# PERTEMUAN III – JAVASCRIPT (BAGIAN 1)

Pratikan akan mempelajari tentang dasar – dasar Javascript. Tujuan setelah mempelajari BAB ini adalah :

1. Mampu memahami dasar – dasar Javascript.
2. Mampu memahami penggunaan Javascript.

## Pengertian

Javascript adalah bahasa pemrograman yang paling populer didunia. Javascript dapat mengubah isi dari HTML, dapat mengubah atribut HTML, dapat mengubah styles (CSS), dapat pula mengenali data input. Javascript juga berguna untuk desaign antar muka web yang responsif.

## Struktur

Penulisan Javascript diawali dengan <script> dan diakhiri </script>.

|  |
| --- |
| <script>  let a = “…”;  function name(){  statement;  }  </script> |

## Penempatan Javascript

1. Embeded Script (Internal Javascript)

JavaScript dapat dituliskan pada berkas HTML dengan menggunakan elemen <script>. Elemen <script> dapat diletakkan didalam elemen <head> atau <body>

1. Section Head

Javascript dapat ditempatkan didalam head. Jika telah menempatkan di head disarankan agar tetap melanjutkan di head dan tidak membuat javascript di body lagi.

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <script>  function myFunction() {  document.getElementById("demo").innerHTML = "Paragraph changed.";  }  </script>  </head>  <body>  <h1>My Web Page</h1>  <p id="demo">A Paragraph</p>  <button type="button" onclick="myFunction()">Try it</button>  </body>  </html> |

1. Section Body

Javascript juga dapat ditempatkan di body. Tidak ada perbedaan dalam penempatan antara head dan body, tetapi lebih baik javascript di body ditempatkan paling bawah sebelum tag penutup body agar tidak memperlambat proses loading halaman web.

|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <h2>My First Web Page</h2>  <p>My First Paragraph.</p>  <p id="demo"></p>  <script>  document.getElementById("demo").innerHTML = "Halooooo";  </script>  </body>  </html> |

1. External Javascript

Javascript juga dapat dibuat dalam file tersendiri, bedanya hanya pada proses pemanggilan. File javascript menggunakan ekstensi .js. Untuk menggunakan Javascript external, tidak perlu menulis tag <script> didalam file javascript, langsung saja ke fungsi yang akan diimplementasikan ke dalam halaman web.

Keuntungan menggunakan Javascript External

* 1. Dapat membagi code antara html dan javascript
  2. Dapat membuat code html dan javascript mudah dibaca
  3. Dapat membuat file javascript diload lebih cepat.

|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <script src="myScript.js"></script>  </body>  </html> |

## Dasar – Dasar Javascript

### Output

Javascript Ouput merupakan keluaran dari hasil yang di cetak didalam Javascript.

|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <h2>My First Web Page</h2>  <p>My First Paragraph.</p>  <p id="demo"></p>  <script>  document.getElementById("demo").innerHTML = 5 + 6;  </script>  </body>  </html> |

### Comment

Digunakan untuk memberi catatan pada source code.

Contoh single comment :

|  |
| --- |
| <script>  // Ini adalah Komentar Tunggal  </script> |

Contoh multi comment :

|  |
| --- |
| <script>  /\*  Ini adalah contoh Komentar  Yang mencakup banyak baris  \*/  </script> |

### Variabel

JavaScript memiliki tiga cara untuk mendeklarasikan variabel: "var", "let", dan "const". Meskipun ketiganya dapat digunakan untuk mendeklarasikan variabel, mereka memiliki perbedaan dalam cara mereka bekerja dan ruang lingkup (scope) variabel tersebut.

1. var

Sebelum adanya ECMAScript 2015 (ES6), keyword "var" adalah satu-satunya cara untuk mendeklarasikan variabel pada JavaScript. Variabel yang dideklarasikan dengan "var" memiliki scope fungsional, yang berarti variabel tersebut dapat diakses di seluruh fungsi tempat variabel tersebut dideklarasikan.

Namun, variabel "var" juga memiliki kelemahan, yaitu variabel "var" dapat diakses bahkan sebelum dideklarasikan. Hal ini disebabkan oleh perilaku hoisting pada JavaScript, di mana deklarasi variabel akan dipindahkan ke atas scope fungsional tempat variabel dideklarasikan, sebelum kode dieksekusi.

*Source code*

|  |
| --- |
| <script>  console.log(nim); // undefined  var nim = 2018112;  console.log(nim); // 2018112  </script> |

Penjelasan :

Saat kita menggunakan var pada variable “nim” apabila kita memanggil variable tersebut sebelum di deklarasikan maka variable tersebut akan bernilai undefined. Hal ini disebabkan oleh adanya hoisting pada javascript sehingga pada source code diatas secara tidak langsung deklarasi variabel "nim" dipindahkan ke atas scope fungsional di mana variabel tersebut dideklarasikan, sehingga ketika console.log pertama dijalankan, variabel "nim" sudah dideklarasikan, meskipun belum diberi nilai, sehingga menghasilkan nilai "undefined". Kemudian variabel "nim" diberi nilai 2018112 dan kemudian nilai tersebut dicetak di console.log kedua.

1. let

Diperkenalkan pada ES6, keyword "let" merupakan cara baru untuk mendeklarasikan variabel pada JavaScript. Variabel yang dideklarasikan dengan "let" memiliki scope blok, yang berarti variabel tersebut hanya dapat diakses di dalam blok kode tempat variabel dideklarasikan.

Selain itu, variabel "let" tidak mendukung hoisting, yang berarti variabel harus dideklarasikan sebelum digunakan. Dalam hal ini, "let" lebih aman digunakan karena variabel hanya dapat diakses di dalam blok tempat variabel dideklarasikan dan tidak dapat diakses sebelum dideklarasikan secara eksplisit.

*Source code*

|  |
| --- |
| <script>  console.log(nim);  let nim = 2018112;  console.log(nim);  </script> |

Penjelasan :

Apabila kita menggunakan let untuk mendeklarasikan variable nim, lalu kita coba melakukan hal yang sama seperti contoh pada penggunaan var maka akan menghasil error ReferenceError. Ini terjadi karena variabel yang dideklarasikan dengan "let" tidak mengalami hoisting seperti yang terjadi pada variabel yang dideklarasikan dengan "var". Oleh karena itu, variabel "nim" harus dideklarasikan terlebih dahulu sebelum dipanggil.

1. const

Keyword "const" juga diperkenalkan pada ES6 dan digunakan untuk mendeklarasikan variabel dengan nilai tetap (konstan). Variabel yang dideklarasikan dengan "const" memiliki scope blok yang sama dengan "let", yang berarti variabel hanya dapat diakses di dalam blok tempat variabel dideklarasikan.

Namun, variabel "const" memiliki perbedaan penting dengan "let", yaitu nilai variabel "const" tidak dapat diubah setelah dideklarasikan. Hal ini menjadikan "const" sangat berguna untuk variabel yang memang memiliki nilai tetap dan tidak perlu diubah dalam program.

*Source code*

|  |
| --- |
| <script>  const nim = 2018112;  nim = 1018112;  </script> |

Penjelasan :

Saat menggunakan const untuk mendeklarasikan variable nim, lalu kita coba untuk mengubah isi dari variable nim tersebut maka kita akan mendapatkan error TypeError. Hal ini disebabkan karena variabel nim dideklarasikan dengan const yang artinya nilainya tidak dapat diubah setelah dideklarasikan.

### Tipe Data

Tipe data dalam Javascript tidak perlu diidentifikasi hanya saja penulisan yang harus benar.

|  |
| --- |
| let x = 5; //untuk number  let y = “hallo”; //untuk string  let z = true; //untuk boolean |

### Operator

Operator Aritmatika Javascript.

|  |  |
| --- | --- |
| **Operator** | **Deskripsi** |
| + | Tambah |
| - | Kurang |
| \* | Kali |
| / | Bagi |
| % | Modulus |
| ++ | Increment |
| -- | Decrement |

Operator Asigment / Penugasan

|  |  |  |
| --- | --- | --- |
| **Operator** | **Example** | **Same As** |
| = | x = y | x = y |
| += | x += y | x = x + y |
| -= | x -= y | x = x - y |
| \*= | x \*= y | x = x \* y |
| /= | x /= y | x = x / y |
| %= | x %= y | x = x % y |

Operator Relasi

|  |  |
| --- | --- |
| **Operator** | **Description** |
| == | Membandingkan kedua nilai **apakah sama** |
| === | Membandingkan kedua **apakah identik** |
| != | Membandingkan kedua nilai **apakah tidak sama** |
| !== | Membandingkan kedua nilai **apakah tidak identik** |
| > | Lebih besar dari |
| < | Lebih kecil dari |
| >= | Lebih besar sama dengan |
| <= | Lebih kecil sama dengan |
| ? | Ternary operator |

Operator Logika

|  |  |
| --- | --- |
| **Operator** | **Deskripsi** |
| && | Menghasilkan true apabila semua kondisi terpenuhi |
| || | Menghasilkan true apabila salah satu kondisi terpenuhi |
| ! | Digunakan untuk membalikan suatu kondisi |

### Function

1. Function Declaration

Fungsi dideklarasikan dengan keyword function dan nama fungsinya

|  |
| --- |
| function name(parameter1, parameter2, parameter3) {  code to be executed  //return x //if return value needed  } |

1. Function Expression

Function expression bisa disimpan kedalam variabel, tidak perlu menuliskan nama fungsinya. Expression berarti mengembalikan nilai, maka dari itu function expression selalu mengembalikan nilai

|  |
| --- |
| const namaVar = function (parameter1, parameter2) {  code to be executed  return x  } |

1. Arrow Function

Arrow function sama seperti function lainnya secara perilaku, tetapi berbeda secara penulisannya

|  |
| --- |
| const sayName = name => {  console.log(`Halo Nama saya ${name}`);  }  sayName("Silva");  const sayHello = () => {  console.log(`Halo Selamat Pagi`);  }  sayHello(); |

Contoh :

|  |
| --- |
| <body>  <p id="demo"></p>  <script>  var x = myFunction(4, 3);  document.getElementById("demo").innerHTML = "Hasil dari Fungsi : " + x;  function myFunction(a, b) {  return a \* b;  }  </script>  </body> |

### Condition

Condition / kondisi digunakan jika membutuhkan kondisi dalam suatu program.

1. Struktur IF

|  |
| --- |
| if (condition) {  block of code to be executed if the condition is true  } |

1. Struktur IF - ELSE

|  |
| --- |
| if (condition) {  block of code to be executed if the condition is true  } else {  block of code to be executed if the condition is false  } |

1. Struktur IF - ELSE IF - ELSE

|  |
| --- |
| if (condition1) {  block of code to be executed if condition1 is true  } else if (condition2) {  block of code to be executed if the condition1 is false and condition2 is true  } else {  block of code to be executed if the condition1 is false and condition2 is false  } |

1. Contoh

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8" />  <meta http-equiv="X-UA-Compatible" content="IE=edge" />  <meta name="viewport" content="width=device-width, initial-scale=1.0" />  <title>Praktikum Web 2022</title>  </head>  <body>  <center>  <form action="" name="myForm">  <label for="wn">Masukkan Warga Negara</label> <br />  <input type="text" name="wn" id="wn" /> <br /><br />  <button type="button" name="simpan" onclick="greeting()">Simpan</button>  </form>  <p id="cetak"></p>  </center>  <script>  function greeting() {  let greeting = "Halo";  let myForm = document.myForm;  let wn = myForm.wn.value;  if (wn == "Indonesia") {  greeting = "Selamat Pagi";  } else if (wn == "Inggris") {  greeting = "Good Morning";  } else if (wn == "Korea") {  greeting = "Annyeong Haseyo";  } else {  greeting;  }  document.getElementById("cetak").innerHTML = greeting;  }  </script>  </body>  </html> |

### Switch

Struktur :

|  |
| --- |
| switch(expression) {     case n:         code block        break;     case n:         code block        break;     default:         default code block } |

Contoh :

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8" />  <meta http-equiv="X-UA-Compatible" content="IE=edge" />  <meta name="viewport" content="width=device-width, initial-scale=1.0" />  <title>Praktikum Web 2022</title>  </head>  <body>  <center>  <form action="" name="myForm">  <label for="nama">Masukkan Nama</label> <br />  <input type="text" name="nama" id="nama" /> <br /><br />  <button type="button" name="simpan" onclick="hari()">Simpan</button>  </form>  <p id="cetak"></p>  </center>  <script>  function hari() {  let myForm = document.myForm;  let nama = myForm.nama.value;  let hariIni = new Date().getDay();  let hari = "";  switch (hariIni) {  case 0:  hari = "Minggu";  break;  case 1:  hari = "Senin";  break;  case 2:  hari = "Selasa";  break;  case 3:  hari = "Rabu";  break;  case 4:  hari = "Kamis";  break;  case 5:  hari = "Jumat";  break;  case 6:  hari = "Sabtu";  break;  }  document.getElementById(  "cetak"  ).innerHTML = `Halo ${nama} hari ini hari ${hari}`;  }  </script>  </body>  </html> |

### Loop

Template :

|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <p id="demo"></p>  <script>  let text = "";  let i = 0;  // statement  document.getElementById("demo").innerHTML = text;  </script>  </body>  </html> |

Macam – Macam Loop :

1. For Loop

|  |
| --- |
| for (i; i < 5; i++) {  text += "The number is " + i + "<br>";  } |

1. While

|  |
| --- |
| while (i < 10) {  text += "The number is " + i + "<br>";  i++;  } |

1. DO / WHILE

|  |
| --- |
| do {  text += "The number is " + i + "<br>";  i++;  } while (i <= 10); |

### Array

1. Array 1 Dimensi

Struktur :

|  |
| --- |
| let points = new Array(40, 100, 1, 5, 25, 10); |

Atau,

|  |
| --- |
| let points = [40, 100, 1, 5, 25, 10]; |

Contoh :

|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <p id="demo"></p>  <script>  let food = ["Gado-Gado", "Tahu Telur", "Nasi Kuning"];  document.getElementById("demo").innerHTML = food;  </script>  </body>  </html> |

1. Array 2 Dimensi

|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <p id="demo"></p>  <script>  let food = [  ["Gado-Gado", "Tahu Telur", "Nasi Kuning"],  ["Teh Manis","Sprite","Joshua"]  ];  document.getElementById("demo").innerHTML = food;  </script>  </body>  </html> |

1. Memproses tiap data array

|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <p id="demo"></p>  <script>  let food = ["Gado-Gado", "Tahu Telur", "Nasi Kuning"];  food.forEach(function(item, index) {  document.getElementById("demo").innerHTML += index + ". " + item + "<br>";  });  </script>  </body>  </html> |

### Objects

Struktur :

|  |
| --- |
| let person = {name:"Dimas", age:"21", gender:"male"}; |

Contoh :

|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <p id="demo"></p>  <script>  let person = {name:"Dimas", age:"21", gender:"male"};  document.getElementById("demo").innerHTML = "The Person name is " + person.name;  </script>  </body>  </html> |

Memproses tiap data pada object

|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <p id="demo"></p>  <script>  let person = {name:"Dimas", age:"21", gender:"male"};  let text = "";  Object.keys(person).forEach(function(key) {  text += key + ": " + person[key] + "<br>";  });  document.getElementById("demo").innerHTML = text;  </script>  </body>  </html> |

## Document Object Model

Document Object Model atau disingkat DOM adalah sebuah programming interface untuk web dokumen. DOM merepresentasikan struktur dan konten dari dokumen web dalam bentuk node dan object supaya kita dapat mengubah struktur, style, dan konten dari dokumen tersebut. Kode JavaScript yang dijalankan di browser environment akan memiliki akses terhadap objek document dan window untuk memanipulasi DOM. Dengan demikian, dokumen web dapat terhubung atau berinteraksi dengan scripts atau bahasa pemrograman.

1. Mencari elemen HTML
2. Mencari elemen HTML berdasarkan id

|  |
| --- |
| document.getElementById(‘id’) |

1. Mencari elemen HTML berdasarkan class

|  |
| --- |
| document.getElementByClassName(‘class’) |

1. Mencari elemen HTML berdasarkan nama tag

|  |
| --- |
| document.getElementByTagName(‘tag’) |

1. Mencari 1 elemen HTML yang pertama kali ditemukan

|  |
| --- |
| document.querySelector(‘.class’) // untuk class  document.querySelector(‘#id’) // untuk id  document.querySelector(‘tag’) // untuk tag |

1. Mencari beberapa elemen HTML sekaligus

|  |
| --- |
| document.querySelectorAll(‘.class’) // untuk class  document.querySelectorAll(‘#id’) // untuk id  document.querySelectorAll(‘tag’) // untuk tag |

1. Memanipulasi elemen HTML

Setelah kita mendapatkan elemen html dengan menggunakan perintah perintah pada nomor 1, selanjutnya kita bisa melakukan manipulasi terhadap elemen tersebut.

1. Mengganti isi dari sebuah elemen HTML

|  |
| --- |
| <body>  <p id="demo"></p>  <script>  document.getElementById("demo").innerHTML = "Teks diubah melalui script";  </script>  </body> |

1. Mengganti style CSS

|  |
| --- |
| <body>  <div style="width: 100px; height: 100px;  background-color: tomato" id="targetObject"></div>  <input type="text" placeholder="masukan warna"  style="margin: 10px 0" id="inputWarna" />  <br />  <button onClick="ubah()">Ubah Warna</button>  <script>  function ubah() {  let warna = document.getElementById("inputWarna").value;  let target = document.getElementById("targetObject");  target.style.backgroundColor = warna;  }  </script>  </body> |

1. Mengubah isi suatu attribute

|  |
| --- |
| <body>  <input type="checkbox" id="check" />  <label for="check"> Checkbox </label>  <div isActive=""></div>  <p>Isi atribut isActive : <span id="kondisi"></span></p>  <script>  let check = document.getElementById("check");  let kondisi = document.getElementById("kondisi");  let div = document.querySelector("div");  check.addEventListener("change", function () {  if (check.checked) {  div.setAttribute("isActive", "true");  kondisi.innerHTML = div.getAttribute("isActive");  } else {  div.setAttribute("isActive", "false");  kondisi.innerHTML = div.getAttribute("isActive");  }  });  </script>  </body> |

1. Menambah elemen HTML

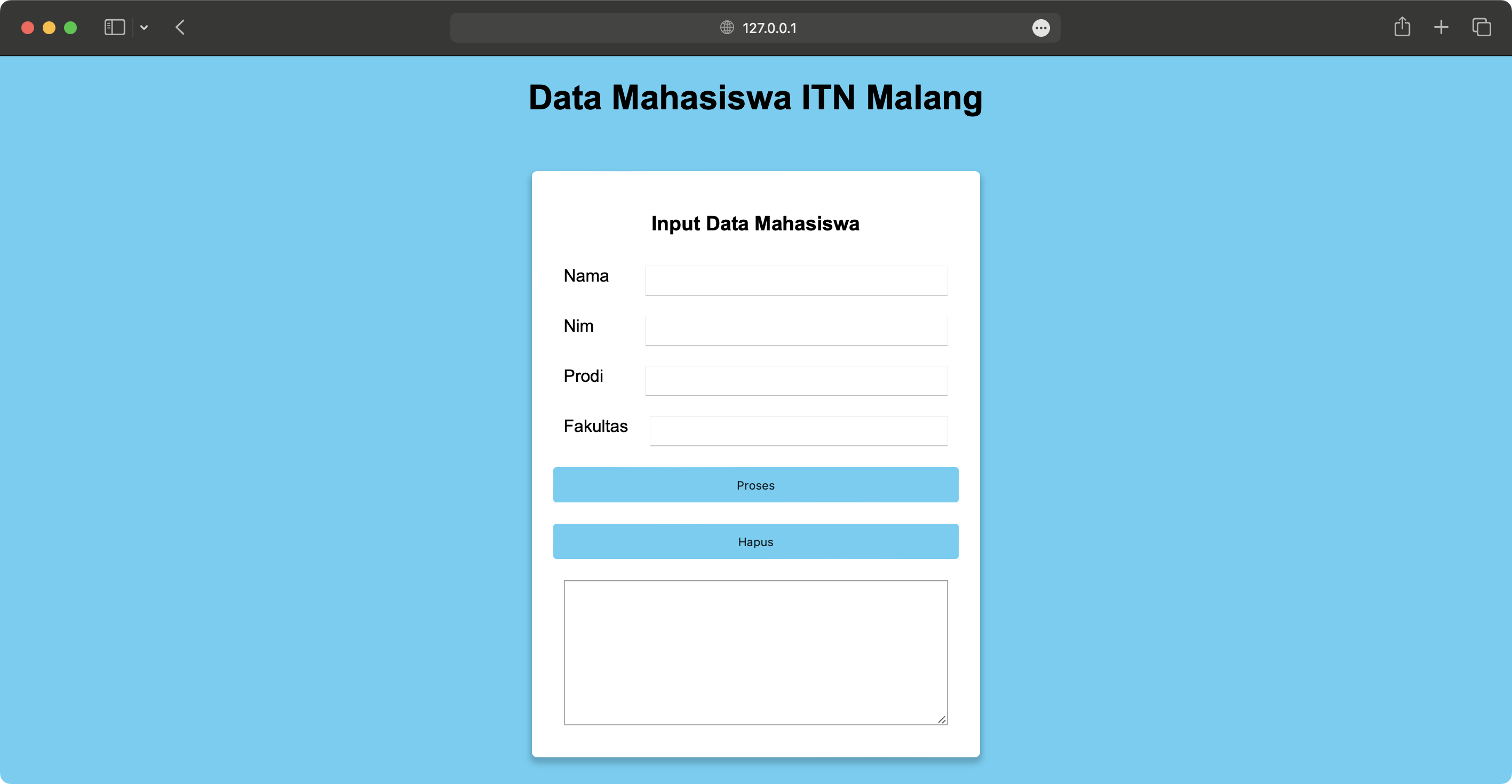
|  |
| --- |
| <body>  <script>  let judul = document.createElement("h1");  judul.textContent = "Membuat Element Dengan Javascript";  document.body.append(judul);  </script>  </body> |

## Latihan

Membuat web form sederhana

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <title>Web Form</title>  <style>  \* {  box-sizing: border-box;  }  body {  background-color: rgb(91, 207, 243);  font-family: Arial, Helvetica, sans-serif;  }  h1 {  margin-bottom: 50px;  }  .container {  display: flex;  flex-direction: column;  background-color: white;  width: 30%;  margin: 20px auto;  padding: 20px;  justify-content: center;  border-radius: 5px;  box-shadow: 0 4px 5px 2px rgba(0, 0, 0, 0.2);  }  .input {  display: flex;  padding: 10px;  }  label {  flex-basis: 20%;  text-align: left;  }  .input\_form {  width: 100%;  padding: 5px;  margin-left: 20px;  }  .button {  width: 100%;  margin: 10px 0;  padding: 10px;  background: rgb(91, 207, 243);  border-radius: 3px;  cursor: pointer;  border: none;  }  .button:hover {  background-color: rgb(59, 194, 235);  font-size: 15px;  }  textarea {  width: 100%;  }  </style>  </head>  <body>  <center>  <h1>Data Mahasiswa ITN Malang</h1>  <div class="container">  <h3>Input Data Mahasiswa</h3>  <form action="#" name="myForm">  <div class="input">  <label for="nama">Nama</label>  <input class="input\_form" type="text" name="nama" id="nama" />  </div>  <div class="input">  <label for="nim">Nim</label>  <input class="input\_form" type="text" name="nim" id="nim" />  </div>  <div class="input">  <label for="prodi">Prodi</label>  <input class="input\_form" type="text" name="prodi" id="prodi" />  </div>  <div class="input">  <label for="fakultas">Fakultas</label>  <input  class="input\_form"  type="text"  name="fakultas"  id="fakultas"  />  </div>  <div class="btn">  <input  type="button"  class="button"  name="btn\_proses"  value="Proses"  onClick="proses()"  />  </div>  <div class="btn">  <input  type="button"  class="button"  name="btn\_hapus"  value="Hapus"  onClick="hapus()"  />  </div>  <div class="input">  <textarea name="hasil" id="" rows="10" disabled></textarea>  </div>  </form>  </div>  </center>  <script>  function proses() {  let myForm = document.myForm;  let nama = myForm.nama.value;  let nim = myForm.nim.value;  let prodi = myForm.prodi.value;  let fakultas = myForm.fakultas.value;  myForm.hasil.value = `Data Mahasiswa  Nama : ${nama}  Nim : ${nim}  Prodi : ${prodi}  Fakultas : ${fakultas}`;  }  function hapus() {  document.myForm.reset();  }  </script>  </body>  </html> |

Hasil :



## Tugas Praktikum

1. Terapkan DOM pada Landing Page / index.html masing masing.
2. Buat Carousel menggunakan javascript (tanpa framework) pada Landing Page / index.html masing masing.