* (object literals) chỉ đơn giản là khai báo một object trong JavaScript
* var birthYear = 2000
* var obj = {
* name: 'John',
* birthYear: birthYear,
* getAge: function(currentYear) {
* return currentYear - obj.birthYear
* }
* }
* ES6 nâng cấp object chân phương, cho phép bạn khai báo tắt thuộc tính của object với biến cùng tên, và khai báo phương thức cho object.
* const birthYear = 2000
* const obj = {
* name: 'John',
* birthYear, // khai báo tắt birthYear: birthYear
* getAge(currentYear) {
* // `this` được gán trực tiếp vào bản thân object
* return currentYear - this.birthYear
* }
* }
* Một lưu ý với this là khi bạn dùng hàm mũi tên, this sẽ được lấy từ this trong tầm vực gần với nó nhất, chứ không trỏ đến đối tượng hiện tại. Do đó…
* {
* getAge: currentYear => currentYear - this.birthYear
* }
* …sẽ không chạy như mong muốn, vì có thể this.birthYear mang giá trị undefined. Để sử dụng hàm mũi tên bạn phải viết lại như trước ES6.
* {
* getAge: currentYear => currentYear - obj.birthYear
* }
* Ngoài ra từ ES6 bạn cũng có thể khai báo thuộc tính cho object một cách linh động bằng cách sử dụng cú pháp [].
* const attr = 'foo'
* const year = 2017
* const obj = { [attr]: 1, ['now' + year]: 'wow' }
* console.log(obj) // { foo: 1, now2017: 'wow' }

Map: là một kiểu dữ liệu tương tự như Set, tuy nhiên với Map thì có cấu trúc dạng key => value còn với Set thì chỉ có value.

### Khởi tạo:

// Tạo một map gồm 3 thông tin

 let map = new Map([

     ["Name", "Nguyen Van Cuong"],

     ["Email", "thehalfheart@gmail.com"],

     ["Website", "freetuts.net"]

]);

console.log(map);

// Map {"Name" => "Nguyen Van Cuong", "Email" => "thehalfheart@gmail.com", "Website" => "freetuts.net"}

### Thêm phần tử:

|  |  |
| --- | --- |
|  | let map = new Map();    map.set('Name', 'Nguyen Van Cuong');    console.log(map);  // Map {"Name" => "Nguyen Van Cuong"} |

### Xóa phần tử:

|  |  |
| --- | --- |
|  | let map = new Map();    map.set('Name', 'Nguyen Van Cuong');    console.log(map);  // Map {"Name" => "Nguyen Van Cuong"}    map.delete("Name");    console.log(map);  // Map {} |

### Kiểm tra phần tử tồn tại:

|  |  |
| --- | --- |
|  | let map = new Map();    map.set('Name', 'Nguyen Van Cuong');    console.log(map.has('domain')); // False  //console.log(map.has('Name')); // True |

### Đếm tổng số phần tử:

|  |  |
| --- | --- |
|  | let map = new Map();    map.set('Name', 'Nguyen Van Cuong');    console.log(map.size); // 1 |

### Xóa toàn bộ phần tử:

|  |  |
| --- | --- |
|  | let map = new Map();    map.set('Name', 'Nguyen Van Cuong');  map.set('Domain', 'freetuts.net');    map.clear();    console.log(map.size); // 0 |

Set :

Là tập hợp các giá trị không bị trùng lặp, nghĩa là trong một set không thể có hai giá trị bằng nhau.

Nếu có 2 giá trị giống nhau trong set thì chỉ có 1 giá trị được lưu.

Thao tác cơ bản

* Khởi tạo: let set = new Set();

let blankSet = new Set();

let blankSet = new Set(['a', 'b', 'c', 'd', 1, 2, 3]);

* Thêm phần tử: set.add(value);

blankSet.add('e');

* Xóa phần tử: set.delete(value);

blankSet.delete('e');

* Kiểm tra tồn tại giá trị: set.has(value);

blankSet.has('c'); // true

blankSet.has('f'); // false

* Đếm tổng số phần tử: set.size;

blankSet.size;//7

* Xóa toàn bộ phần tử: set.clear();

blankSet.clear();

//Set(0) {}

* Vòng lặp với Set

Để lặp qua các phần tử ta dùng cú pháp **for...of**

let numbers = new Set([1, 2, 3, 4]);

for (let number of numbers){

    console.log(number);

}

// Output:

// 1

// 2

// 3

// 4

* Chuyển đổi Set sang Array

Để chuyển đổi Set sang Array thì bạn sử dụng ba dấu chấm đặt trước biến Set.

let numbers = new Set([1, 2, 3, 4]);

let arr\_numbers = [...numbers];

// Lúc này  biến arr\_numbers sẽ là một mảng gồm 4 phần tử

// [1, 2, 3, 4]