

Modul Panduan Praktikum Pemrograman Web

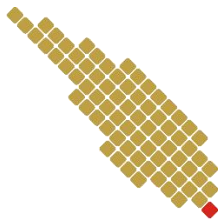
PERTEMUAN

4

JavaScript I

Asisten Praktikum

- Aminudin Fadila
- Defangga Aby Vonega
- Markus Togi Fedrian Rivaldi Sinaga



TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI PRODUKSI INDUSTRI DAN INFORMASI
INSTITUT TEKNOLOGI SUMATERA
2022

Dasar Teori

1. JavaScript



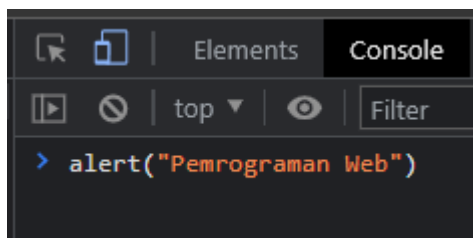
Logo JavaScript

JavaScript adalah bahasa pemrograman yang awalnya dirancang untuk berjalan di atas browser. Namun, seiring perkembangan zaman, JavaScript tidak hanya berjalan di atas browser saja. Saat ini JavaScript juga dapat digunakan pada sisi Server, Game, IoT, Desktop, dsb. JavaScript awalnya bernama Mocha, lalu berubah menjadi LiveScript saat browser Netscape Navigator 2.0 rilis versi beta (September 1995). Namun, setelah itu dinamai ulang menjadi JavaScript. Terinspirasi dari kesuksesan JavaScript, Microsoft mengadopsi teknologi serupa. Microsoft membuat 'JavaScript' versi mereka sendiri bernama JScript. Lalu di tanam pada Internet Explorer 3.0. Hal ini mengakibatkan 'perang browser', karena JScript milik Microsoft berbeda dengan JavaScript racikan Netscape. Akhirnya pada tahun 1996, Netscape mengirimkan standarisasi ECMA-262 ke Ecma International. Sehingga lahirlah standarisasi kode JavaScript bernama ECMAScript atau ES.

2. Penggunaan JavaScript

Untuk menggunakan JavaScript dapat kita lakukan melalui beberapa cara, antara lain:

- **Menggunakan Console**



Console pada browser kita dapat digunakan untuk menjalankan JavaScript. Untuk membuka console pada browser dapat dilakukan dengan :

Klik Kanan → **Inspect** → **Console** atau (**Ctrl+Shift+i**) → **Console** atau menekan **F12** → **Console** (*shortcut* dapat berbeda tergantung browser).

- **Inline Kode JavaScript**

```
<button onclick="alert('Pemrograman Web')">
  Click Me!
</button>
```

Menggunakan atribut **event listener** (dapat berbeda tergantung elemen), untuk menjalankan JavaScript ketika elemen diberikan aksi tertentu.

- **Internal (Embed) Script**

```
.
.
<script>
  alert("Pemrograman Web");
</script>
</body>
```

Menggunakan tag `<script>` sebelum penutup tag `<body>` untuk menjalankan JavaScript dalam sebuah dokumen HTML.

- **External JavaScript**

JavaScript

```
1 alert("Pemrograman Web");
```

HTML

```
.
.
<script src="script.js"></script>
</body>
```

Menggunakan file JavaScript eksternal, melalui tag `<script>` dengan atribut **src** berisi lokasi file JavaScript tersebut sebelum penutup tag `<body>` untuk menjalankan JavaScript dalam sebuah dokumen HTML.

Keempat contoh kode diatas bila dijalankan di browser akan memiliki hasil yang sama sebagai berikut :



3. Variabel dalam JavaScript

- **let**

let merupakan *constructor* variabel yang paling umum digunakan di JavaScript. **let** dapat digunakan untuk membuat variabel dengan tipe data yang dinamis.

```
// Numbers
let myAge      = 22;
let myGrades   = 99.9;

// String
let myFullName = "Alexander The Great";

// Booleans
let iAmHandsome = true;
let iAmOld      = myAge > 50;

// Arrays
let myFriends   = ["Alpha", "Beta", "Charlie", "Deden"];
let favNumbers  = [12, 4, 108, 72];

// Object
let shinobi     = { name: "Naruto", clan: "Uzumaki" };
```

- **const**

const tidak berbeda jauh dengan **let**, perbedaannya terletak pada :

- kita harus menginisialisasikan nilai saat mendeklarasikan variabel menggunakan **const**

```
> let meBeYourMan;
< undefined
> const antinople;
✖ Uncaught SyntaxError: Missing
  initializer in const declaration
```

- kita tidak dapat mengubah nilai pada variabel yang dideklarasikan menggunakan **const** setelah kita inisialisasikan nilai di awal

```
> const antinople = "Constantinople";
< undefined
> antinople = "Sleman";
✖ ▶ Uncaught TypeError: Assignment to
  constant variable.
```

Kapan kita menggunakan **let** dan kapan kita menggunakan **const**? Mengapa harus menggunakan **const** bila dengan menggunakan **let** saja sudah cukup, dan kita tidak dibatasi seperti saat menggunakan **const**? Faktanya **const** sangat berguna, ketika kita menggunakannya, ini menunjukkan pada orang lain (mis. anggota lain dalam tim kita) bahwa nama variabel tersebut tidak akan digunakan untuk nilai lainnya, dan nilai dari variabel tersebut tidak akan berubah di seluruh *flow* dari program. Sehingga kapanpun mereka melihatnya, mereka akan tahu, variabel tersebut merujuk pada nilai apa.

“Gunakan **const** sedapat mungkin, dan gunakan **let** saat dibutuhkan”

- **var**

Di masa-masa awal pembuatan JavaScript, satu-satunya *constructor* variabel yang tersedia adalah **var**. Berbeda dengan **let**, **var** mendukung perilaku yang disebut dengan **hoisting** dan juga mendukung pendeklarasian variabel yang sama lebih dari satu kali. Karena kedua perilaku tersebut justru sering membuat kita bingung saat membaca kode program kita dan juga karena **var** rawan-*error*, maka **let** diciptakan, dan saat ini, penggunaan **let** sangat direkomendasikan dibanding penggunaan **var**.

var

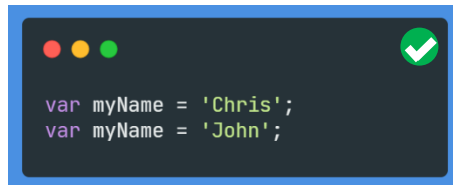


```
myName = 'Chris';

function logName() {
  console.log(myName);
}

logName();

var myName;
```



```
var myName = 'Chris';
var myName = 'John';
```

let

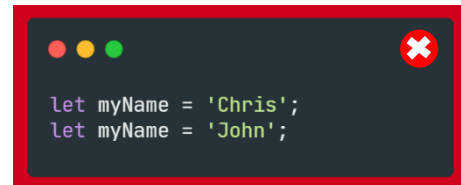


```
myName = 'Chris';

function logName() {
  console.log(myName);
}

logName();

let myName;
```



```
let myName = 'Chris';
let myName = 'John';
```

Selengkapnya tentang variabel di JavaScript dapat dibaca di [Storing the information you need — Variables](#)

4. Percabangan dalam JavaScript

A. Percabangan "if else"



```
if (condition1) {
  statement1;
} else if (condition2) {
  statement2;
} else if (conditionN) {
  statementN;
} else {
  statementLast;
}
```

B. Percabangan "switch case"

```
switch (expression) {  
  case label1:  
    statements1;  
    break;  
  case label2:  
    statements2;  
    break;  
  // ...  
  default:  
    statementsDefault;  
}
```

C. Conditional (Ternary) Operator

Conditional (Ternary) Operator merupakan satu-satunya operator dalam JavaScript yang menerima 3 buah operan :

- Kondisi yang diikuti tanda tanya (?)
- *Statement* untuk dieksekusi bila kondisi benar diikuti dengan titik dua (:)
- *Statement* untuk dieksekusi bila kondisi salah

Operator ini sering digunakan sebagai alternatif (biasa disebut juga dengan *shorthand*) untuk percabangan "if else" (hanya **if else**, tanpa **else if**).

```
condition ? statementIfTrue : statementIfFalse
```

Contoh :

```
let status = nilai > 78 ? "Lulus" : "Tidak Lulus";
```

Sama saja dengan :

```
let status;  
  
if (nilai > 78) {  
  status = "Lulus";  
} else {  
  status = "Tidak Lulus";  
}
```

Selengkapnya tentang Percabangan di JavaScript dapat dibaca di [Conditional Statements](#), dan tentang Ternary Operator di [Ternary Operator](#).

5. Perulangan dan Iterasi dalam JavaScript

A. Perulangan "for"

```
for (let i=0; i<N; i++) {  
  console.log(name[i]);  
}
```

B. Perulangan "while"

```
let i = 0;  
while (i < N) {  
  console.log(name[i]);  
  i++;  
}
```

C. Perulangan "do while"

```
let i = 0;  
do {  
  console.log(name[i]);  
  i++;  
} while (i < N);
```

D. Method "forEach ()"

Array.prototype.forEach() merupakan metode yang dapat kita gunakan pada sebuah array untuk mengeksekusi fungsi untuk masing-masing elemen di dalam array tersebut.

```
name.forEach( function(element){  
  console.log(element);  
});
```

forEach() memiliki 3 parameter default yaitu :

- **element** merujuk pada elemen array saat ini
- **index** merujuk pada indeks dari elemen array saat ini
- **array** merujuk pada array itu sendiri

Selengkapnya tentang **forEach()** dapat dibaca di [Array forEach](#).

E. Method "repeat ()"

`String.prototype.repeat()` merupakan metode yang dapat kita gunakan pada sebuah string untuk membuat string baru dengan cara menggabungkan (*concatenate*) sejumlah salinan terdefinisi dari string awal yang kita berikan metode ini.

```
console.log("Harta, Tahta, Sinaga".repeat(10));
```

`repeat()` menerima satu buah argumen berupa bilangan bulat antara 0 dan positif tak-hingga.

F. Perulangan "for ... in"

(materi opsional) dapat dibaca di [for...in statement](#)

G. Perulangan "for ... of"

(materi opsional) dapat dibaca di [for...of statement](#)

Selengkapnya terkait Perulangan dan Iterasi dalam JavaScript dapat dibaca di [Loops and Iteration](#)

6. Fungsi Dalam JavaScript

A. Fungsi Tanpa Parameter

```
function sayHello() {  
  console.log("Hello World!");  
}  
  
sayHello(); // Hello World!
```

B. Fungsi Dengan Parameter

```
function sayHelloAgain (nama, prodi) {  
  let sifat = prodi === "IF" ? "keren" : "payah";  
  console.log("Halo " + nama + ", kamu " + sifat + "!")  
}  
  
sayHelloAgain("Togi", "IF"); // Halo Togi, kamu keren!
```

C. Arrow Function

Eksresi *Arrow Function* merupakan alternatif ringkas untuk ekspresi fungsi tradisional, namun dengan beberapa batasan tertentu. Perbedaan antara *Arrow Function* dan ekspresi fungsi tradisional dapat dibaca [di sini](#).

Contoh penggunaan ekspresi *Arrow Function* :

```
const arrowHello = (nama, prodi) => {  
  let sifat = prodi === "IF" ? "keren" : "payah";  
  console.log("Halo " + nama + ", kamu " + sifat + "!")  
}  
  
arrowHello("Togi", "IF"); // Halo Togi, kamu keren!
```

Selengkapnya tentang fungsi dalam JavaScript dapat dibaca di [JavaScript Functions](#).

Latihan Percobaan

Dengan menggunakan JavaScript buatlah program yang menerima masukan berupa bilangan x bulat antara 0 dan 21, dengan ketentuan sebagai berikut :

- Bila $x \leq 0$ atau $x \geq 21$ maka program menampilkan pesan *error*.
- Bila x merupakan bilangan ganjil maka program akan menampilkan deret fibonacci dari 1 hingga $\leq (x * 10)$.
- Bila x merupakan bilangan genap maka program akan menampilkan deret perkalian faktorial dari $x/2$ (bukan hanya hasil akhirnya).

Contoh input dan output **i** :

```
> latihan(0);  
Pilih bilangan antara 1 sampai 20
```

Contoh input dan output **ii** :

```
> latihan(30);  
Pilih bilangan antara 1 sampai 20
```

Contoh input dan output **iii** :

```
> latihan(16);  
1x2x3x4x5x6x7x8=40320
```

Contoh input dan output **iv** :

```
> latihan(17);  
1 1 2 3 5 8 13 21 34 55 89 144
```

Tugas

1. Mengerjakan Latihan Percobaan dengan tampilan yang dapat digunakan di halaman website, bukan melalui console.
2. Modifikasi hasil Tugas Pertemuan 3 agar memiliki fungsi mengubah mode tema (**light** / **dark**) dengan menggunakan tombol.

Parameter Penilaian	Bobot Penilaian
Kelengkapan	20%
Kesesuaian penggunaan percabangan, perulangan, dan fungsi pada JavaScript	30%
Program berjalan sesuai spesifikasi yang diberikan (nomor 1)	25%
Tampilan yang semenarik mungkin (nomor 2)	25%

Daftar Pilihan Referensi

- [mdn web docs](#)
- [W3Schools](#)
- [JavaScript Dasar - Web Programming Unpas \(YouTube\)](#)
- [JavaScript Lanjutan - Web Programming Unpas \(YouTube\)](#)