Javascript

Part 1

Presented by Fei

01

What is JavaScript?

02

Data types & variables

03

String

04

Array and object

05

Control Flow

06

Function

07

Class and comments

08

Import and export

What is JavaScript?

Programming language yang sering dipakai untuk membuat website yang interaktif dan dinamis. Penulisan dari Javascript juga tidak terlalu kaku seperti programming language lain.
Javascript seringkali digabungkan dengan HTML dan CSS sehingga design dari website dapat menjadi lebih interaktif dan memenuhi fungsionalitas yang diharapkan.

HTML, CSS, and JS

Data Types & Variables

Data Types (7)
Null, undefiined, boolean, string, symbol, number, dan object

Variables - Scope (3)
Var, let, dan const

Operasi yang dapat dilakukan pada variables adalah declaration, assignment, dan initialization.

Gunakan object.freeze(objName) agar tidak dapat diassign value lain

Visibility (2)
Global and private variable

Data types and variables in JS

Declaring String Variable

Tipe data string dalam Javascript dapat dideklarasikan dengan berbagai cara.
Cara 1: menggunakan double quotes (“...”)
Cara 2: menggunakan single quote (‘...’)
Cara 3: menggunakan backtick (`...`)

Bagaimana jika kita menggunakan cara 1 namun ingin agar value dari string juga memiliki petikan langsung di dalamnya?

Gunakan escape sequence

```
1  var escapeChar = "\"I'm gonna be a doctor\", she said";
2  var singleQuote = '"I\'m gonna be a doctor", she said';
3  var backTick = `"I'm gonna be a doctor", she said`;
4
5  var strNumber = " first";
6  var strOne = "This is my" + strNumber + " string";
7  console.log(strOne);
8  strOne += " but you can have it too";
9  console.log(strOne);
10
```

String variable declaration and escape sequence

Template Literals in String

Gunakan backtick untuk membuat multi-sentences dan dollar dengan bracket untuk menambahkan value dari sebuah variable pada sebuah string.

```
1  const greeting = `Hello, my name is ${pet.name}!
2  I am ${pet.age} years old`;
3  console.log(greeting);
4
```

Template literals in JS

Array

Struktur data yang dapat menyimpan banyak value dengan berbagai jenis data types di dalamnya.

Array dapat terdiri dari satu dimensi atau lebih.
Operasi yang dapat dilakukan untuk memanipulasi sebuah array (4):

- push() - add di belakang
- unshift() - add di depan
- shift() - remove dari belakang
- pop() - remove dari depan

```
1  var myArray = [["Fei", 20]];
2
3  console.log(myArray);
4
5  myArray.push(["Watermelon", 3.5]);
6  console.log(myArray);
7  console.log(myArray[1][0]);
8
9  myArray.pop();
10 console.log(myArray);
11
12 myArray.unshift(["Apple", 10]);
13 console.log(myArray);
14
15 myArray.shift();
16 console.log(myArray);
```

```
[ [ 'Fei', 20 ] ]
[ [ 'Fei', 20 ], [ 'Watermelon', 3.5 ] ]
Watermelon
[ [ 'Fei', 20 ] ]
[ [ 'Apple', 10 ], [ 'Fei', 20 ] ]
[ [ 'Fei', 20 ] ]
```

Array in Javascript

Object

Tipe data komposit yang dapat menyimpan banyak properties yang memiliki values dengan berbagai tipe data.

Operasi yang dapat dilakukan adalah deklarasi, menambahkan, mengedit, dan menghapus properti tertentu dari sebuah object.

Untuk mengecek apakah suatu object memiliki sebuah property, kita dapat menggunakan objName.hasOwnProperty(propName)

Object in Javascript

```
var myDogss = {
    "first dog": {
        "name": "Alpha",
        "age": 4
    },
    "second dog": {
        "name": "Beta",
        "age": 2
    }
}

console.log(myDogss["first dog"].name);
console.log(myDogss["second dog"]["age"]);

var search = "first dog";
myDogss[search]["tail"] = 2;
console.log(myDogss);

myDogss["second dog"]["age"] = 5;
console.log(myDogss);

delete myDogss["first dog"]["tail"];
console.log(myDogss);
```

```
Alpha
2
{
  'first dog': { name: 'Alpha', age: 4, tail: 2 },
  'second dog': { name: 'Beta', age: 2 }
}
{
  'first dog': { name: 'Alpha', age: 4, tail: 2 },
  'second dog': { name: 'Beta', age: 5 }
}
{
  'first dog': { name: 'Alpha', age: 4 },
  'second dog': { name: 'Beta', age: 5 }
}
```

Destructuring Array and Object

```
var grades = {"momo": 87, "jessie": 99, "brownie": 95};
const {"momo" : a, "jessie" : b, "brownie" : c} = grades;
console.log(b);

const [d, e, , f] = [2, 3, 4, 5, 6, 7];
console.log(d, e, f);

```

Hasil:

```
99
2 3 5
```

Array and object destructuring in JS

Control Flow

Javascript akan meng-execute code dengan aturan top-bottom.

Beberapa cara untuk mengontrol flow program:

- Sequential execution
- Switch statements
- Conditional execution
- Looping statements

Hasil:

```
Value n = 5
Value n = 2
Value n = 12
You have been directed to page c
You have been kicked
You have been directed to admin page
```

```
1  var n = 5;
2  console.log("Value n = " + n);
3  n -= 3;
4  console.log("Value n = " + n);
5  n += 10;
6  console.log("Value n = " + n);
```

```
1  function switchStatement(opt) {
2      switch(opt) {
3          case 1:
4              console.log("You have been directed to page a");
5              break;
6          case 2:
7              console.log("You have been directed to page b");
8              break;
9          case 3:
10         case 4:
11             console.log("You have been directed to page c");
12             break;
13         case "Admin":
14             console.log("You have been directed to admin page");
15             break;
16         default:
17             console.log("You have been kicked");
18             break;
19     }
20 }
21
22 switchStatement(3);
23 switchStatement(5);
24 switchStatement("Admin");
```

Control flow in JS

Control Flow
(Conditional Execution)

Hasil:

```
Enough quack
Too many
```

```
// if - else
var duck = 15;
if(duck % 3 == 0) {
    console.log("Enough quack");
}
else {
    console.log("Buy more duck");
}

// else if
var duck = 10;
if(duck < 3) {
    console.log("Less than 3")
}
else if(duck < 6) {
    console.log("More than 3 but less than 6");
}
else if(duck == 9) {
    console.log("Perfect amount")
}
else {
    console.log("Too many");
}
```

Control flow in JS

Control Flow – Loop

Hasil:

```
Animals fed (for loop): 6
Animals fed (while): 5
Animals fed (do-while): 5
```

```
// for loop
var food = 60;
var fed = 0;

for(var i = food; i >= 10; i -= 10) {
    fed++;
}

console.log("Animals fed (for loop): " + fed);

// while loop
var food = 50;
var fed = 0;

while(food >= 10) {
    fed++;
    food -= 10;
}

console.log("Animals fed (while): " + fed);

// do-while loop
var food = 50;
var fed = 0;

do {
    fed++;
    food -= 10;
} while(food >= 10);

console.log("Animals fed (do-while): " + fed);
```

Control flow in JS

Function & Arrow Function

Reusable code yang dapat digunakan untuk melakukan suatu task tertentu yang dapat melibatkan parameter.

Arrow function memiliki syntax yang lebih pendek daripada regular function. Dapat digunakan jika ingin membuat function yang anonymous.

Hasil:

```
true
false
[ 4, 25, 1 ]
```

```
function isRemainingZero(arg1) {
    return arg1 % 2 == 0 ? true : false;
}

console.log(isRemainingZero(8));
console.log(isRemainingZero(9));

// arrow function with filter and map function
const realNum = [2, 3.4, 5, -2.4, -5, 3.9, 1];
const squareList = (arr) => {
    const squaredIntegers = arr.filter(num => Number.isInteger(num) && num > 0).map(x => x * x);
    return squaredIntegers;
};

console.log(squareList(realNum));
```

Function in Javascript

Higher Order Function

Merupakan function yang mereturn function lain dan dimanfaatkan untuk mengabstraksi cara kerja function.

```
function firstFunction() {
    return function(number) {
        console.log(`Number ${number}`);
    }
}

var secondFunction = firstFunction();
secondFunction(5);
```

Hasil:

```
Number 5
```

Function in Javascript

Class & Comments

Sebuah instance yang dapat memiliki constructor untuk mereturn new dan memiliki atribut serta behavior yang dapat diakses dalam methods yang dimiliki.
Sebuah class juga biasanya memiliki getters and setters method.

```
function makeClass() {
    // input: celcius, output: fahrenheit
    class Thermostat {
        constructor(temp) {
            /* '_' means that this variable is a private var
            that can only be accessed within this class */
            this._temp = 9/5 * (temp + 32);
        }

        // return the last temperature stored in private var
        get temperature() {
            return this._temp;
        }

        // function to update the temperature
        set temperature(updated) {
            this._temp = 9/5 * (temp + 32);
        }

    }

    return Thermostat;
}

const Thermostat = makeClass();
const thermos = new Thermostat(32);
let temp = thermos.temperature;
console.log("Before update = " + temp);
thermos.temperature = 34;
temp = thermos.temperature;
console.log("After updated = " + temp);
```

Hasil:

```
Before update = 115.2
After updated = 264.96
```

Class and comments in JS

Import & Export

Code akan direstruktur agar menjadi lebih rapi dengan membaginya menjadi beberapa file berbeda. Setiap function yang dapat digunakan oleh file lain akan memiliki “export”.

Hasil:

```
Before: hEllo EveRyoNe
After: HELLO EVERYONE
```

Import and export in JS

```
const capitalizeString = str => str.toUpperCase();

export {capitalizeString};
```

```
import { capitalizeString } from './strfunction.js';
const str = "hEllo EveRyoNe";

console.log(`Before: ${str}`);
console.log(`After: ${capitalizeString(str)}`);
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Javascript - Fei</title>
</head>
<body>
    <script type="module" src="index.js"></script>
</body>
</html>
```

Thank you!