Javascript - Jquery

Javascript

- Javascript là một ngôn ngữ lập trình cho HTML và website
- Javascript là 1 trong 3 ngôn ngữ lập trình web bạn bắt buộc phải học
- Html tạo ra nội dung website
- CSS để tạo ra giao diện và trang trí cho website
- JS được sử dụng để tạo ra các xử lý trên webiste
- JS không chỉ được ứng dụng trong website mà còn được các máy chủ sử dụng như nodejs . Một số cơ sở dữ liệu như mongo db cũng sử dụng javascript ...

Javascript và Java

Javascript và Java là 2 ngôn ngữ lập trình hoàn toán khác biệt

Javascript được phát minh vào năm 1995

Javascript chủ yếu được sử dụng để hoạt động trên các trình duyệt để tạo các xử lý hiệu ứng cho website

Giới thiệu về Javascript

Javascript có thể làm được những gì .

- Javascript có thể thay đổi nội dung của thẻ HTML
- Javascript có thể tạo ra các popup thông báo hay xác nhân
- Javascript có thể thay đổi thuộc tính của thẻ html
- Javascript có thể ẩn/hiện các thẻ html
- Javascript có thể thay đổi CSS của thẻ html
- Javascript có thể tính toán các chức năng trên website

Javascript được viết tại đâu

Trong file html Javascript được viết trong 1 cặp thẻ <script></script>

```
<script>

    document.getElementById("demo").innerHTML = "My
First JavaScript";

</script>
```

Javascript được viết tại 1 file js bên ngoài

Javascript có thể được viết từ 1 file có extension là .js sau đó được dẫn vào file html như sau :

```
<script type="text/javascript"
href="file.js" /></script>
```

Khả năng hiển thị của JS

JS có thể hiển thị dữ liệu theo nhiều cách khác nhau :

- Hiển thị nội dung trong 1 thẻ html bằng cách sử dụng innerHTML
- Hiển thị nội dung trong màn hình console của trình duyệt
- Hiển thị nội dung thông báo qua popup alert()
- Hiển thị nội dung trong file html bằng lệnh document.write()

Sử dụng innerHTML

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My First Paragraph</p>

<p id="demo"></p>

<script>
   document.getElementById("demo").innerHTML = 5 + 6;
</script>

</body>
</html>
```

Sử dụng document.write()

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
   document.write(5 + 6);
</script>

</body>
</html>
```

JS alert()

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
window.alert(5 + 6);
</script>

</body>
</html>
```

JS console.log()

```
<!DOCTYPE html>
<html>
<body>

<script>
console.log(5 + 6);
</script>

</body>
</html>
```

Câu lệnh trong JS

Ví dụ về câu lệnh (statements) trong JS

```
var x, y, z;     // Statement 1

x = 5;           // Statement 2

y = 6;           // Statement 3

z = x + y;       // Statement 4
```

Chương trình trong JS

Một chương trình (program) trong JS

Một chương trình trong JS là 1 tập hợp các câu lệnh được thực thi bở máy tính

Trong HTML , JS được thực thi bởi trình duyệt

Câu lệnh trong JS

Câu lệnh trong JS là sự kết hợp của các từ khóa (keywords) giá trị (value) toán tử (operator) biểu thức (expression) và chú thích (comments)

Ví dụ về 1 câu lệnh viết ra text vào trong 1 thẻ html có id là demo :

```
document.getElementById("demo").innerHTML = "Hello Dolly.";
```

Dấu chấm phẩy ;

Dấu chấm phẩy được sử dụng để phân chia các câu lệnh trong JS

Kết thúc mỗi câu lệnh trong JS ta đều sử dụng dấu chấm phẩy ;

```
var a, b, c;      // Declare 3 variables

a = 5;            // Assign the value 5 to a

b = 6;            // Assign the value 6 to b

c = a + b;        // Assign the sum of a and b to c
```

Dấu ; trên 1 dòng

Cách viết nhiều câu lệnh trên 1 dòng sử dụng dấu ; để phân chia giữa các câu lệnh

```
a = 5; b = 6; c = a + b;
```

Khoảng trống trong JS

Trong JS lờ đi các khoảng trắng . Các khoảng trắng giúp cho các câu lệnh JS dễ đọc hơn .

```
var person = "Hege";
```

```
var person="Hege";
```

Độ dài của 1 dòng trong JS và cách ngắt dòng

Để cho dễ đọc hơn các dòng thường không nên dài quá 80 ký tự

Nếu 1 dòng trong JS quá dài hãy tìm cách ngắt dòng sau 1 toán như như + , - , * , / , =

```
document.getElementById("demo").innerHTML =

"Hello Dolly!";
```

Javascript code block

Các câu lệnh trong JS có thể được gom lại thành 1 khối (block) thông qua việc đặt các câu lệnh đó trong 1 dấu ngoặc xoắn {}

Mục đích của việc gom các câu lệnh thành block là để cho chúng được thực thi cùng nhau

Hàm trong JS là nơi bạn thường thấy sự xuất hiện của code block

```
function myFunction() {
    document.getElementById("demo1").innerHTML = "Hello
Dolly!";
    document.getElementById("demo2").innerHTML = "How are
you?";
}
```

JS Keywords Từ khóa trong Javascript

Js thường sử dụng các từ khóa để định nghĩa các hành động sẽ được thực thi

Đây là 1 vài từ khóa quan trọng :

break ;

continue ;

do … while ; ...

Cú pháp trong JS

Javascript là tập hợp các quy tắc viết code , các chương trình trong JS sẽ được thực hiện như thế nào .

```
var x, y;           // How to declare variables

x = 5; y = 6;       // How to assign values

z = x + y;          // How to compute values
```

JS variables (biến trong JS)

Biến là 1 đối tượng được sử dụng để lưu trữ thông tin

Nó như một chiếc USB để chứa dữ liệu

ta sử dụng từ khóa var để khai báo 1 biến và kết thúc bằng dấu ;

```
var x;
```

Ta sử dụng dấu = để gán 1 giá trị cho biến

```
x = 6;
```

Ví dụ về khai báo biến

```
// Biến x lưu trữ giá trị 5

var x = 5;

// Biến y lưu trữ giá trị 6

var y = 6;

// Biến z lưu trữ tổng của x và y

var z = x + y;
```

Cách khai báo tên biến trong JS

Tên biến trong JS cần là 1 tên duy nhất không trùng với tên biến khác

quy tắc đặt tên biến :

tên biến có thể chứa ký tự , số , _ , dấu $

tên biến phải bắt đầu bằng 1 chữ

tên biến có thể bắt đầu bằng _ hay $

tên biến có phân biệt hoa thường y khác Y

không sử dụng các từ khóa như if else do while var để khai báo tên biến

để gán giá trị cho 1 biến ta sử dụng dấu =

var x = 5;

2 kiểu dữ liệu số và chuỗi trong JS

```
// Kiểu dữ liệu số có thể là 1 số nguyên hay 1 số thập phân

var pi = 3.14;

// Kiểu chuỗi được đặt trong 1 dấu nháy kép  “” hay nháy đơn ‘’

var person = "John Doe";

// Khi bạn đặt 1 số trong 1 dấu nháy đơn ‘’ hay nháy “” JS sẽ hiểu đó là 1 chuỗi
```

Khai báo tên biến và gán giá trị

```
// Cách 1 khai báo biến và gán giá trị trong 2 dòng code

var carName;

carName = "Volvo";

// Cách 2 khai báo biến và gán giá trị trong 1 dòng code

var carName = "Volvo";
```

Document get element by ID

```
document.getElementById("demo").innerHTML = carName;
```

=> document trong lệnh trên là toàn bộ văn bản html

getElementById là 1 phương thức cho phép lấy 1 đối tượng có id được truyền vào trong dấu ngoặc ()

innerHTML là nội dung HTML bên trong thẻ có id tương ứng

Ví dụ về get element by id

```
<p id="demo"></p>

<script>

var carName = "Volvo";

document.getElementById("demo").innerHTML = carName;

</script>
```

Value = undefined

Trong JS có 1 kiểu dữ liệu là undefined khi bạn khai báo biến mà không gán giá trị nào cho biến cả thì khi đó kiểu dữ liệu của biến là undefined

Nối chuỗi trong JS

Để nối chuỗi trong JS bạn sử dụng dấu +

trong ví dụ :

var x = 4 + 5;

đây là phép cộng nhưng khi viết là

var x = “hello” + “js”;

lúc này dấu + sẽ là phép nối chuỗi

Nối chuỗi trong JS

var x = “kết quả là : ” + 3;

lúc này dấu + cũng sẽ là 1 phép nối chuỗi

Toán tử trong JS

dấu = là toán tử gán

var x = 5;

dấu + là toán tử phép cộng

var x = 5 + 3;

Toán tử + - * /

dấu - là toán từ phép trừ

var x = 5 - 3;

dấu * là toán tử phép nhân

var x = 5 * 3;

Toán tử chia / và %

dấu / là toán từ phép chia

var x = 15/3;

phép chia % lấy dư

var x = 12%5;

khi này x sẽ là 2 vì 12 chia cho 5 dư 2

Toán tử ++ và --

toán tử ++ là toán tử tăng thêm 1 đơn vị

var x = 5;

x++;

x lúc này là 6 vì 5 tăng 1 đơn vị là 6

Toán tử --

toán tử giảm 1 đơn vị

var x = 5;

x--;

khi này x là 4 vì 5 giảm 1 đơn vị còn 4

Dạng toán tử viết tắt của phép gán

var x = x + y;

cách viết thứ 2 là x += y;

var x = x - y;

cách viết thứ 2 là x -= y;

Dạng toán tử viết tắt của phép gán

var x = x * y;

cách viết thứ 2 là x *= y;

var x = x / y;

cách viết thứ 2 là x /= y;

Dạng toán tử viết tắt của phép gán

var x = x % y;

cách viết thứ 2 là x %= y;

Dấu + phép nối chuỗi

khi đi với các số dấu + là phép cộng

khi đi với chuỗi có số hay các chuỗi dấu + là phép nối chuỗi

```
var txt1 = "John";

var txt2 = "Doe";

var txt3 = txt1 + " " + txt2;
```

Thực hành phép nối chuỗi

```
// kết quả là 10

var x = 5 + 5;

// kết quả là 55

var y = "5" + 5;

kết quả là Hello5

var z = "Hello" + 5;
```

Các kiểu dữ liệu trong JS

1 kiểu số

```
var length = 16;
```

Kiểu chuỗi

```
var lastName = "Johnson";
```

Kiểu mảng

```
var cars = ["Saab", "Volvo", "BMW"];
```

ta sử dụng [] để khai báo mảng

mảng là 1 danh sách các phần tử

các phần tử phân tách bằng dấu ,

mảng có giá trị và key (số thứ tự)

số thứ tự trong mảng bắt đầu từ 0

Kiểu đối tượng

```
var person = {firstName:"John", lastName:"Doe", age:50,
eyeColor:"blue"};
```

Kiểu đối tượng khai báo bằng {}

đối tượng có các thuộc tính và giá trị

từ khóa typeof

từ khóa typeof cho ta biết kiểu của 1 biến

```
var x;

x = 5;

console.log(typeof x);
```

Hàm trong JS

Hàm trong js là 1 block (khối) các câu lệnh đặt trong dấu {}

được viết ra để thực hiện 1 chức năng mà bạn sẽ không phải viết lại nhiều lần khi cần sử dụng

ví dụ như hàm tính chu vi hình tròn , hàm tính chu vi hình vuông , hàm thu nhập cá nhân …

khi cần sử dụng hàm ta chỉ cần gọi hàm và truyền các tham số (dữ liệu cần thiết cho hàm)

Vi dụ

```
function myFunction(p1, p2) {
    return p1 * p2;                // The function returns
the product of p1 and p2

}
```

Cấu trúc

```
function name(parameter1, parameter2, parameter3) {
    code to be executed
}
```

Cách gọi hàm

```
var x = myFunction(4, 3);     // Function is called, return
value will end up in x



function myFunction(a, b) {

    return a * b;               // Function returns the
product of a and b

}
```

Từ khóa return

từ khóa return sẽ kết thúc 1 hàm và trả về dữ liệu theo sau từ khóa return

Các câu lệnh viết sau return sẽ không được thực thi

Câu lệnh điều kiện if else else if

Cú pháp :

```
if (condition) {
    block of code to be executed if the condition is true
}
```

Ví dụ

```
if (hour < 18) {

      greeting = "Good day";

}
```

```
if (hour < 18) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

```
if (time < 10) {
    greeting = "Good morning";
} else if (time < 20) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

Lệnh switch

```
switch (new Date().getDay()) {
    case 0:
      day = "Sunday";
         break;
    case 1:
      day = "Monday";
         break;
    case 2:
      day = "Tuesday";
         break;
    case 3:
      day = "Wednesday";
         break;
    case 4:
      day = "Thursday";
         break;
    case 5:
      day = "Friday";
         break;
    case 6:
      day = "Saturday";
}
```

Vòng lặp for

Ví dụ for

```
var i;

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

    text += cars[i] + "<br>";

}
```

Vòng lặp

Trong lập trình vòng lặp for được sử dụng để thực hiện các công việc lặp lại theo chu kỳ

ta cần chú ý đến 3 điều :

điểm bắt đầu của vòng lặp

điều kiện của vòng lặp

sự thay đổi sau mỗi vòng lặp

```
for (i = 0; i < 5; i++) {
    text += "The number is " + i + "<br>";
}
```