**Intro**

JavaScript adalah bahasa pemrograman yang powerfull yang dapat menambahkan interaktivitas ke situs web. JavaScript serbaguna dan ramah bagi pemula. Dengan lebih banyak pengalaman, anda akan dapat membuat game, grafis animasi 2D dan 3D, aplikasi berbasis database yang komprehensif, dan banyak lagi.

**Pemrograman**

JavaScript termasuk bahasa pemrograman seperti PHP, Java, Pascal, dan lain-lain. JavaScript dengan Java bukanlah bahasa yang sama walaupun memiliki kata “java”. Pemrograman memiliki kata dasar yaitu program yang berarti urutan instruksi terstruktur, yang dirancang agar komputer dapat berperilaku sesuai dengan ketentuan, dengan tujuan untuk menyelesaikan sebuah pekerjaan. Berarti pemrograman merupakan kegiatan membuat program.

**Sejarah**

JavaScript pertama kali dikembangkan oleh Brendan Eich pada tahun 1995 di Netscape Communications Corporation. Saat itu, bahasa pemrograman yang populer adalah Java, dan Netscape ingin mengembangkan bahasa scripting untuk browser web yang lebih ringan dan mudah dipelajari. Maka lahirlah JavaScript.

JavaScript segera populer di antara pengembang web karena kemampuannya untuk menambahkan interaktifitas pada halaman web. Kemudian, pada tahun 1997, Netscape memperkenalkan JavaScript ke European Computer Manufacturers Association (ECMA) untuk menjadi standar internasional.

ECMA mengambil alih pengembangan JavaScript dan mengubah namanya menjadi ECMAScript. Versi pertama ECMAScript dirilis pada tahun 1997 dan disebut ECMAScript 1. Setelah itu, ECMAScript 2 dan ECMAScript 3 dirilis pada tahun 1998 dan 1999 masing-masing.

Pada tahun 2005, ECMAScript 3.1 diumumkan, tetapi tidak pernah dirilis. Sebaliknya, ECMAScript 4 yang dikembangkan pada saat itu dihentikan karena perbedaan pendapat di antara para pengembang.

Pada tahun 2009, ECMAScript 5 dirilis. Versi ini menambahkan banyak fitur baru, termasuk metode objek baru, objek JSON, dan fungsi array baru. ECMAScript 5 menjadi sangat populer di antara pengembang dan digunakan di banyak proyek besar.

Pada tahun 2015, ECMAScript 6 (juga dikenal sebagai ECMAScript 2015) dirilis. Versi ini menghadirkan banyak fitur baru seperti kelas, modul, lambda, dan banyak lagi. ECMAScript 6 menjadi sangat populer di antara pengembang dan digunakan di banyak proyek besar.

Sejak dirilisnya ECMAScript 6, ECMA telah merilis ECMAScript baru setiap tahun dengan peningkatan fitur dan kemampuan. Saat ini, versi terbaru adalah ECMAScript 2021.

JavaScript masih menjadi bahasa pemrograman yang sangat populer dan digunakan di seluruh dunia oleh jutaan pengembang web.

**How To Use?**

Ada 2 cara dalam menempatkan code JavaScript, yaitu pengerjaan code pada Tag HTML<script> dan pemanggilan library external.

-Tag HTML <script>

Pembuatan code JavaScript dapat dilakukan di dalam Tag <script>

-Library external

Pembuatan code JavaScript dapat dilakukan pada file js(ekstensi JavaScript) lalu memanggil file tersebut melalui Tag HTML <script>. Pertama, tuliskan code Javascript lalu simpan dengan nama file misalnya script.js. Lalu panggil file js tersebut pada tag <script> dengan memperhatikan posisi file script.js dibuat. Bila file html dan script.js dalam folder yang sama contohnya <script src=”script.js”></script>. Bila file script.js diluar posisi 1 folder dengan html contohnya <script src=”../script.js”></script>.

**URL**

MDNMozilla : <https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics#what_is_javascript>

WPU : <https://www.youtube.com/watch?v=Ncrlg9kTC6U&list=PLFIM0718LjIWXagluzROrA-iBY9eeUt4w&index=4>

PenempatanTagScript : https://www.duniailkom.com/tutorial-belajar-javascript-posisi-terbaik-meletakkan-kode-javascipt-di-dalam-html/

**Output**

JavaScript memiliki beberapa cara untuk menghasilkan output atau tampilan ke layar. Output di JavaScript dapat digunakan untuk menampilkan informasi, memberikan feedback kepada pengguna, atau memecah bagian-bagian kode agar lebih mudah dipahami. Berikut ini beberapa cara untuk menghasilkan output di JavaScript.

1. Metode console.log() Metode console.log() adalah cara paling umum untuk menghasilkan output di JavaScript. Metode ini digunakan untuk mencetak informasi ke konsol browser atau terminal. Kita dapat menggunakan console.log() untuk mencetak nilai variabel atau pesan teks.
2. Metode document.write() Metode document.write() digunakan untuk menulis HTML atau teks ke dalam dokumen web. Metode ini dapat digunakan untuk menampilkan pesan teks di halaman web.
3. Metode innerHTML Metode innerHTML digunakan untuk menetapkan isi dari elemen HTML. Metode ini dapat digunakan untuk mengganti isi elemen HTML yang sudah ada atau menambahkan elemen HTML baru ke halaman web.
4. Metode alert() Metode alert() digunakan untuk menampilkan pesan pop-up di jendela browser. Metode ini sering digunakan untuk memberikan feedback kepada pengguna atau memberikan peringatan.

**URL**

W3schools : https://www.w3schools.com/js/js\_output.asp

**Variabel & Tipe Data**

**VARIABEL**

Var, let,const

Kata kunci **var** digunakan untuk mendeklarasikan variabel global atau lokal di dalam suatu fungsi. Variabel yang dideklarasikan dengan **var** bersifat hoisting, artinya variabel tersebut dapat diakses sebelum dideklarasikan. Namun, variabel yang dideklarasikan dengan **var** memiliki cakupan atau scope yang lebih luas dibandingkan dengan **let** dan **const**.

Kata kunci **let** digunakan untuk mendeklarasikan variabel lokal di dalam suatu blok kode, seperti di dalam for loop atau if statement. Variabel yang dideklarasikan dengan **let** tidak bersifat hoisting, artinya variabel tersebut harus dideklarasikan sebelum digunakan. Variabel yang dideklarasikan dengan **let** memiliki cakupan atau scope yang lebih sempit dibandingkan dengan **var**, sehingga variabel yang dideklarasikan dengan **let** hanya dapat diakses di dalam blok kode dimana variabel tersebut dideklarasikan

Kata kunci **const** digunakan untuk mendeklarasikan variabel konstan yang tidak dapat diubah nilainya setelah dideklarasikan. Variabel yang dideklarasikan dengan **const** bersifat hoisting dan harus dideklarasikan saat didefinisikan. Variabel yang dideklarasikan dengan **const** memiliki cakupan atau scope yang sama dengan **let**.

**TIPE DATA**

1. Tipe Data Angka (Number)

Tipe data angka digunakan untuk menyimpan nilai numerik seperti bilangan bulat (integer) atau bilangan desimal (float). Contohnya:

1. Tipe Data Teks (String)

Tipe data teks digunakan untuk menyimpan nilai berupa teks atau karakter. Nilai teks diapit oleh tanda kutip (petik tunggal atau petik ganda). Contohnya:

1. Tipe Data Boolean

Tipe data boolean digunakan untuk menyimpan nilai true atau false (benar atau salah). Contohnya:

1. Tipe Data Objek (Object)

Tipe data objek digunakan untuk menyimpan nilai yang lebih kompleks, seperti array atau objek. Contohnya:

**URL**

W3schools : <https://www.w3schools.com/js/js_variables.asp>

MDN : https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Grammar\_and\_types

**Aritmatika & Operator**

Dalam pemrograman JavaScript, aritmatika dan operator digunakan untuk melakukan operasi matematika pada nilai-nilai numerik. Operator digunakan untuk menggabungkan, memanipulasi, dan membandingkan nilai-nilai tersebut. Berikut adalah beberapa operator aritmatika yang umum digunakan dalam JavaScript:

* Operator Penjumlahan (+): Digunakan untuk menambahkan dua nilai.
* Operator Pengurangan (-): Digunakan untuk mengurangkan satu nilai dari nilai lain.
* Operator Perkalian (\*): Digunakan untuk mengalikan dua nilai.
* Operator Pembagian (/): Digunakan untuk membagi satu nilai dengan nilai lain.
* Operator Modulus (%): Digunakan untuk mendapatkan sisa pembagian dua nilai.
* Operator Increment (++) dan Decrement (--): Digunakan untuk menambah atau mengurangkan satu dari nilai yang ada.

Operator Penugasan (Assignment Operators): Operator penugasan digunakan untuk memberikan nilai pada variabel dan melakukan operasi aritmatika secara bersamaan. Berikut adalah contoh beberapa operator penugasan yang sering digunakan:

* Operator Penugasan Penjumlahan (+=): Menambahkan nilai ke variabel dan menyimpan hasilnya.
* Operator Penugasan Pengurangan (-=): Mengurangkan nilai dari variabel dan menyimpan hasilnya.
* Operator Penugasan Perkalian (\*=): Mengalikan nilai ke variabel dan menyimpan hasilnya.
* Operator Penugasan Pembagian (/=): Membagi nilai ke variabel dan menyimpan hasilnya.
* Operator Penugasan Modulus (%=): Menghitung sisa pembagian dan menyimpan hasilnya.

1. Operator Perbandingan (Comparison Operators): Operator perbandingan digunakan untuk membandingkan dua nilai dan menghasilkan nilai boolean (true atau false) sebagai hasilnya. Berikut adalah contoh beberapa operator perbandingan yang umum digunakan:
   * Operator Sama dengan (==): Memeriksa kesetaraan nilai, tanpa memeriksa tipe data.
   * Operator Tidak Sama dengan (!=): Memeriksa ketidaksamaan nilai

**URL**

W3schools : https://www.w3schools.com/js/js\_arithmetic.asp

WPU : https://www.youtube.com/watch?v=EnXClrVdpTM&list=PLFIM0718LjIWXagluzROrA-iBY9eeUt4w&index=10

**String**

Dalam JavaScript, tipe data string digunakan untuk merepresentasikan teks dan kumpulan karakter. String dapat terdiri dari satu karakter atau lebih, dan mereka sering digunakan untuk memanipulasi, memanipulasi, dan menyajikan data teks. Dalam materi ini, kita akan menjelajahi berbagai operasi dan metode yang tersedia untuk memanipulasi string dalam JavaScript.

1. Membuat String: Anda dapat membuat string dalam JavaScript dengan mengapit teks dengan tanda kutip tunggal ('') atau tanda kutip ganda (""). Contohnya:
2. Menggabungkan String (Concatenation): Anda dapat menggabungkan (concatenate) dua atau lebih string menggunakan operator "+" atau dengan menggunakan metode **concat()**. Contohnya:
3. Panjang String: Untuk mendapatkan panjang sebuah string, Anda dapat menggunakan properti **length**. Contohnya:
4. Mengakses Karakter dalam String: Anda dapat mengakses karakter individual dalam string dengan menggunakan indeks. Indeks dimulai dari 0 untuk karakter pertama, 1 untuk karakter kedua, dan seterusnya. Contohnya:

**Escape Character**

Pada JavaScript, escape character (karakter pelarian) digunakan untuk memasukkan karakter khusus ke dalam string yang biasanya sulit atau tidak mungkin diwakili secara langsung. Escape character ditandai dengan penggunaan tanda \ (backslash) diikuti dengan karakter tertentu. Berikut ini beberapa escape character yang umum digunakan dalam JavaScript:

**URL**

W3schools : <https://www.w3schools.com/js/js_strings.asp>

WPU : https://www.youtube.com/watch?v=ud322\_5-M3s&list=PLFIM0718LjIWXagluzROrA-iBY9eeUt4w&index=12

**Array**

Array adalah salah satu struktur data yang paling penting dalam pemrograman. Dalam JavaScript, array digunakan untuk menyimpan dan mengelola kumpulan nilai dalam satu variabel. Array dapat berisi tipe data apa pun, termasuk angka, string, objek, dan bahkan array lainnya. Dalam materi ini, kita akan membahas penggunaan dan operasi dasar yang dapat dilakukan dengan array di JavaScript.

1. Membuat Array: Untuk membuat array di JavaScript, Anda dapat menggunakan sintaksis berikut:

|  |
| --- |
| 1. let namaArray = []; // Array kosong 2. let namaArray = [elemen1, elemen2, elemen3]; // Array dengan elemen-elemen awal |

1. Mengakses Elemen dalam Array: Anda dapat mengakses elemen dalam array menggunakan indeksnya. Indeks dimulai dari 0 untuk elemen pertama dan bertambah satu untuk setiap elemen berikutnya. Contoh:
2. Mengubah dan Menambahkan Elemen dalam Array: Anda dapat mengubah nilai elemen dalam array dengan mengakses indeksnya dan menugaskan nilai baru. Contoh:

|  |
| --- |
| let fruits = ['Apel', 'Jeruk', 'Pisang'];  fruits[1] = 'Mangga';  console.log(fruits); // Output: ['Apel', 'Mangga', 'Pisang'] |

Untuk menambahkan elemen baru ke akhir array, Anda dapat menggunakan metode **push()**:

|  |
| --- |
| let fruits = ['Apel', 'Jeruk', 'Pisang'];  fruits.push('Mangga');  console.log(fruits); // Output: ['Apel', 'Jeruk', 'Pisang', 'Mangga'] |

1. Menghapus Elemen dalam Array: Anda dapat menghapus elemen dalam array menggunakan metode **splice()** atau **pop()**. Metode **splice()** memungkinkan Anda menghapus elemen berdasarkan indeksnya:

|  |
| --- |
| let fruits = ['Apel', 'Jeruk', 'Pisang'];  fruits.splice(1, 1); // Menghapus elemen dengan indeks 1  console.log(fruits); // Output: ['Apel', 'Pisang'] |

Metode **pop()** digunakan untuk menghapus elemen terakhir dalam array:

|  |
| --- |
| let fruits = ['Apel', 'Jeruk', 'Pisang'];  fruits.pop(); // Menghapus elemen terakhir  console.log(fruits); // Output: ['Apel', 'Jeruk'] |

1. Menghitung Jumlah Elemen dalam Array: Anda dapat menggunakan properti **length** untuk menghitung jumlah elemen dalam array:

|  |
| --- |
| let fruits = ['Apel', 'Jeruk', 'Pisang'];  console.log(fruits.length); // Output: 3 |

**URL**

W3schools : https://www.w3schools.com/js/js\_arrays.asp

WPU Arrays : https://www.youtube.com/watch?v=pDSz3sWnPRw&list=PLFIM0718LjIWXagluzROrA-iBY9eeUt4w&index=35

**Iterasi**

Iterasi, atau sering disebut juga perulangan, adalah salah satu konsep penting dalam pemrograman. Dalam konteks JavaScript, iterasi digunakan untuk mengulangi serangkaian perintah atau tindakan tertentu berulang kali. Dalam materi ini, kita akan membahas beberapa metode iterasi yang umum digunakan dalam JavaScript.

For Loop: For loop adalah jenis perulangan yang paling umum digunakan dalam JavaScript. For loop mengulangi blok kode berdasarkan kondisi tertentu. Format dasar for loop adalah sebagai berikut:

While Loop: While loop digunakan untuk mengulangi blok kode selama kondisi tertentu benar. While loop akan terus berulang sampai kondisi menjadi salah. Format dasar while loop adalah sebagai berikut:

Do-While Loop: Do-while loop adalah varian dari while loop. Perbedaannya, blok kode dalam do-while loop akan dijalankan setidaknya sekali sebelum kondisi diperiksa. Format dasar do-while loop adalah sebagai berikut:

Array.forEach(): Metode forEach() digunakan untuk mengulangi elemen-elemen dalam Array dan menjalankan fungsi callback pada setiap elemen. Metode ini tidak mengembalikan nilai apa pun. Format dasar penggunaan Array.forEach() adalah sebagai berikut:

|  |
| --- |
| const arr = [1, 2, 3, 4, 5];  arr.forEach(function(element) {  console.log(element);  }); |

**URL**

W3schools : <https://www.w3schools.com/js/js_iterables.asp>

MDN : https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops\_and\_iteration

**Percabangan**

Pengkodisian merupakan salah satu konsep dasar dalam pemrograman. Dalam JavaScript, pengkodisian memungkinkan Anda untuk mengontrol alur eksekusi program berdasarkan kondisi-kondisi tertentu. Dalam materi ini, kami akan membahas berbagai cara untuk melakukan pengkodisian dalam JavaScript, termasuk pernyataan if, else if, else, serta operator logika seperti AND (&&) dan OR (||).

**IF . ELSE IF**

. Pernyataan if: Pernyataan if adalah cara paling dasar untuk melakukan pengkodisian di JavaScript. Dalam format umumnya, pernyataan if terdiri dari kata kunci "if", diikuti oleh suatu kondisi yang dievaluasi, dan blok kode yang akan dieksekusi jika kondisi tersebut benar.

. Pernyataan if else: Pernyataan if else memungkinkan kita untuk mengeksekusi blok kode tertentu jika kondisi bernilai benar, dan blok kode lainnya jika kondisi bernilai salah.

Pernyataan else if: Pernyataan else if memungkinkan kita untuk mengevaluasi beberapa kondisi secara berurutan. Jika kondisi pertama tidak terpenuhi, maka kondisi berikutnya akan dievaluasi.

IV. Operator Logika: Selain menggunakan pernyataan if, JavaScript juga menyediakan operator logika untuk menggabungkan kondisi-kondisi dalam pengkodisian.

* Operator AND (&&): Operator AND digunakan untuk memeriksa apakah dua kondisi bernilai benar secara bersamaan. Jika kedua kondisi benar, maka blok kode yang terkait akan dieksekusi.

Operator OR (||): Operator OR digunakan untuk memeriksa apakah salah satu dari dua kondisi bernilai benar. Jika salah satu kondisi benar, maka blok kode yang terkait akan dieksekusi.

**SWICTH**

Pernyataan switch: Pernyataan switch digunakan ketika Anda memiliki beberapa kasus yang berbeda yang membutuhkan pengkodisian berdasarkan nilai dari suatu ekspresi. Switch memungkinkan Anda untuk membandingkan nilai ekspresi dengan serangkaian kasus yang mungkin, dan menjalankan blok kode yang sesuai dengan kasus yang cocok.

**TERNARY**

VI. Ternary Operator: Ternary operator adalah cara singkat untuk melakukan pengkodisian dalam satu baris. Operator ini menggunakan tanda tanya (?) dan titik dua (:) untuk mengevaluasi suatu kondisi dan menghasilkan dua kemungkinan nilai berdasarkan kondisi tersebut.

**URL**

W3schools (IF Else IF) : <https://www.w3schools.com/js/js_if_else.asp>

W3schools (Switch): <https://www.w3schools.com/js/js_switch.asp>

WPU : https://www.youtube.com/watch?v=u0bl5DXNmVw&list=PLFIM0718LjIWXagluzROrA-iBY9eeUt4w&index=22

**Events**

Dalam JavaScript, sebuah event adalah suatu kejadian yang terjadi pada elemen HTML atau dokumen secara keseluruhan, seperti klik tombol, memasukkan teks pada form, atau menggulir halaman. Events di JavaScript memungkinkan kita untuk menangani interaksi pengguna dengan halaman web.

Ada beberapa cara untuk mendefinisikan event di JavaScript, yaitu:

1. Inline Event Cara pertama yaitu dengan menambahkan event secara langsung pada elemen HTML menggunakan atribut on<EventName>, seperti onclick, onmouseover, onsubmit, dan sebagainya. Contohnya:
2. Event Handler Property Cara kedua yaitu dengan menambahkan event handler property menggunakan JavaScript. Contohnya:

|  |
| --- |
| const button = document.querySelector('button');  button.onclick = function() {  alert('Hello world!');  }; |

1. addEventListener() Method Cara ketiga yaitu dengan menggunakan metode addEventListener(), yang memungkinkan kita untuk menambahkan lebih dari satu fungsi penangan ke satu event. Contohnya:

|  |
| --- |
| const button = document.querySelector('button');  button.addEventListener('click', function() {  alert('Hello world!');  });  button.addEventListener('click', function() {  console.log('Button clicked!');  }); |

Ada banyak jenis event di JavaScript. Beberapa di antaranya adalah:

1. Mouse Events Mouse Events terjadi ketika pengguna berinteraksi dengan halaman web menggunakan mouse. Beberapa contoh mouse events antara lain:

* click
* dblclick
* mouseover
* mouseout
* mousemove
* mousedown
* mouseup

1. Keyboard Events Keyboard Events terjadi ketika pengguna menekan tombol pada keyboard. Beberapa contoh keyboard events antara lain:

* keydown
* keyup
* keypress

1. Form Events Form Events terjadi ketika pengguna mengisi atau mengirimkan formulir. Beberapa contoh form events antara lain:

* submit
* reset
* focus
* blur
* change

1. Window Events Window Events terjadi ketika halaman web dimuat atau saat pengguna mengubah ukuran jendela browser. Beberapa contoh window events antara lain:

* load
* resize
* scroll
* unload

**URL**

MDN : https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Building\_blocks/Events

**Function**

Function (fungsi) adalah blok kode yang dapat dipanggil untuk melakukan tugas tertentu. Dalam JavaScript, function merupakan bagian integral dari bahasa tersebut dan digunakan secara luas dalam pengembangan web.

|  |
| --- |
| function namaFunction(parameter1, parameter2, ...) {  // Kode yang akan dijalankan  return nilaiKembalian;  } |

* **function**: Kata kunci untuk mendefinisikan sebuah function.
* **namaFunction**: Nama unik untuk function yang Anda definisikan.
* **parameter1, parameter2, ...**: Parameter (opsional) yang diterima oleh function.
* **kode yang akan dijalankan**: Bagian kode yang akan dijalankan ketika function dipanggil.
* **return nilaiKembalian**: Statement **return** (opsional) yang mengembalikan nilai dari function.

Function dapat dipanggil dengan menggunakan nama function diikuti oleh tanda kurung dan argumen (jika ada) di dalamnya.

## Scope Variabel

Variabel yang didefinisikan di dalam function hanya dapat diakses di dalam function tersebut. Variabel yang didefinisikan di luar function dapat diakses dari dalam function.

|  |
| --- |
| var nama = "Sarah";  function sapa() {  var pesan = "Halo, ";  console.log(pesan + nama);  }  sapa(); // Output: "Halo, Sarah"  console.log(pesan); // Error: pesan is not defined |

**URL**

W3schools : <https://www.w3schools.com/js/js_functions.asp>

MDN : https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Functions

WPU : https://www.youtube.com/watch?v=6-UqHXBtYkg&list=PLFIM0718LjIWXagluzROrA-iBY9eeUt4w&index=27