Tìm hiểu về JS

1. **comment**

// co dau ; cos the viet lien duoc neu ko co thi viet xuong dong

1. Mục đích sử dụng comment:

- Ghi chú

- Vô hiệu hoá mã

2. Loại comment

- comment 1 dòng như trên

- comment nh dòng đang xài

3. phím tắt

- win : ctrl + /

1. **Các hàm build-in function:**

1. alert

2. console

3. confirm

4. prompt

5. set timeout

6. set interval

Vd:

// console.log('Day la dong log')

// console.log(fullName)

// console.error(fullName)

//confirm('xac nhan ban du tuoi: ');

// prompt('Xac nhan ban du tuoi')

// setTimeout( function() {

// alert('ban khong du do tuoi');

// }, 1000)

// timeout : hang 1 lan xai

// setInterval(function() {

// console.log('day la day so:' + Math.random());

// }, 1000)

1. **các toán tử thêm so với những toán tử cũ bên c**

1.\*\* : luỹ thừa

2.toán tử ++ và -- có cả tiền tố (prefix) và hậu tố (postfix)

\*/

1. **Các kiểu dữ liệu trong JS**

// 1. Dữ liệu nguyên thuỷ

// - Number

// - String

// - Boolean

// - Undefined

// - Null

// - Symbol

// 2. Dữ liệu phức tạp

// - Function

// - Object

// một vài vd:

// var vd = 'Nam \'thanh';

// console.log(vd)

// var myObject = {

// name :'Nam thanh',

// age : 20,

// address: 'Ha Noi'

// // cần định nghĩa key

// };

// console.log('myObject',myObject);

// var myArray = [

// 'Javascript',

// 'PHP',

// 'F2P'

// ];

// console.log(myArray)

// var a = 1;

// var isNull = null;

// //kiểm tra loại dữ liệu

// console.log(typeof fullName);

// console.log(typeof isNull);

// var isNaN = NaN;

// console.log(isNaN);

/\*

1. **toán tử ms**

- === -> so sánh cả value và cả datatype

- !== -> ngược lại cả ông trên

\*/

//vd:

1. **Chuỗi trong Javascript**

// 1. Tạo chuỗi 2 cách:

// - var Name = 'nam thanh';

// - var Name = new String('nam thanh');

// nên sd cách 1 vì cách 2 dài và biến đổi loại dữ liệu từ string sang object

//cách sd backlash

//console.log(fullName.length);

//

//một số template string ES6

//console.log(`Tôi là : ${fullName}`)

//---------------------------------------------------

/\*

1. **Mảng trong javascript - array**

1. Tạo mảng

- cách tạo

- Sử dụng cách nào? Tại sao?

- Kiểm tra dât type?

2. Truy xuất mảng

- Độ dài mảng

- Lấy phần tử theo index

\*/

// thay vì toạ nhiều biến -> sử duungj mảng

//tips : ctrl c cuối dòng để copy nhanh hơn mà ko cần bôi đen

//P1: Tạo mảng

// c1:

// var languages = [

// 'Javascript',

// 'PHP',

// 'Ruby',

// 'Dart',

// null,

// undefined,

// function(){

// },

// {},

// 123,

// ];

// console.log(languages)

//------------------------------------

//c2:

// var languages = new Array(

// 'Javascript',

// 'PHP',

// 'Ruby',

// 'Dart',

// null,

// undefined,

// function(){

// },

// {},

// 123,

// );

//tips: kiểm tra array hay ko:

//console.log(Array.isArray(...)) với ... là thứ cần kiểm tra.

//-------------------------------------

//**P2 : Truy xuất mảng**

// index, key ...

// var languages = [

// 'Javascript',

// 'PHP',

// 'Ruby',

// 'Dart',

// ];

// console.log(languages.length)

// ->4

// console.log(languages[2])

//-> Ruby

//--------------------------------------------

//--------------------------------------------

1. // **Toán tử 3 ngôi**

// var course = {

// name: 'Javascript',

// coin : 250

// }

// vd: cần chuyển đổi 2 ngôi :

// if (course.coin > 0){

// console.log (`${course.coin} coins`);

// } else {

// console.log('Miễn phí');

// }

// chuyển đổi sang 3 ngôi

// var result = course.coin > 0 ? `${course.coin} Coins` : 'Miễn phí';

// console.log(result);

//-------------------------------------------

/\*

1. **Vòng lặp - Loop**

1.for - lặp với điều kiện đúng

2.for/in - lặp qua key của đối tượng

3.for/of - lặp qua value của đối tượng

4.while - Lặp khi điều kiện đúng

5.do/while - Lặp ít nhất 1 lần, sau đó lặp khi điều kiện đúng

\*/

// vd:

// for loop :

// for (var i = 1; i<=1000; i++){

// //code

// }

//-> code chạy 1 nghìn lần

// có thể khai báo ngoài cho biến chạy trong for

/\* vd:

var myArray = [

'Javascript',

'PHP',

'Java',

'Daft'

];

var arrayLength = myArray.length;

for(var i = 0; i< arrayLength; i++){

console.log(myArray[i]);

}

\*/

// for/in loop

//vd : vs object

/\* var myInfo = {

name: 'Thanh Nam',

age: 20,

address : 'Hanoi, VN'

};

for (var key in myInfo) {

console.log(myInfo[key]);

} \*/

//vd : vs string

/\* var myString = 'Javascript';

var languages = 'Javascript'

for (var key in languages) {

console.log(languages[key]);

}

-> tach chu

\*/

//for/of loop

/\*var myInfo = {

name: 'Thanh Nam',

age: 20

};

for(var value of Object.keys(myInfo)) {

console.log(myInfo[value]);

}

\*/

//----------------------------------------------

/\*

**Array methods:**

**forEach()**

**every()**

**some()**

**find()**

**filter()**

**map()**

**reduce()**

\*/

var courses = [

{

id: 1,

name: 'Javascript',

coin: 0

},

{

id: 2,

name: 'HTML,CSS',

coin: 1

},

{

id: 3,

name: 'Ruby',

coin: 1

},

{

id: 4,

name: 'Python',

coin: 1

},

{

id: 5,

name: 'Daft',

coin: 200

},

];

function courseHandler(course, index, originArray) {

//console.log(course);

return {

id: course.id,

name: `Khoa hoc: ${course.name}`,

coin: course.coin,

coinText: `Gia ${course.coin}`,

index: index,

originArray: originArray,

};

}

var newCourses = courses.map(courseHandler);

console.log(newCourses);

Hàm map() trong Javascript là một hàm dành cho đối tượng array,

hàm này có công dụng tương tự như vòng lăp forEach.

map được sử dụng với mục đích return những tham số sau khi cải biên = return

vd những vị bác sĩ ở iSoftcare cũng sẽ được định dạng dưới dạng Array thì

return `<h2>${doctor.name}</h2>`;

...

để liên kết với html, css \*/

//--------------------------

/\* tìm hiểu khái niệm callback

là hàm (function) được truyền qua đối số khi gọi hàm khác

1.là hàm

2. được truyền qua đối số

3. được gọi lại (trong hàm nhận đối số)

function myFunction(param){

//console.log(typeof param)

if (typeof param === 'function'){

param('Thanh nam');

}

}

function myCallback(){

console.log('Value', value);

}

myFunction(myCallback);

// ->day la mot ham callback

// chu y phai kiem tra de xem du lieu co cung kieu hay khong?

\*/

//----------------

//Callback 2

// Array.prototype.map2 = function(callback) {

// console.log(this);

// var output = [], arrayLength = this.length;

// for(var i = 0; i < arrayLength; i++ ){

// var result = callback(this[i], i);

// output.push(result);

// }

// return output;

// }

// // tren la base cua map

// var courses = [

// 'Javascript',

// 'PHP',

// 'Ruby'

// ]

// var htmls = courses.map2(function (course){

// return `<h2>${course}</h2>`;

// });

// console.log(htmls.join(''));

// tren la tu viet mot ham map2 co chac nang giong ham map de hieu hon

//------

// var htmls = course.map2(function(course){

// return `<h2>${course}</h2>`;

// });

// console.log(htmls.join(''));

//------------------------------------

// forEach

//khái niệm :

/\*Hàm forEach sẽ lần lượt truyền các phần tử của mảng vào một hàm được cung câp trước đó,

hàm này sẽ tiến hành xử lý các thành phần tử cửa mảng như là một tham số truyền vào.

Hàm forEach sẽ không thức hiện hàm cung cấp đối với những phần tử không có giá trị.\*/

Hiểu đơn giản là : phương pháp này giúp lặp các item qua mảng

// day la ham forEach 2 viet lai de hieu ban chat

// hasOwnProperty la 1 ham de kiem tra tinh dung sai do ke thua

// Array.prototype.forEach2 = function(callback) {

// for (var index in this) {

// if (this.hasOwnProperty(index)){

// callback(this[index], index, this);

// }

// }

// };

// var courses = new Array(100);

// courses.push('JS','Ruby');

// courses.forEach2(function (course,index,array){

// console.log(course, index, array);

// });

//----------------------------------------------------------------------

/\*

filter()

Hàm filter trong Javascript có công dụng đúng ý nghĩa với tên gọi của nó.

Hàm filter được tích hơp sẵn trong đối tượng mảng giống như hàm map.

Về cách thức hoạt động thì nó giống hàm map,

nghĩa là nó sẽ có một tham số dạng function và function này sẽ xử lý cho mỗi lần lặp,

nếu function return true thì phần tử lần lặp đó được chấp nhận,

ngược lại sẽ không được chấp nhận.

lưu ý : không lặp qua array trống

\*/

// var courses = [

// {

// id: 1,

// name: 'Javascript',

// coin: 0

// },

// {

// id: 2,

// name: 'HTML,CSS',

// coin: 1

// },

// {

// id: 3,

// name: 'Ruby',

// coin: 1

// },

// {

// id: 4,

// name: 'Python',

// coin: 1

// },

// {

// id: 5,

// name: 'Daft',

// coin: 200

// },

// ];

// var filterCourses = courses.filter(function (course,index,coin) {

// return course.coin > 1;

// })

// console.log(filterCourses);

//viet lai ham filter2

// Array.prototype.filter2 = function(callback) {

// var output = [];

// for( var index in this){

// if (this.hasOwnProperty(index)){

// var result = callback(this[index], index);

// if (result) {

// output.push(this[index]);

// }

// }

// }

// return output;

// };

// var courses = [

// {

// id: 1,

// name: 'Javascript',

// coin: 0

// },

// {

// id: 2,

// name: 'HTML,CSS',

// coin: 1

// },

// {

// id: 3,

// name: 'Ruby',

// coin: 1

// },

// {

// id: 4,

// name: 'Python',

// coin: 1

// },

// {

// id: 5,

// name: 'Daft',

// coin: 200

// },

// ];

// var filterCourses = courses.filter(function (course,index,coin) {

// return course.coin > 1;

// })

// console.log(filterCourses);

//-------------------------------------------------------------------------

// find()

/\*Hàm find sẽ trả về giá trị của phần tử đầu tiên trong mảng thỏa mãn được điều kiện kiểm tra( được truyền vào như một hàm!).

Lưu ý:

Hàm find không thực hiện hàm kiểm tra với các phần tử không có giá trị.

Hàm find sẽ không thay đổi mảng ban đầu.

\*/

// viet lai ham find

Array.prototype.find2 = function(callback) {

var output;

for (var i = 0; i < this.length; ++i){

var result = callback(this[i], i, this);

if (result) {

output = this [i];

break;

}

}

return output;

};

var courses = [

{

id: 1,

name: 'Javascript',

coin: 0

},

{

id: 2,

name: 'HTML,CSS',

coin: 1

},

{

id: 3,

name: 'Ruby',

coin: 1

},

{

id: 4,

name: 'Python',

coin: 1

},

{

id: 5,

name: 'Daft',

coin: 200

},

];

var course = courses.find2(function (course,index,coin) {

return course.coin > 2;

})

console.log(course);

phân biệt ách sử dụng map(), filter(), reduce()

khi nào sử dụng map : nếu chúng ta muốn thực hiện cùng một thao tác chuyển đổi trên từng phân tử của array và lấy lại array mới có cùng length với các value được chuyển đổi, như ví dụ trên.

Filter đơn giản là xoá các mục khôgn thoả mãn điều kiện.

Reduce….