**Object Prototyping**

Konsep objek dalam JavaScript dapat dipahami dalam kehidupan sehari-hari seperti sebuah barang yang memiliki properti. Kita akan jadikan piring sebagai contoh objek. Sebuah piring adalah objek dengan banyak properti. Piring memiliki warna, desain, berat, bahan, dan lainnya. Dengan cara sama, objek JavaScript juga punya banyak properti yang menjelaskan karakteristiknya.

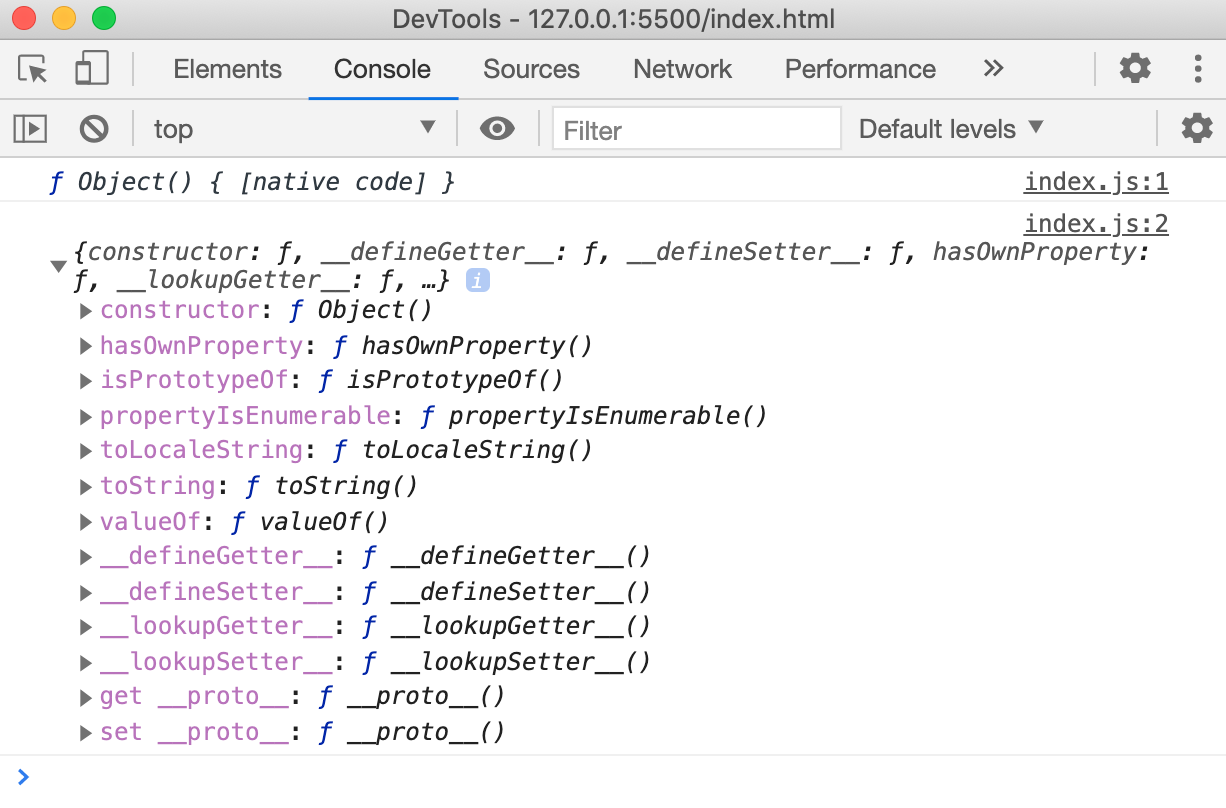
Di dalam Object.prototype JavaScript *engine* menyediakan banyak sekali *function* atau *built-in properties* seperti toString(), valueOf(), dan masih banyak lagi.

Berikut adalah hasil dari *prototype object* jika dilakukan console.log():

console.log(Object);

console.log(Object.prototype);

Hasil dari console.log kode di atas adalah sebagai berikut:



Setiap *function* atau *built-in properties* sudah memiliki fungsi dasar yang sering dipakai, lalu kita bisa menambahkan properti baru ke dalam objek jika dibutuhkan. Namun, kita tidak bisa menambahkan properti dari objek secara langsung.

**Menambahkan objek *prototyping***

Contoh struktur kode dasar untuk menambahkan properti pada sebuah objek menggunakan *prototype*:

// Mendeklarasikan fungsi baru

function NamaObjek(parameter1, parameter2) {

this.namaProperti1 = parameter1;

this.namaProperti2 = parameter2;

}

// Menambahkan properti secara langsung menggunakan prototyping

NamaObjek.prototype.namaProperti = "isi properti";

// Melihat hasil data

console.log(namaObject.prototype.namaProperti); // "isi properti"

Berikut ini adalah contoh sebuah kasus menambahkan properti dalam sebuah objek Person:

function Person(name, age, hobby, job) {

this.name = name;

this.age = age;

this.hobby = hobby;

this.job = job;

}

// Menambahkan properti secara langsung menggunakan prototyping

Person.prototype.nationality = "Indonesia";

// Buat instance ke dua

let myFather = new Person("William", 30, "Play games", "Entrepreneur");

let myMother = new Person("Jessica", 30, "Reading book", "Accountant");

// Melihat hasil data

console.log(myFather.name); // Output: 'William'

console.log(myFather.nationality); // Output: 'Indonesia'

console.log(myMother.name); // Output: 'Jessica'

console.log(myMother.nationality); // Output: 'Indonesia'

**Menambahkan *method/function prototyping***

Kita juga bisa menambahkan *method* untuk *constructor* yang telah kita buat. Sehingga *method* akan bisa digunakan untuk semua *instance* objek.

Contoh struktur kode dasar untuk menambahkan *method/function prototyping* :

function NamaObjek(parameter1, parameter2) {

this.namaProperti1 = parameter1;

this.namaProperti2 = parameter2;

}

// Menambahkan method

NamaObjek.prototype.namaMethod = function () {

return "hasil dari fungsi";

};

Berikut adalah contoh sebuah kasus menambahkan *method/function prototyping* dalam sebuah objek Person:

function Person(name, age, hobby, job) {

this.name = name;

this.age = age;

this.hobby = hobby;

this.job = job;

}

// Menambahkan properti secara langsung menggunakan prototyping

Person.prototype.nationality = "Indonesia";

// Menambahkan method menggunakan prototyping

Person.prototype.greeting = function () {

return "Hello " + this.name;

};

// Membuat instance ke dua

let myFather = new Person("William", 30, "Play games", "Entrepreneur");

let myMother = new Person("Jessica", 30, "Reading book", "Accountant");

// Melihat hasil data

console.log(myFather.name); // Output: 'William'

console.log(myFather.nationality); // Output: 'Indonesia'

console.log(myFather.greeting()); // Output: 'Hello William'

Saat ini kamu sudah berhasil menambahkan properti dan *method* baru pada objek menggunakan *prototype* dari JavaScript.

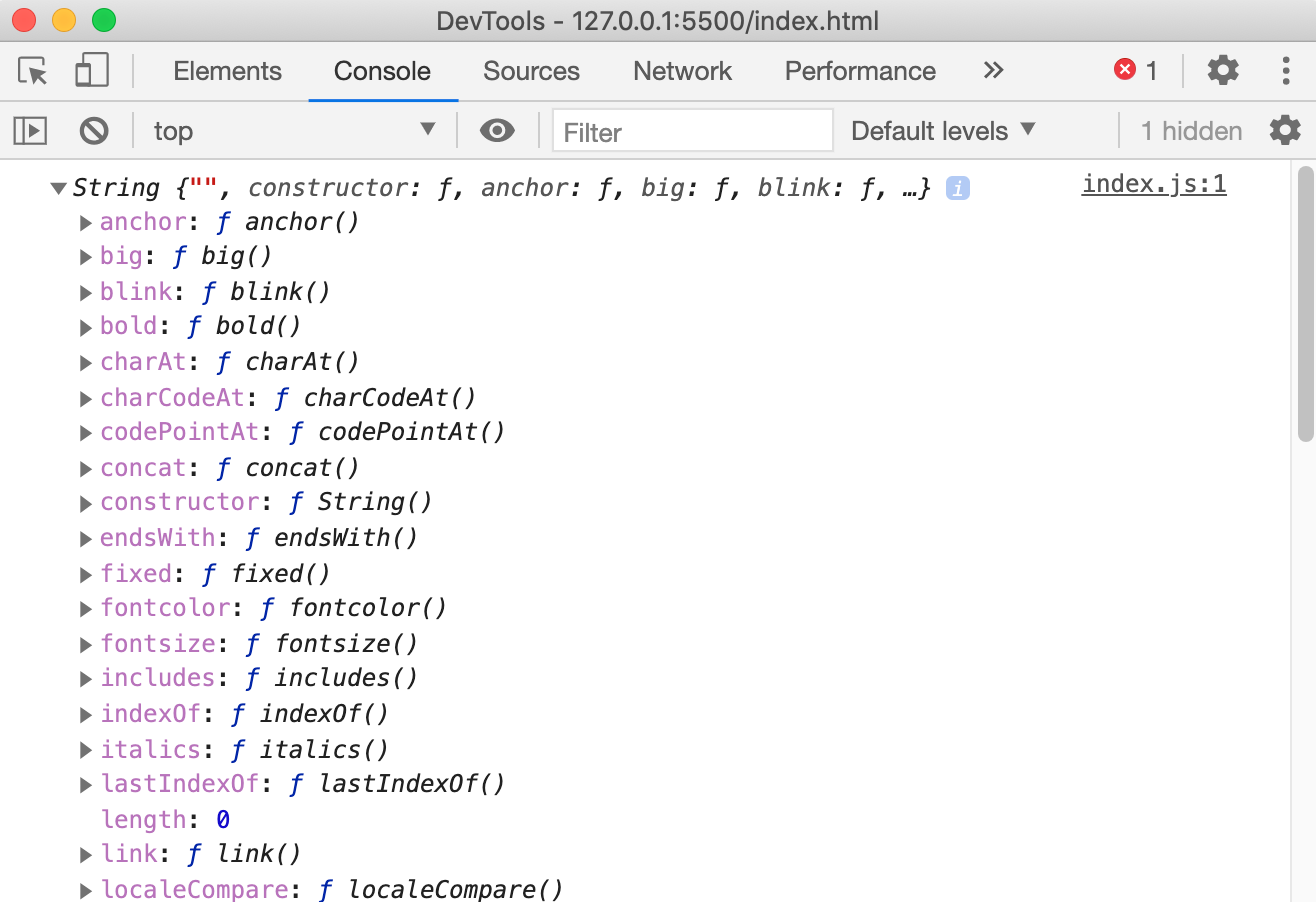
# Data Type Prototyping

Setiap tipe data di JavaScript memiliki prototype bawaannya masing-masing berupa property dan method.

Berikut ini adalah hasil dari prototype tipe data String jika dilakukan console.log():

// Digunakan untuk melihat properti apa saja yang dimiliki oleh prototype String

console.log(String.prototype);



Jika dari list tersebut kamu tidak menemukan method built-in atau property yang kamu butuhkan, maka ini saatnya untuk kamu membuat property tambahan atau custom pada tipe data string menggunakan prototype. Tidak hanya pada tipe data string, tapi kamu juga bisa menambahkannya pada semua tipe data.

dataType.prototype.namaMethod = function(n){

// kode program

}

Agar dapat memahami dengan baik, mari kita perhatikan contoh berikut:

Tipe data string memiliki prototype method repeat(). Sekarang kita akan membuat method baru dengan menggunakan method repeat() :

// Kita membuat properti tambahan untuk String

// Method untuk mengulang string selama yang kita butuhkan

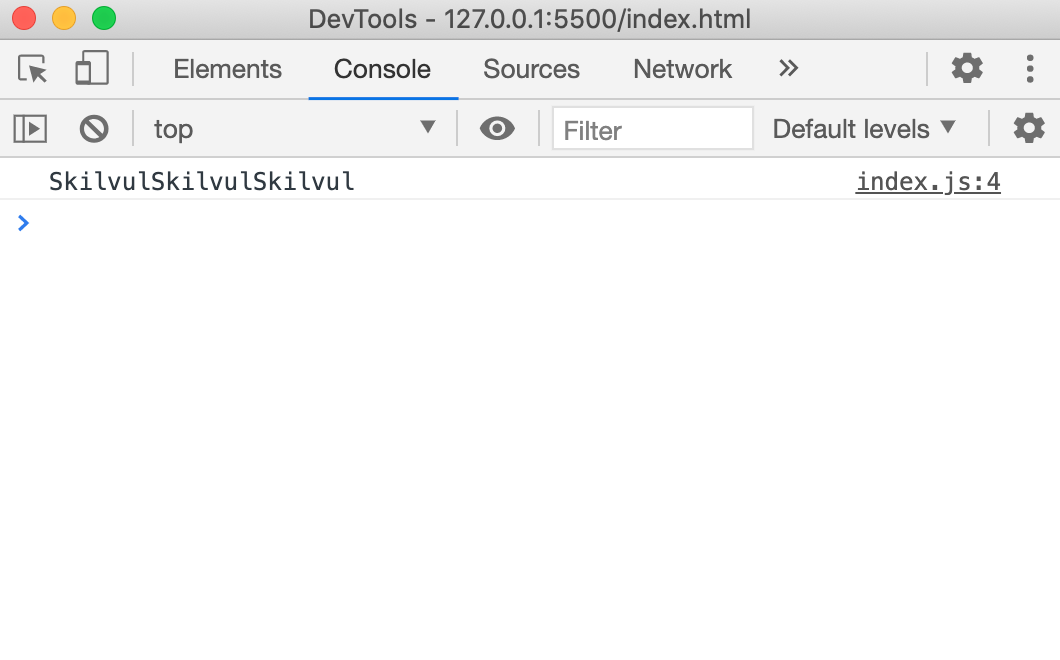
String.prototype.repeatString = function (n) {

return this.repeat(n);

};

console.log("Skilvul".repeatString(3));

Berikut hasil console.log() saat memanggil method repeatString() yang baru saja kita buat pada tipe data string :



Jika kita cek kembali dengan melakukan console.log() pada prototype string, maka hasilnya method yang kita tambahkan sudah terdaftar pada prototype tipe data string seperti berikut ini :

