JavaScript Standard Library

Eko Kurniawan Khannedy

Eko Kurniawan Khannedy

- Technical architect at one of the biggest ecommerce company in Indonesia
- 10+ years experiences
- www.programmerzamannow.com
- youtube.com/c/ProgrammerZamanNow

Eko Kurniawan Khannedy

- Telegram : @khannedy
- Facebook : fb.com/ProgrammerZamanNow
- Instagram : instagram.com/programmerzamannow
- Youtube : youtube.com/c/ProgrammerZamanNow
- Telegram Channel : t.me/ProgrammerZamanNow
- Email : echo.khannedy@gmail.com

Sebelum Belajar

- HTML
- CSS
- JavaScript Dasar
- JavaScript Object Oriented Programming

Agenda

- Number
- String
- Array
- Object
- JSON
- Dan lain-lain

Number

Number

- Number merupakan function yang digunakan untuk melakukan konversi ke tipe data number
- Jika data tidak bisa dikonversi ke number, secara otomatis hasilnya adalah NaN
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number

Kode : Number

```
const input = "12345";
const number = Number(input);

console.info(number);
```

Number Static Properties

- Number memiliki banyak static properties, seperti :
- Number.MIN_VALUE untuk mendapat number minimal
- Number.MAX_VALUE untuk mendapat number maksimal
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number#static_properties

Kode : Number Static Properties

```
console.info(Number.MIN_VALUE);
console.info(Number.MAX_VALUE);
```

Number Static Method

- Number memiliki banyak static method, seperti :
- Number.isNaN(value) untuk mengecek apakah value NaN atau bukan
- Number.isInteger(value) untuk mengecek apakah value berupa integer atau bukan
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number#static_methods

Kode : Number Static Method

```
const data = Number('12345');

console.info(Number.isInteger(data));
console.info(Number.isNaN(data));
```

Number Instance Method

- Number memiliki banyak instance method, seperti :
- Number.toLocalString(locale) untuk mengubah number menjadi string sesuai locale : https://www.lansweeper.com/knowledgebase/list-of-currency-culture-codes/
- Number.toString(radix) untuk mengubah number menjadi string sesuai radix
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number#instance_methods

Kode : Number Instance Method

```
const value = Number(12345);
console.info(value.toString(2));
console.info(value.toLocaleString("id-ID"));
```

String

String

- Tipe data String sudah kita bahas pada materi JavaScript Dasar
- Namun kita belum membahas instance method atau juga instance properties yang terdapat di String
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String

String Instance Method dan Properties

- String sendiri memiliki banyak sekali instance method dan properties
- Hal ini menjadikan untuk memanipulasi data String sangat mudah di JavaScript, seperti mengubah menjadi lowercase, UPPERCASE, memotong string menjadi array dan lain-lain
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String#instance_methods

Kode : String Instance Method dan Properties

```
const name = "Eko Kurniawan Khannedy";

console.info(name.length);
console.info(name.toLowerCase());
console.info(name.toUpperCase());
console.info(name.split(" "));
```

Array

Array

- Tipe data Array sudah kita bahas di materi JavaScript Dasar
- Namun kita hanya membahas beberapa instance method yang ada di Array
- Sebenarnya, terdapat banyak sekali Instance Method yang ada di Array, dan kita akan coba bahas beberapa di sini
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array

Array Loop

- Sebelumnya kita biasa menggunakan for in dan for of untuk melakukan iterasi Array, namun Array juga memiliki method yang bernama forEach()
- Method forEach bisa digunakan juga untuk melakukan iterasi data array

Kode : Array Loop

```
const array = ["Eko", "Kurniawan", "Khannedy"]

array.forEach(function (value, index) {
    console.info(`${index} : ${value}`);
})

array.forEach((value, index) => console.info(`${index} : ${value}`));

array.forEach(value => console.info(value));
```

Array Queue

- Dalam struktur data, terdapat tipe struktur data bernama Queue (Antrian)
- Dimana data masuk akan diposisikan di urutan paling belakang
- Sedangkan data keluar akan diposisikan dari urutan paling depan
- Mirip sekali dengan antrian, atau istilahnya FIFO (first in first out)
- Kita bisa menggunakan Array dengan bantuan function push() untuk menambah data di belakang, dan shift() untuk mengambil dan menghapus data paling depan

Kode : Array Queue

```
const queue = [];

queue.push("Eko")
queue.push("Kurniawan")
queue.push("Khannedy")

console.info(queue.shift());
console.info(queue.shift());
console.info(queue.shift());
```

Array Stack

- Struktur data Stack (Tumpukan) kebalikan dari Queue, dimana aturannya mirip dengan tumpukan kartu
- Saat kita memasukkan data, kita akan memasukkan di urutan paling belakang (atau atas)
- Sedangkan saat kita mengambil data, kita akan mengambil data dari paling belakang (atau atas) terlebih dahulu
- Stack ini sifatnya LIFO (last in first out)
- Untuk menambah diurutan belakang, kita bisa menggunakan function push()
- Dan untuk mengambil dan menghapus paling belakang, kita bisa menggunakan function pop()

Kode : Array Stack

```
const stack = [];

stack.push("Eko")
stack.push("Kurniawan")
stack.push("Khannedy")

console.info(stack.pop());
console.info(stack.pop());
console.info(stack.pop());
```

Array Search

Array memiliki banyak function untuk melakukan pencarian :

Function	Keterangan
find(value => boolean) : value	Mencari data sesuai dengan kondisi
findIndex(value => boolean) : number	Mencari data index sesuai dengan kondisi
includes(value) : boolean	Mengecek apakah terdapat data
indexOf(value) : number	Mengecek posisi index data
lastIndexOf(value) : number	Mengecek posisi index data terakhir

Kode : Array Search

```
const source = [1, 2, 3, 4, 5, 1, 2, 3, 4, 5]

console.info(source.find(value => value > 3)) // 4
console.info(source.findIndex(value => value > 3)) // 3
console.info(source.includes(7)) // false
console.info(source.indexOf(5)) // 4
console.info(source.lastIndexOf(5)) // 9
```

Array Filter

Array memiliki function untuk melakukan filter data

Function	Keterangan
filter(value => boolean) : Array	Melakukan filter data yang kondisinya bernilai true

Kode : Array Filter

```
const numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

console.info(numbers.filter(value => value % 2 === 1))
console.info(numbers.filter(value => value % 2 === 0))
```

Array Transform

Array juga memiliki function yang digunakan untuk melakukan transformasi

Function	Keterangan
map(value => result) : Array<result>	Melakukan transform tiap value dan menghasilkan array result
reduce(resultBefore, value => result) : result	Melakukan transform dengan menggunakan value array dan value selanjutnya, lalu hasilnya dilanjutkan ke iterasi selanjutnya
reduceRight(resultBefore, value => result)	Sama seperti reduce(), namun dilakukan dari belakang

Kode : Array Transform

```
const names = "Eko Kurniawan Khannedy".split(" ");
console.info(names.map(value => value.toUpperCase()));

const numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
console.info(numbers.reduce((result, value) => result + value))
console.info(numbers.reduceRight((result, value) => result + value))
```

Dan Function lain-lain

- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array#instance_methods

Object

Object

- Tipe data object sudah sering sekali kita bahas di JavaScript Dasar dan JavaScript OOP
- Pada materi ini, kita akan bahas banyak static method yang terdapat pada Object
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object

Object Freeze & Seal

- Secara default, object bisa diubah atau dihapus properties nya
- Jika kita ingin mengubah sebuah object menjadi object yang tidak bisa diubah atau dihapus, kita bisa menggunakan beberapa static method
- Object.freeze() digunakan untuk mengubah object menjadi object yang tidak bisa diubah atau dihapus attribute nya
- Object.seal() digunakan untuk mengubah object menjadi object yang tidak bisa dihapus attribute nya

Kode : Object Freeze

```
const person = {
    firstName: "Eko",
    lastName: "Kurniawan"
};

Object.freeze(person);

person.firstName = "Diubah"; // tidak berubah
delete person.firstName; // tidak berubah
console.info(person);
```

Object Assign

- Kadang kita ada kasus menggabungkan semua attribute dari sebuah object ke object lain
- Hal ini bisa kita lakukan dengan menggunakan Object.assign(target, source)

Kode : Object Assign

```
const target = {firstName: "Eko"};
const source = {lastName: "Khannedy"}

Object.assign(target, source);
console.info(target);
```

Object Property Name & Value

- Object juga memiliki static method untuk digunakan mengambil semua properties names dan values
- Object.values() digunakan untuk mengambil semua property value
- Object.getPropertyNames() digunakan untuk mengambil semua properti name

Kode : Object Property Name & Value

```
const person = {
    firstName: "Eko",
    lastName: "Kurniawan"
};

console.info(Object.values(person));
console.info(Object.getOwnPropertyNames(person));
```

Dan Function Lain-Lain

- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object

JSON

Pengenalan JSON

- JSON singkatan dari JavaScript Object Notation
- JSON merupakan data String yang bentuknya mirip dengan JavaScript Object
- Saat ini JSON banyak sekali digunakan untuk komunikasi antara Server dan Client
- https://www.json.org/json-en.html

JSON

- JavaScript mendukung konversi data dari String json ke Object ataupun sebaliknya
- JSON.stringify() digunakan untuk melakukan konversi dari Object ke String
- JSON.parse() digunakan untuk melakukan konversi dari String ke Object
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON

Kode : JSON

```
const person = {
    firstName: "Eko",
    lastName: "Khannedy",
    address: {
        country: "Indonesia",
        city: "Subang"
    }
};
```

```
const json = JSON.stringify(person);
const personAgain = JSON.parse(json);

console.info(json);
console.info(personAgain);
```

BigInt

BigInt

- BigInt merupakan tipe data Number yang bisa mencakup data angka lebih dari Number.MAX_SAFE_INTEGER
- Untuk kasus number yang lebih dari itu, sangat disarankan menggunakan tipe data BigInt
- Cara penggunaan BigInt sama saja dengan penggunaan Number dan juga operator nya
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/BigInt

Kode : BigInt

```
const a = BigInt(Number.MAX_SAFE_INTEGER);
const b = BigInt(Number.MAX_SAFE_INTEGER);

const c = a + b;

console.info(c);
console.info(typeof c);
```

Date

Date

- JavaScript memiliki tipe data untuk representasi waktu dan tanggal, yaitu Date
- Date merupakan representasi milisecond sejak tanggal 1 Januari 1970, atau dikenal dengan Epoch & Unix Timestamp
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date

Membuat Object Date

Untuk membuat object Date, kita bisa menggunakan new Date(), dimana terdapat constructor parameter

new Date()	Membuat date saat ini
new Date(year, month, date)	Membuat date dengan tanggal
new Date(year, month, date?, hour?, minute?, second?, milis?)	Membuat date dengan parameter sampai milis
new Date(timestamp)	Membuat date dari epoch atau unix timestamp

Kode : Membuat Object Date

```
const date1 = new Date();
console.info(date1);

const date2 = new Date(2020, 10, 10);
console.info(date2);

const date3 = new Date(2020, 10, 10, 1, 0, 0, 0);
console.info(date3);
```

Epoch & Unix Timestamp

- Dalam menggunakan tipe data waktu, biasanya disemua bahasa pemrograman akan mendukung yang namanya epoch & unix timestamp
- Epoch & Unix timestamp merupakan hitungan miliseconds setelah tanggal 1 Januari 1970
- JavaScript pun mendukung pembuatan waktu dalam bentuk epoch dan unix timestamp
- Untuk mendapatkan waktu saat ini dalam epoch & unix timestamp, kita bisa menggunakan Date.now()
- Untuk mengubah dari object date ke epoch & unix timestamp, kita bisa menggunakan function getTime()

Kode : Unix Timestamp

```
const timestamp = Date.now();
console.info(timestamp);

const dateUnix = new Date(timestamp);
console.info(dateUnix);

console.info(dateUnix.getTime());
```

Parsing Date

- Kita juga bisa melakukan parsing membuat date dari string menggunakan method Date.parse(value)
- Format string harus YYYY-MM-DDTHH:mm:ss.sssZ
- Dimana jika kita hanya membuat date berisi tanggal saya, kita cukup gunakan YYYY-MM-DD
- Jika date dengan tanggal dan waktu, gunakan YYYY-MM-DDTHH:mm:ss.sss
- Jika date dengan tanggal, waktu dan timezone, gunakan YYYY-MM-DDTHH:mm:ss.sssZ
- Hasil parsing adalah unix timestamp, bukan object date

Format Date

YYYY	Tahun
MM	Bulan
DD	Tanggal
T	Pemisah tanggal dan waktu
HH	Jam
mm	Menit
ss	Detik
sss	Milidetik
Z	Timezone

Kode : Parsing Date

```
const parseTimestamp = Date.parse("2020-12-18T10:10:10.123+07:00")
console.info(parseTimestamp);

const parseDate = new Date(parseTimestamp);
console.info(parseDate);
```

Date Getter dan Setter

- Date juga memiliki banyak sekali method untuk mendapatkan informasi date dan juga mengubah informasi date, atau istilahnya adalah getter dan setter
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date#instance_methods

Kode : Date Getter dan Setter

```
const date = new Date();
date.setFullYear(2020);

console.info(date.getFullYear());
console.info(date.getMonth());
console.info(date.getDate());
console.info(date.getHours());
console.info(date.getMinutes());
console.info(date.getSeconds());
console.info(date.getMilliseconds());
console.info(date.getTimezoneOffset());
```

Math

Math

- Math merupakan class di JavaScript yang berisikan static property dan method untuk operasi matematika
- Ada banyak sekali static property dan method di Math
- Math hanya bisa digunakan untuk tipe data Number, tidak bisa digunakan untuk tipe data BigInt
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math

Kode : Math

```
console.info(Math.max(10, 11, 9, 13, 8)); // 13
console.info(Math.min(10, 11, 9, 13, 8)); // 8
console.info(Math.round(10.5)); // 11
console.info(Math.round(10.3)); // 10
console.info(Math.floor(10.8)); // 10
```

Boolean

Boolean

- Boolean merupakan wrapper class untuk tipe primitif boolean
- Boolean memiliki method toString() untuk mengkonversi ke String
- Dan memiliki method valueOf() untuk mengkonversi ke tipe boolean primitif
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Boolean

Kode Boolean

```
const boolean = Boolean(true);

console.info(boolean);
console.info(boolean.toString());
console.info(boolean.valueOf());
```

Map

Map

- Map merupakan representasi dari struktur data key-value
- Map mirip dengan tipe data object, hanya saja pada Map, semua method untuk manipulasi data sudah disediakan
- Map mengikuti kontrak iterable, sehingga bisa di iterasi secara default
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Map

Perbedaan Map dan Object

Map	Object
Pertama dibuat, tidak memiliki key	Karena memiliki prototype, jadi bisa jadi memiliki default key ketika pertama dibuat
Key bisa tipe data apapun	Key hanya bisa string atau symbol
Jumlah key bisa diketahui dengan mudah dengan attribute size	Tidak bisa diketahui, harus manual menggunakan iterasi
Secara default tidak bisa dikonversi ke JSON	Bisa dikonversi ke JSON secara otomatis

Map Instance Method & Property

Method & Property	Keterangan
size	Panjang Map
clear()	Menghapus semua isi Map
delete(key)	Menghapus data Map berdasarkan key
get(key) : value	Mendapatkan data Map berdasarkan key
has(key) : boolean	Mengecek apakah Map berisi data key
set(key, value)	Mengubah data Map dengan key = value
forEach((key, value) =>)	Melakukan iterasi Map

Kode : Map

```
const map = new Map();
map.set("Name", "Eko Kurniawan");
map.set("Address", "Indonesia");

console.info(map);
console.info(map.get("Name"));
console.info(map.get("Address"));

for (const element of map) {
  console.info(`${element[0]} : ${element[1]}`);
}
```

Set

Set

- Set merupakan implementasi dari struktur data yang berisikan data-data unique
- Set mirip seperti Array, hanya saja isi datanya selalu unique
- Jika kita menambahkan data yang sama, maka data hanya akan diterima satu saja
- Set mengimplementasikan kontrak iterable, sehingga bisa diiterasi secara default
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Set

Set Instance Method & Property

Method & Property	Keterangan
size	Panjang Set
add(value)	Menambah data ke Set
has(value)	Mengecek apakah Set memiliki value
delete(value)	Menghapus value dari Set
forEach(value =>)	Melakukan iterasi Set

Kode : Set

```
const set = new Set();

set.add("Eko");
set.add("Eko");
set.add("Kurniawan");
set.add("Khannedy");

console.info(set);
set.forEach(value => console.info(value));
```

Symbol

Symbol

- Symbol merupakan tipe data yang digaransi akan selalu unique setiap kali kita membuat data symbol
- Symbol kadang banyak digunakan untuk membuat key pada object
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Symbol

Kode : Symbol

```
const firstName = Symbol();
const lastName = Symbol();

const person = {};
person[firstName] = "Eko";
person[lastName] = "Khannedy";

console.info(person);
console.info(person[firstName]);
console.info(person[lastName]);
```

Symbol For

- Kadang agak sulit membuat symbol harus selalu menggunakan variable
- Pembuatan symbol juga bisa menggunakan static method Symbol.for(key)
- Jika kita menggunakan key yang sama, Symbol yang sama akan sekalu dikembalikan

Kode : Symbol For

```
const person = {};
person[Symbol.for("firstName")] = "Eko";
person[Symbol.for("lastName")] = "Khannedy";

console.info(person);
console.info(person[Symbol.for("firstName")]);
console.info(person[Symbol.for("lastName")]);
```

RegExp

RegExp

- RegExp merupakan implementasi dari regular expression di JavaScript
- Regular expression merupakan fitur untuk mencari text dengan pola
- Membuat regular expression di JavaScript bisa dilakukan dengan dua cara, yaitu menggunakan literal notation atau membuat object RegExp
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/RegExp

Membuat RegExp

```
const regex1 = /[a]/
const regex2 = new RegExp("[a]")
const regex3 = new RegExp(/[a]/)
```

RegExp Instance Method

Instance Method	Keterangan
exec(value) : result	Eksekusi regex, jika menemukan data sesuai pola, maka kembalikan result nya, jika tidak maka null
test(value) : boolean	Eksekusi regex, jika menemukan data sesuai pola, maka return true, jika tidak maka false

Kode : RegExp Instance Method

```
const name = "eko kurniawan eko khannedy";
const regex = /eko/

let result = regex.exec(name);
console.info(result);

let test = regex.test(name);
console.info(test);
```

RegExp Modifier

RegExp memiliki modifier untuk mengubah sifat cara pencarian

Modifier	Keterangan
i	Regex menjadi incase sensitive
g	Pencarian dilakukan secara global, secara default setelah menemukan data, pencarian akan berhenti
m	Multiline, pencarian dilakukan di tiap baris (enter)

Kode : RegExp Modifier

```
const name = "eko kurniawan\neKo khannedy\nEdi Kurniawan";
const regex1 = /eko/g
const regex2 = /Kurniawan/ig

let result;
while ((result = regex1.exec(name)) !== null) {
    console.info(result);
}

while ((result = regex2.exec(name)) !== null) {
    console.info(result);
}
```

Fitur RegExp Lainnya

- Assersions : indikasi awal dan akhir teks
  https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions/Assertions
- Character Classes : membedakan antara huruf dan angka
  https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions/Character_Classes
- Group dan Range : melakukan grouping atau range huruf atau angka
  https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions/Groups_and_Ranges
- Quantifiers : menentukan jumlah huruf atau angka
  https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions/Quantifiers

Kode : Contoh Fitur RegExp

```
const regex = /ek[aiueo]/ig;
const text = "eko ado eki eka ejo emi elo eke";

while ((result = regex.exec(text)) !== null) {
    console.info(result);
}
```

Regular Expression di String

Di JavaScript, tipe data String memiliki instance method yang dapat memanfaatkan RegExp untuk melakukan pencarian

String Method	Keterangan
match(regex) : Array	Mencari semua data yang sesuai dengan regex
search(regex) : index	Mencari index data yang sesuai dengan regex
replace(regex, value)	Mengubah data dengan value yang sesuai regex
replaceAll(regex, value)	Mengubah semua data dengan value yang sesuai regex
split(regex) : Array	Memotong string dengan regex

Kode : Regular Expression di String

```
const string = "eko ado eki eka ejo emi elo eke";

console.info(string.match(/ek[aiueo]/ig)); // ["eko", "eki", "eka", "eke"]
console.info(string.search(/ek[aiueo]/ig)); // 0
// di replace semua karena regex
console.info(string.replace(/ek[aiueo]/ig, "keren")); // keren ado keren keren ejo emi elo keren
console.info(string.replaceAll(/ek[aiueo]/ig, "keren")); // keren ado keren keren ejo emi elo keren
console.info(string.split(/e/ig)); // ["", "ko ado ", "ki ", "ka ", "jo ", "mi ", "lo ", "k", ""]
```

Proxy

Proxy

- Proxy merupakan fitur yang bisa digunakan sebagai wakil sebuah data
- Dengan menggunakan proxy, semua interaksi ke data akan selalu melalui Proxy terlebih dahulu
- Dengan ini, kita bisa melakukan apapun sebelum interaksi dilakukan ke data yang dituju
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Proxy

Proxy Handler

- Pembuatan Proxy perlu menggunakan handler, dimana dalam handler, kita bisa membuat function yang dinamakan interceptor yang digunakan ketika mengambil data atau mengubah data ke target
- Untuk membuat Proxy, kita bisa menggunakan new Proxy(target, handler)

Kode : Membuat Handler

```
const target = {}

const handler = {
    get: function (target, property) {
        return target[property];
    },
    set: function (target, property, value) {
        target[property] = value;
    }
}
```

Kode : Membuat Proxy

```
const proxy = new Proxy(target, handler);
proxy.firstName = "Eko";
proxy.lastName = "Khannedy";

console.info(proxy.firstName);
console.info(proxy.lastName);
```

Proxy dan Handler

- Saat kita mengubah data proxy, secara otomatis data akan dikirim ke target melalui handler dengan memanggil function set(target, property, value)
- Saat kita mengambil data proxy, secara otomatis data akan diambil dari target melalui handler dengan memanggil function get(target, property)
- Artinya, jika kita ingin melakukan sesuatu sebelum dan setelah nya, bisa kita lakukan di handler

Kode : Log Handler

```
const handler = {
    get: function (target, property) {
        console.info(`Access property ${property}`);
        return target[property];
    },
    set: function (target, property, value) {
        console.info(`Change property ${property} : ${value}`);
        target[property] = value;
    }
```

Reflect

Reflect

- Reflect merupakan class yang digunakan untuk mengeksekusi JavaScript function
- Reflect tidak memiliki constructor, dan cara penggunaan Reflect tidak dengan membuat object dengan new Reflect
- Penggunaan Reflect adalah menggunakan banyak sekali static methodnya
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect

Kode : Reflect

```
const person = {};

Reflect.set(person, 'firstName', 'Eko')
Reflect.set(person, 'lastName', 'Eko')

console.info(person);

console.info(Reflect.has(person, 'firstName')); // true
console.info(Reflect.has(person, 'middleName')); // false
```

Encode

Encode

- Saat kita menulis URL, kadang kita ingin menambahkan informasi tambahan seperti query parameter misalnya
- URL sendiri sudah memiliki standard encoding penulisan URL
- Standard encoding ini dilakukan agar penulisan URL aman ketika diterima oleh server
- Aman disini dalam artian informasi URL tidak berubah
- Contoh paling sederhana, misal, walaupun kita bisa mengirim spasi dalam url, tapi disarankan untuk di encode agar nilai spasi tidak benar-benar terlihat seperti spasi pada URL nya

Contoh Masalah Jika Tidak Menggunakan Encode

- Misal ada query parameter dengan nama data, lalu kita ingin mengirimkan nilai kesana
- Misal nilainya adalah &eko=eko&, alhasil URL nya akan seperti berikut
- http://contoh.com?data=&eko=eko&
- URL diatas terlihat tidak ada masalah, tapi sebenarnya ketika diterima oleh server, parameter data bernilai kosong, kenapa? karena & dianggap sebagai pemisah antar parameter
- Artinya tanda seperti & dan lain-lain perlu di encode, agar tidak terjadi kesalahan data yang kita kirim

Encode Function

Function	Keterangan
encodeURI(value)	Melakukan encode value, namun karakter yang dipesan di URI tidak akan diubah ;,/?:@&=+$#
encodeURIComponent(value)	Melakukan encode value semua karakter
decodeURI(encoded)	Melakukan decode value hasil encodeURI()
decodeURIComponent(encoded)	Melakukan decode value hasil encodeURIComponent()

Kode : encodeURI dan decodeURI

```
const url = 'http://www.contoh.com/?name=Eko Kurniawan Khannedy';
console.info(url);

const encoded = encodeURI(url);
console.info(encoded);

const decoded = decodeURI(encoded);
console.info(decoded)
```

Kode : encodeURIComponent dan decodeURICompoentn

```
const value = "Eko&Kurniawan=Khannedy;"
const url = 'http://www.contoh.com/?name=';

const encoded = encodeURIComponent(value);
console.info(url + encoded);

const decoded = decodeURIComponent(encoded);
console.info(url + decoded);
```

Base64

Base64

- Base64 merupakan binary to text encoding, representasi binary data dalam format string
- Base64 merupakan format text data yang aman untuk dikirimkan di web
- Base64 merupakan encoding yang biasanya digunakan ketika perlu mengirim data dari client ke server
- Karena encoding Base64 merupakan text, oleh karena itu sangat aman digunakan pada query param URL atau text body dalam form
- https://developer.mozilla.org/en-US/docs/Glossary/Base64

Base64 Function

JavaScript memiliki function bawaan untuk melakukan encode Base64 atau decode base64

Function	Keterangan
btoa(value)	Encode ke base64 dari value
atob(encoded)	Decode dari base64 ke value

Kode : Base64

```
const original = "Eko Kurniawan Khannedy";

const encoded = btoa(original);
console.info(encoded);

const decoded = atob(encoded);
console.info(decoded);
```

Eval

Eval

- Eval merupakan function yang digunakan untuk mengeksekusi kode JavaScript dari String
- Fitur ini sangat menarik, namun perlu hati-hati ketika menggunakannya
- Jika sampai salah penggunaan, maka bisa jadi kita malah mengeksekusi kode program yang bisa menyebabkan masalah keamanan di website kita
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval

Kode : Eval

```
const script = "alert('Ups, Anda di Heck')";

eval(script);
```

Materi Selanjutnya

Materi Selanjutnya

- JavaScript Modules
- JavaScript Document Object Model
- JavaScript Async
- JavaScript Web API