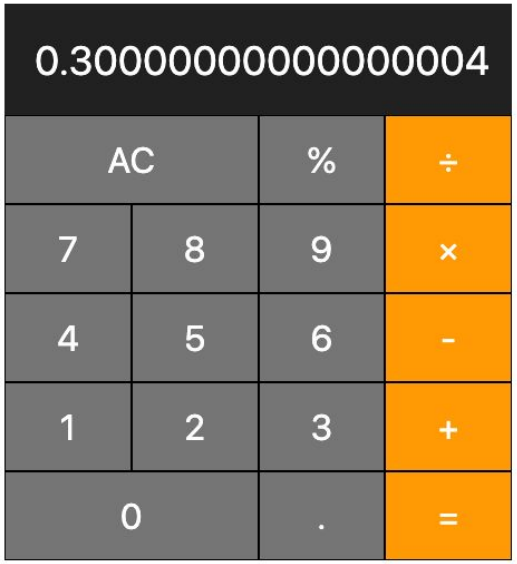
- Jika currentNumber sudah termasuk titik desimal, selesaikan function inputDecimal sebelum titik desimal ditambahkan ke currentNumber.

```
inputDecimal = (dot) => {
  if(currentNumber.includes('.')) {
    return
  }
  currentNumber += dot
}
```

Hasil Kalkulasi yang tidak Akurat

- Mari kita coba menghitung “0.1 + 0.2”. Hasilnya akan sebuah angka yang sangat aneh seperti gambar dibawah ini.
- Ini memang masalah yang ada dengan javascript saat ini, jadi untuk saat ini, tidak perlu hiraukan masalah ini.

https://www.w3schools.com/js/js_numbers.asp



Step 8:

Menyelesaikan suatu masalah

Apa yang akan Anda lakukan di Langkah ini

- Fitur-fitur utama dari aplikasi calculator sudah selesai, namun masih ada 1 masalah.

- In kerap terjadi saat kunci operator mengklik 2 kali secara beruntun.
- Jika Anda menemukan masalah lain, cobalah terlebih dahulu untuk membetulkan secara mandiri.

Selesaikan permasalahannya terlebih dahulu

- Masalah ini muncul karena variable-variable diperbarui setiap tombol operator (manapun) di klik.

- Yang Anda harus lakukan adalah perbarui variable prevNumber hanya saat tombol operator diklik terlebih dahulu.

```
const inputOperator = (operator) => {
  if (calculationOperator === '') {
    prevNumber = currentNumber
  }
  calculationOperator = operator
  currentNumber = '0'
}
```

KITA BERHASIIL!!!!!