***Member: Nguyễn Thị Bích Ni***

**JS1**

1. **JS là gì, các thuật ngữ liên quan**

* **JavaScript là một ngôn ngữ kịch bản của WEB (script language), được chạy bởi công cụ JavaScript của trình duyệt web.**
* Mã lệnh của JS được **tích hợp và nhúng trong HTML.**
* Hoạt động trên đa trình duyệt và đa thiết bị;
* Đặc tả ECMAScript cung cấp các tiêu chuẩn mà JavaScript thực hiện.
* JS là ngôn ngữ bất đồng bộ.

1. **Các cách output dữ liệu**

* **Inner:** document.getElementById("demo").innerHTML = 5 + 6;
* **Write:** document.write(5 + 6);
* **Alert:** window.alert(5 + 6);
* **Console.log:** console.log(5 + 6);
* **Print:** <button onclick="window.print()">Print this page</button>

1. **Comment JS**

// comment…

1. **Biến, kiểu dữ liệu**

* ***Biến: Var, let, Const***
  + **Var/Let, Const** khác nhau về Scope, Hoisting. Var được hỗ trợ Hoisting
  + **Let/Const (ES6) :** chỉ có phạm vi trong các khối code như: if else, loop, { }…
  + **Const:** Khi định nghĩa biến và không gán lại biến đó.
* ***Kiểu dữ liệu:***

1. ***Kiểu dữ liệu nguyên thuỷ***

+ Number: var length = 16;

+ String: var lastName = "Ni";

+ Boolean: var isSuccess = true;

+ Underfine: var x;

+ Null: var isNull = null;

+ Symbol: var id = Symbol('id') // unique

1. ***Kiểu dữ liệu phức tạp***

+ Function: var x = function(){...}

+ Object: var x = {firstName:"John", lastName:"Doe"};

1. **Phép gán**

= , += , -= , \*= , /= , %=

*Ví dụ: x\*=y <=> x=x\*y*

1. **Toán tử**

* **Toán tử số học:**

*+ 🡪 Cộng;*

*- 🡪 Trừ;*

*\* 🡪 Nhân;*

*\*\* 🡪 Luỹ thừa;*

*/ 🡪 Chia;*

*% 🡪 Chia lấy số dư;*

*++ 🡪 Tăng 1 giá trị số;* var a = 5; a++ //6

*- - 🡪 Giảm 1 giá trị số;*

***\* Toán tử ++ -- với tiền tố và hậu tố***

* **Tiền tố:**

Var a = 6;

Var output = ++a //7

* + - **Giải thích:** Việc 1: +1 cho a, a = a + 1 => a = 7;

Việc 2: Trả về a sau khi a + 1;

* **Hậu tố:**

Var a = 6;

Var output = a++ //6

Console.log(a) //7

* + - **Giải thích:** Việc 1: `a copy`, `a copy` = 6

Việc 2: Cộng 1 của a, a = a + 1, => a = 7;

Việc 3: Trả về `a copy`

1. **Function**

* **Hàm?**

+ Là khối mã

+ Làm một công việc cụ thể

* **Loại hàm?**

+ Built-in

+ Tự định nghĩa

* Declare Function:

*function showMessage(){*

*Console.log(“Declare”);*

*}*

* Express Function:

*var showMessage2 = function(){*

*Console.log(“Express”);*

*}*

🡪 *Declare có thể gọi trước khi định nghĩa.*

* **Tính chất**

+ Không thực thi khi định nghĩa

+ Sẽ thực thi khi được gọi

+ Có thể nhận tham số

+ Có thể trả về 1 giá trị

* **Arguments**
* function write(){
* var myString = "";
* for(var params of *arguments*){
* myString += `${params} - `;
* }
* console.log(myString);
* }
* write("Log1","Log2","Log3") // Log1 - Log2 - Log3 -

1. **Lệnh điều kiện**

* ***6 giá trị trả về là false***

+ 0

+ false

+ ‘ ‘ – “ “

+ undefined

+ NaN

+ null

🡪 còn lại sẽ là True

* ***Lệnh rẽ nhánh Switch***

*Switch(a){*

*Case 1: console.log(“So 1”);*

*Break;*

*Case 2: console.log(“So 2”);*

*Break;*

*Default: console.log(“Khong ton tai”);*

*}*

1. **Vòng lặp**

* For: Lặp với điều kiện đúng
* For/in: Lặp qua key của đối tượng
* For/of: Lặp qua value của đối tượng
* While: Lặp khi điều kiện đúng
* Do/while: Lặp ít nhất 1 lần, sau đó lặp khi điều kiện đúng

1. **String và các method liên quan**

|  |  |
| --- | --- |
| Code | Result |
| \' | **‘** |
| \" | **“** |
| \\ | **\** |
| \b | Backspace |
| \f | Form Feed |
| \n | New Line |
| \t | Horizontal Tabulator |
| \v | Vertical Tabulator |
| \r | Carriage Return |

* **Method**

Var myString = “Ni Nguyen”

**+ Length**: console.log(myString.length) //9

+ **Find index**: console.log(myString.indexOf(a)) //-1

console.log(myString.indexOf(N,1)) //3

console.log(myString.lastIndexOf(n)) //8

console.log(myString.search(n)) //chỉ truyền được 1 đối số

**+ Cut String:**

* slice(*start*, *end*)
* slice(*start)*
* substring(*start*, *end*)
* substr(*start*, *length*)

**+ Replace:**

var txt = str.replace("Microsoft","W3Schools");

**+ Split**

var str = "a,b,c,d,e,f";

var arr = str.split(",");

document.getElementById("demo").innerHTML = arr[0]; //a

+ **Concat**

var text1 = "Hello";  
var text2 = "World";  
var text3 = text1.concat(" ", text2); //Hello World

**+ Trim**

var str = "       Hello World!        ";  
console.log(str.trim()); // Hello World!

**+ CharAt:**

var str = "HELLO WORLD";  
str.charAt(0);            // returns H

**+ Upper and Lower Case**

var text1 = "Hello World!";       // String  
var text2 = text1.toUpperCase(); // HELLO WORLD!

1. **Number và các method liên quan**

* ***isNaN()***

var x = 100 / "Apple";  // x will be NaN (Not a Number)

* toFixed()

var x = 9.656;  
x.toFixed(0); //9

* toString()

var x = 9.656;  
x.toString(); // “9.656”

* The Number() method

Number(true);          // returns 1  
Number(false);         // returns 0  
Number("10");          // returns 10  
Number(" 10  ");       // returns 10  
Number("10.33");       // returns 10.33  
Number("10,33");       // returns NaN

* The parseInt() method

parseInt("10");         // returns 10  
parseInt("10.33");      // returns 10  
parseInt("10 20 30");   // returns 10  
parseInt("10 years");   // returns 10  
parseInt("years 10");   // returns NaN

* The parseFloat() method

parseFloat("10");        // returns 10  
parseFloat("10.33");     // returns 10.33  
parseFloat("10 years");  // returns 10  
parseFloat("years 10");  // returns NaN

1. **Array và các method liên quan**

* **Tạo mảng:**

var points = new Array(40, 100, 1, 5, 25, 10); // Bad  
var points = [40, 100, 1, 5, 25, 10];          // Good

* **Kiểm tra Array:**

Console.log(Array.isArray(a))

* **Method:**

**+ ToString**

Console.log(array.toString());

Console.log(array.join(‘’));

+ **Pop:** xoá phần tử cuối mảng, return phần tử đã xoá

+ **Push:** thêm 1 or nhiều phần tử vào cuối mảng, return mảng mới.

Console.log(array.push(“ni”,”na”)

+ **Shift:** xoá phần tử đầu mảng, return phần tử đã xoá

+ **Unshift:** thêm 1 or nhiều phần tử vào đầu mảng, return mảng mới.

Console.log(array.push(“ni”,”na”)

**+ Splice:** arr.splice(vị trí, số ptử xoá, giá trị thêm vào)

**+ Concat:** arr.concat(arr2)

**+ Slice:** cắt lấy 1 vài element

arr.slice(start, end)

1. **Date và các method liên quan**

Var date = new Date();

* Get Method

Var year = date.getFullYear();

Var month = date.getMonth(); //0-11

Var hour = date.getHours();

* Set Method

Var setDate = date.setDate(15);

Var setHour = date.setHours(0);

1. **Object Math**

* Math.PI
* Math.round(4.9); làm tròn số gần nhất    // returns 5
* Math.ceil(4.4); làm tròn lên    // returns 5
* Math.floor(4.9); làm tròn xuống   // returns 4
* Math.abs(-4.7);     // returns 4.7
* Math.random();  return số thập phân nhỏ hơn 1
* Math.min(0, 150, 30, 20, -8, -200);  // returns -200
* Math.max(0, 150, 30, 20, -8, -200);  // returns 150

1. **Boolean và các toán tử Logic, So sánh**

**Toán tử so sánh:** == ; != ; > ; < ; >= ; <= ; === ; !==

**Kiểu dữ liệu boolean:** var isSuccess = true;

**Toán tử Logical:** && ; || ; !

**InnerText và textContent**

* <h1 *id*="heading">
* <span>New</span>
* <span>Heading</span>
* </h1>
* <script>
* var headingNode = document.getElementById("heading");
* headingNode.innerText = "New Heading"
* console.log(headingNode.innerText); // New Heading
* console.log(headingNode.textContent) //
* New
* Heading
* //
* </script>

**+ InnerText:** Chỉ lấy nội dung text

**+ textContent:** Lấy toàn bộ khoảng trắng

**JS3**

1. **XML và các thuộc tính liên quan**

* **XML** là từ viết tắt của từ **Extensible Markup Language** là ngôn ngữ đánh dấu mở rộng. XML có chức năng truyền dữ liệu và mô tả nhiều loại dữ liệu khác nhau. Tác dụng chính của XML là đơn giản hóa việc chia sẻ dữ liệu giữa các nền tảng và các hệ thống được kết nối thông qua mạng Internet.
* <!-- This is a comment -->
* Khoảng trắng không được bảo toàn trong XML

1. **JSON và các thuộc tính liên quan**

* JSON stands for JavaScript Object Notation
* JSON is a text format for storing and transporting data
* JSON is "self-describing" and easy to understand
* Converting JSON strings into JavaScript objects: JSON.parse()
* Converting an object into a JSON string: JSON.stringify()

In **JSON**, *values* must be one of the following data types:

* a string
* a number
* an object
* an array
* a boolean
* null

In **JavaScript** values can be all of the above, plus any other valid JavaScript expression, including:

* a function
* a date
* undefined

**JS5**

1. **Window**

* The Browser Object Model (BOM) allows JavaScript to "talk to" the browser.
* BOM (Browser Object Model) là các đối tượng liên quan đến trình duyệt và mỗi một trình duyệt sẽ có những đối tượng khác nhau nên giữa các BOM với nhau nó sẽ không tồn tại một chuẩn chung nào cả.
* Dom cũng là một BOM ( nằm trong BOM window )
* Đối tượng window có tác dụng như với tên của nó là tác động đến của sổ trình duyệt
* Tất cả các đối tượng, hàm và biến JavaScript toàn cục sẽ tự động trở thành thành viên của đối tượng window.
* **Method:**
  + **Lấy width và height của trình duyệt**

let w = window.innerWidth

let h = window.innerHeight

* + **Mở một của sổ mới**

*window.open(url, name, specs, replace);*

*vd: function openW() {*

*window.open('http://ninguyen.com', 'ninguyen','width=500,height=600');*

*}*

* + **Đóng một cửa sổ**

wObject.close();

* + **Di chuyển trình duyệt**

wObject.moveTo(top,left)

* + **Thay đổi kích thước cửa sổ**

wObject.resizeTo(width,height);

1. **Document**

* Đối tượng document trong JavaScript đại diện cho toàn bộ tài liệu HTML.
* Method:

*document.getElementById(id*)

*document.createElement(element)*

*document.removeChild(element)*

*document.replaceChild(new, old)*

*document.anchors*

*document.cookie*

*…*

1. **Screen**

-Đối tượng screen trong javascript chứa các thông số về màn hình của người dùng.

- Cú pháp:

window.screen;

//hoặc

screen;

* Hai thuộc tính này trả về chiều rộng và chiều cao (Đơn vị pixel) của màn hình thiết bị mà người dùng đang sử dụng.
  + screen.width;
  + screen.height;
* Màn hình của trình duyệt:
  + screen.availWidth;
  + screen.availHeight;
* Số lượng màu sắc hiển thị được trên màn hình: *screen.colorDepth;*

1. **Location**

- Đối tượng location là con của đối tượng window và đối tượng location này chuyên dùng để xử lý, điều hướng url của trang web.

* window.location.href returns the href (URL) of the current page
* window.location.hostname returns the domain name of the web host
* window.location.pathname returns the path and filename of the current page
* window.location.protocol returns the web protocol used (http: or https:)
* window.location.assign() loads a new document
* Redirect trang web bằng javascript.

*function redirect(){*

*window.location.href = "http://ninguyen.com";*

*}*

* Phương thức reload(): *window.location.reload();*
* Phương thức replace(): Phương thức này cũng có tác dụng redirect url như thuộc tính **href**, nhưng khi sử dụng **replace**() thì trình duyệt sẽ không lưu vào lịch sử, còn **href** thì có lưu.

*function replace(){*

*window.location.replace('http://ninguyen.com');*

*}*

**JS6**

1. **localStorage**

LocalStorage giúp website lưu trữ dữ liệu vĩnh viễn trên trình duyệt. Trừ khi người dùng xóa cache hoặc làm các hành động như cài lại trình duyệt.

**Syntax:** window.localStorage

**SAVING data to localStorage:** localStorage.setItem("key", "value");

**READING data:** var lastname = localStorage.getItem("key");

**REMOVING data:** localStorage.removeItem("key");

**REMOVING ALL the localStorage:** localStorage.clear();

1. **sessionStorage**

**sessionStorage** khá giống với localStorage. Vì chúng đều thuộc về web storage API.

**Ví dụ:** sessionStorage.setItem('key', 'value');

* + **Sự khác nhau của localStorage và sessionStorage**
* Khi close tab hay close browser thì data ở localStorage vẫn tồn tại, và chỉ bị mất khi user xoá cache hoặc clear web data. Còn đối với sessionStorage thì data sẽ bị mất ngay khi close tab hoặc close browser.
* **Về phạm vi**: sessionStorage: giới hạn trong một cửa sổ hoăc thẻ của trình duyệt. Một trang web được mở trong hai thẻ của cùng một trình duyệt cũng không thể truy xuất dữ liệu lẫn nhau. Như vậy, khi bạn đóng trang web thì dữ liệu lưu trong sessionStorage hiện tại cũng bị xóa. Còn localStorage: có thể truy xuất lẫn nhau giữa các cửa sổ trình duyệt. Dữ liệu sẽ được lưu trữ không giới hạn thời gian.

1. **Cookie**

* **localStorage** chỉ access được trên browser client; còn **cookies** thì có thể access được ở **browser client và cả phía server** (khi tạo một http request thì cookies của browser sẽ được attach vào header Cookie, từ đó phía server có thể parse header này và get được data cookie).
* **cookies** có thời gian hết hạn expires, sau thời gian này thì cookies sẽ biến mất khỏi browser.
* **Cookie** thường được dùng để đăng nhập, lưu vài giá trị cơ bản và đơn giản. Cookie không bị giới hạn bởi độ dài dung lượng hay ký tự.

**Ví dụ hàm set-cookie**

document.cookie = "username=John Doe; expires=Thu, 18 Dec 2013 12:00:00 UTC; path=/";

**JQuery-1**

1. **jQuery là gì, syntax**

* jQuery is a JavaScript Library**.**
* Basic syntax is: **$(*selector*).*action*()**

$(document).ready(function(){  
  $("p").click(function(){  
    $(this).hide();  
  });  
});

1. **Selector**

* Nó dựa trên Bộ chọn CSS hiện có và ngoài ra, nó có một số bộ chọn tùy chỉnh riêng.

Vd: $("\*"), $(this)

* Tất cả các bộ chọn trong jQuery đều bắt đầu bằng ký hiệu đô la và dấu ngoặc đơn: $ ().

1. **Traversing(Lên, xuống, ...)**

* *Traversing up*

parent()

parents()

parentsUntil(): trả về các phần tử giữa 2 đối tượng

$(document).ready(function(){  
  $("span").parentsUntil("div");  
});

* *Traversing down*

children()

find()

* *Sideways*

siblings()

next()

nextAll()

nextUntil()

prev()

prevAll()

prevUntil()

* *Traversing – Filtering*

first()

last()

eq(): the index numbers start at 0

filter():cho phép chỉ định một tiêu chí.

$(document).ready(function(){

$("p").filter(".intro").css("background-color", "yellow");

});

not(): ngược lại với filter

$(document).ready(function(){

$("p").not(".intro").css("background-color", "yellow");

});

1. **Events, on, off và trigger**

* *On*( ): $(*selector*).on(*event,childSelector,data,function,map*)

$( "#dataTable tbody tr" ).on( "click", **function**() {

console.log( $( this ).text() );

});

* *Off* ( ): The off() method is most often used to remove event handlers attached with the on( ) method.

$("button").click(function(){  
  $("p").off("click");  
});

* *Trigger*( ): Thực hiện tất cả các xử lý và đính kèm các loại sự kiện nhất định tới thành phần được chọn.

.trigger(Sự kiện,Tham số)

$('input').trigger(select);

1. **Callback function**

Thực this au khi kết thúc hiệu ứng hiện tại

Typical syntax: $(selector).hide(speed,callback);

1. **Chaining**
2. **Get/set content HTML**
3. **Add, remove, append,... element**
4. **Css method**
5. **Các method liên quan điến width/height, tọa độ**
6. **Xử lý submit và validation form**
7. **Each method**
8. **Animation**