- Introduction
  - HTML + CSS = web that doesnt do anything
  - JavaScript adds interactivity and custom behaviors
  - Runs on the user's machine not on the server
  - JavaScript created by Brendan Eich at Netscape in 1995, awalnya namanya LiveScript

- JavaScript and HTML
  - <script>...</script>
  - Can be placed **anywhere** within HTML doc

```
<html>
<head>
        <title>Hello World</title>
</head>
<body>
        <script type="text/javascript">
                document.write("Hello World")
        </script>
        <noscript>
                Your browser doesn't support
                or has disabled JavaScript
        </noscript>
</body>
</html>
```

  - We can include JavaScript code from our own website or anywhere

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

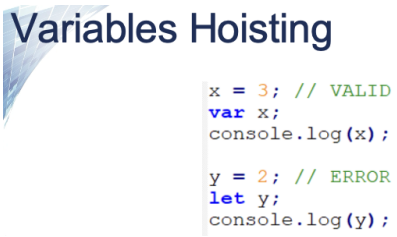
```
<script type="text/javascript"
src="http://someserver.com/script.js"></script>
```

- Variables
  a. var
     - Function scope = variabel yang dideklarasikan hanya akan tersedia di dalam fungsi di mana ia dideklarasikan atau **global** jika **di luar fungsi**. Jika dideklarasikan di dalam blok (seperti di dalam if, for), variabel tetap tersedia di luar blok tersebut
     - Hoisting = variabel yang dideklarasikan **di-hoist** (dipindahkan ke atas) tetapi tidak akan diinisialisasi hingga eksekusi mencapai baris deklarasi tersebut

```
var x = 10;
console.log(x); // Output: 10
if (true) {
    var x = 20;  // Re-declare
    console.log(x); // Output: 20
}
console.log(x); // Output: 20
```

b. let
   - block scope = Variabel hanya akan tersedia di dalam blok di mana ia dideklarasikan (misalnya dalam blok {} dari if, for, atau function)
   - Hoisting = di-hoist, tetapi tidak dapat diakses sebelum deklarasinya karena berada dalam "Temporal Dead Zone (TDZ)" hingga variabel benar-benar dideklarasikan

c. const
   - block scope, cannot be reassigned

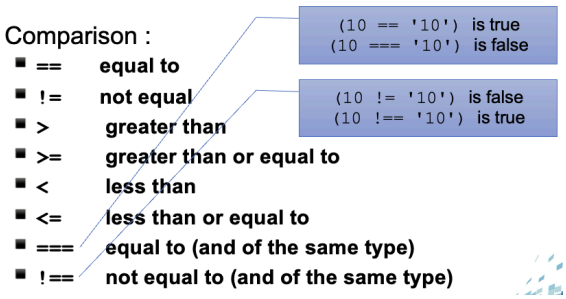


- Dynamic Type
  let x; //x is undefined
  let x = 5; //x is a Number
  let x = “John”; //x is a String

- Latihan
  1. let x = 16 + “Volvo”; //16Volvo
  2. let x = 16 + 4 + “Volvo”; //20Volvo
  3. let x = “Volvo” + 16 + 4; //Volvo164
     When adding a number and a string, JavaScript will treat the number as a string

  1. n = “123” //string
  2. n *= 1 //convert jadi number
  3. n = 123 //number
  4. n += “” //convert jadi string

- Comparison



- Console
  - To log something, primarily for debugging purposes

```
> const name = 'John Doe';
<· undefined
> console.log(name);
  John Doe
<· undefined
> |
```

- Functions

```
function hello(name) {
    console.log('Hello!'+name);
    console.log('Welcome to JavaScript');
}
hello();
```

- Arrays
  - To store multiple values in a single variable
  - push = insert new value into an array
  - pop = deletes last inserted value and returns it

```
let arr = [];
top = ['R', 'G', 'Y'];
mid = ['W', 'R', 'O'];
bot = ['Y', 'W', 'G'];
face = [top, mid, bot];
face = [['R','G','Y'], ['W','R','O'],
['Y','W','G']];
```

```
sports = ["Football", "Tennis", "Baseball"];
console.log("Start = " + sports + "<br />");
sports.push("Hockey");
console.log("After Push = " + sports + "<br />");
removed = sports.pop();
console.log("After Pop = " + sports + "<br />");
console.log("Removed = " + removed + "<br />");
```

- Associative Arrays
  - Arrays with named indexes
  - JS tidak support arrays with named indexes, jika pake, JS will redefine the array to an objects

```
const person = [];
person["firstName"] = "John";
person["lastName"] = "Doe";
person["age"] = 46;
person.length;     // Will return 0
```

- Objects

```
balls = {
    "golf": "Golf balls, 6",
    "tennis": "Tennis balls, 3",
    "soccer": "Soccer ball, 1",
    "ping": "Ping Pong balls, 1 doz"
}

for (let b in balls)
    document.write(b + " = " + balls[b] + "<br />")
```

- Class in JavaScript
    a. ES6
        - “this” untuk refer other fields and methods in class

```
const x = new SomeClass();
const y = new SomeClass();
y.someMethod();

class ClassName {
  constructor(params) {
    this.someField = someParam;
  }
  methodName() {
    const someValue = this.someField;
  }
}
```

    b. Functional

```
var person = {
  firstName: "John",
  lastName : "Doe",
  id       : 5566,
  fullName : function() {
    return this.firstName + " " + this.lastName;
  }
};
```

    c. ES5

```
function Circle(id, x, y, radius) {
    this.id = id;
    this.x = x;
    this.y = y;
    this.radius = radius;
}
Circle.prototype.constructor = Circle;

Circle.prototype.toString = function() {
    return 'Circle > ' + this.id;
};

Circle.prototype.getLocation = function() {
    return {
        x: this.x,
        y: this.y
    };
};

// test the classes
var myCircle = new Circle('mycircleid', 100, 200, 50); // create new instance
console.log(myCircle.toString()); // Circle > mycircleid
console.log(myCircle.getLocation()); // { x: 100, y: 200 }
```