Bài 1. JavaScript là gì? Tìm hiểu ngôn ngữ lập trình javascript

1. Javascript là gì

* JavaScript là một ngôn ngữ lập trình kịch bản dựa vào đối tượng phát triển có sẵn hoặc tự định nghĩa
* Được sử dụng rộng rãi trong các ứng dụng Website
* Javascript được hỗ trợ hầu như trên tất cả các trình duyệt như Firefox, Chrome,.. trên máy tính lẫn điện thoại
* Nhiệm vụ của Javascript là xử lý những đối tượng HTML trên trình duyệt
* Nó có thể can thiệp với các hành động như thêm/ xóa / sửa các thuộc tính css và các thẻ HTML một cách dễ dàng
* Là một ngôn ngữ lập trình trên trình duyệt ở phía client
* Sự xuất hiện của NodeJS đã giúp cho JavaScript có thể làm việc ở backend
* Sự xuất hiện của các framwork như NodeJs(chuyên code backend), ExpressJS(NodeJS framework), và nhiều thư viện frontend khác như Angular, jQuery, ReactJS ra đời, giúp tạo ra một cơn sốt với từ kháo Javascript

1. Javscript Framework là gì

* Là một bộ thư viện được xây dựng dựa vào ngôn ngữ lập trình JavaScript
* Mỗi framework thường được tạo ra để phục vụ cho một lĩnh vực nào đó
* Ví dụ Angular và React thì chuyên xử lý frontend, NodeJS thì chuyên xử lí backend
* jQuery là một thư viện rất mạnh khi xử lý DOM HTML và CSS
* Agular:
* NodeJS:Một thư viện được phát triển phía Server dùng để xây dựng ứng dụng realtime
* Sencha Touch: Một Framework dùng để xây dựng ứng dụng Mobie
* ExtJS: Một Framework dùng để xây dựng ứng dụng quản lý(Web Applications)
* jQuery: Một thư viện rất mạnh về hiệu ứng
* ReactJS: Một thư viện viết ứng dụng mobie

1. Ưu điểm và nhược điểm của Javascript

Ưu điểm

* Giúp thao tác với người dùng ở phía client và tách biệt giữa các client
* Có thể hoạt động trên nhiều nền tảng khác nhau
* Là một ngôn ngữ dễ tiếp cận, học mà không cần cài đặt quá nhiều phần mềm

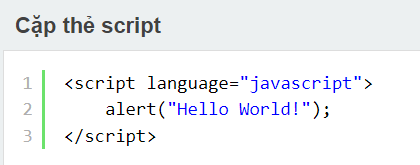
Nhược điểm:

* Dễ dàng bị soi code

1. Cách viết chương trình javascript đầu tiên

Cặp thẻ mở và thẻ đóng:

Tất cả những đoạn mã Javascript đều phải đặt trong cặp thẻ mở <script> và thẻ đóng </script>

Ví dụ: 

Đặt thẻ script ở đâu

* Có 3 cách đặt thường được sử dụng:
* Cách 1: Internal – viết trong file html hiện tại: Viết những đoạn mã trên phần head
* Cách 2: External – viết ra một file js khác rồi import vào
* Cách 3: Inline – viết trực tiếp trong thẻ HTML

1. Các câu hỏi thường gặp khi học Javascript

* Thẻ script là gì
* Thẻ script dùng để khai báo cho trình duyệt biết nội dung bên trong sẽ là một đoạn mã Javascript
* Type = text/javascript là gì?
* Khi khai báo thẻ script thì sẽ có thuộc tính type = “text/javascript”. Đây là tham số dùng để khai báo media type cho dữ liệu nằm bên trong của thẻ script. Mặc định nếu không khai báo thì nó sẽ hiểu là text/javascript
* Javascript có là được backend không?
* Java là ngôn ngữ client nên không thể thao tác trên phía server
* Tuy nhiên khi server có cài đặt NodeJS thì ở server hàn toàn có thể dùng ngôn ngữ Javascript, bởi NodeJs sử dụng Javascript.
* Có nên học JavaScript không
* JavaScript rất nên học và học các FW JS khác nữa

Bài 2: Các cách khai báo biến trong Javascript

Tìm hiểu 3 cách khai báo biến trong Javascript: var – let – const

Học thêm cách dùng hàm document.write() để in giá trị biến ra ngoài trình duyệt

Việc khai báo biến trong Javascript rất đơn giản, chúng ta không cần phải khai báo kiểu dữ liệu cho biến mà nó sẽ tự nhận kiểu dữ liệu khi chúng ta gán giá trị lần đầu cho nó

1. Các cách khai báo biến trong javascript

* Khai báo biến trong JS bằng từ khóa var

var username;

* Khai báo biến trong JS bằng từ khóa let:

let username;

Chú ý: biến loại này chỉ có phạm vi trong khối khai báo mà thôi. Vì vậy nó sẽ tối ưu trong trường hợp chúng ta cần khai báo một biến chỉ sử dụng tạm thời trong một phạm vi nhất định. Phạm vi này ta gọi là block scope



* Khai báo biến trong JS bằng từ khóa const

Const là một hằng số, vì vậy khi khai báo biến const thì chúng ta phải gán giá trị cho nó luôn

Khi gán giá trị rồi thì không thể thay đổi giá trị cho biến

1. Cách đặt tên cho biến trong Javascript

* Tên biến phải là các chữ không dấu viết hoa hoặc viết thường, các chữ số từ 0-9 và dấu gạch dưới (\_).
* Tên biến bắt đầu phải là chữ hoặc dấu gạch dưới (\_), nếu bắt đầu bằng số là sai
* Tên biến có thể đặt dài hay ngắn tùy vào mỗi người lập trình

1. Cách gán giá trị cho biến trong Javascript

* Sử dụng dấu ( = ) để gán giá trị cho biến

1. Kiểu giá trị của biến trong Javascript

* Kiểu chuỗi
* Kiểu số
* Kiểu mảng
* Kiểu đối tượng

1. In giá trị của bến JS ra trình duyệt

* Sử dụng hàm document.write(value) để in giá trị của biến hoặc một chuỗi nào đó ra trình duyệt.

1. Các phép toán thường dùng trên biến trong Javascript

* Gán giá trị
* Cộng trừ nhân chia các số
* Nối chuỗi

Bài tập câu hỏi thường gặp liên quan:

Hãy khai báo 5 biến đúng và 5 biến sai trong Javascript.

Nhớ: Khai báo biến phải bắt đầu bằng chữ hoặc \_

Khai báo biến không được bắt đầu bằng số hay kí tự đặc biệt, khi khai báo nhiều biến phải có dấu phẩy ngăn cách tên giữa các biến

Trong Javascript có thể gán giá trị cho biến lúc khai báo không?

🡪 Có thể gán giá trị cho biến lúc khai báo thông qua phép gán

Javascript – khai báo biến không có từ var được không?

🡪 Trong Javascript khi khai báo nên đặt từ khóa var ở đằng trước vì đây là cú pháp chuẩn của Javascript

Javascript chia làm hai chế độ như sau:

* Strick Mode: Đây là chế độ nghiêm ngặt, tất cả cú pháp đều phải tuân thủ theo đúng nguyên tắc
* None Strick Mode: Đây là chế độ không nghiêm ngặt(mặc định sẽ là chế độ này, nghĩa là nó có thể bỏ qua một số lỗi cơ bản)

Bài 3: Hàm alert() – confirm() – prompt() trong javascript

Ba hàm rất thông dụng trong javascript, đó là hàm alert(), confirm() và prompt(). Đây là ba hàm thường được sử dụng tạo hộp thoại thông báo và lấy thông tin từ người dùng

Tính năng hiển thị popup để lấ thông tin người dùng hoặc hiển thị popup thông báo rất là phổ biến

Tuy nhiên thực tế thì các website lại rất ít khi sử dụng các hàm này bởi giao diện hiển thị của nó rất là củ chuôi. Thay vào đó thì sẽ sử dụng thư viện jQuery popup khác chuyên nghiệp hơn

1. Hàm alert() trong javascript

Hàm alert() có nhiệm vụ in một thông báo popup, nó có một tham số truyền vào và tham số này chính là nội dung ta muốn thông báo với người dùng

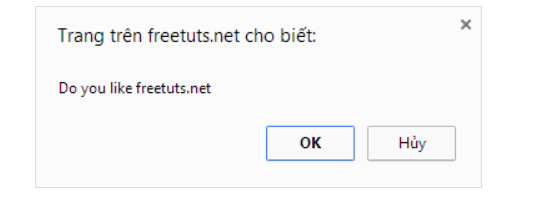
Ví dụ: Viết chương trình khi click bào button thì xuất hiện một thông báo

Có hai cách

1. Viết qua hàm: sử dụng cặp thẻ <script></script>
2. Viết trực tiếp inline: onclick không cần sử dụng cặp thẻ <script></script>

2. Hàm confirm() trong javascript

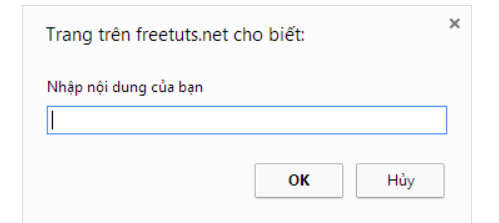
Hàm confirm() cũng sẽ xuất hiện một thông báo popup nhưng nó có thêm hai sự lựa chọn là YES và NO, nếu người dùng chọn YES thì nó trả về TRUE và ngược lại nếu chọn NO thì nó sẽ trả về FALSE. Nó cũng sẽ có mọt tham số truyền vào và tham số này chính là nội dung thông báo

o

Ví dụ: Viết chương trình hiển thị thông báo xác nhận và dùng hàm alert() để hiện thị kết quả người dùng chọn

3. Hàm prompt() trong javascript

Hàm prompt() dùng để lấy thông tin người dùng, gồm có hai tham số truyền vào là nội dung thông báo và giá trị ban đầu. Nếu người dùng không nhập vào thì giá trị nó sẽ trả về là NULL



Câu hỏi thường gặp liên quan:

* Sử dụng hàm alert để in ra thông tin cá nhân của bạn
* Trường hợp nào thì sử dụng hàm prompt, confirm và alert
* Sử dụng hàm confirm xác nhận thao tác của người dùng
* Sử dụng hàm prompt lấy thông tin của khách hàng

**Bài 4: Các toán tử trong Javascript**

1. **Toán tử toán học trong javascript**

Phép cộng (+):

var a = 20:

var b = 5;

// Kết quả biến c có giá trị 25:

var c = a + b;

Phép trừ (-) và dùng với number

Phép nhân (\*) và dùng với number

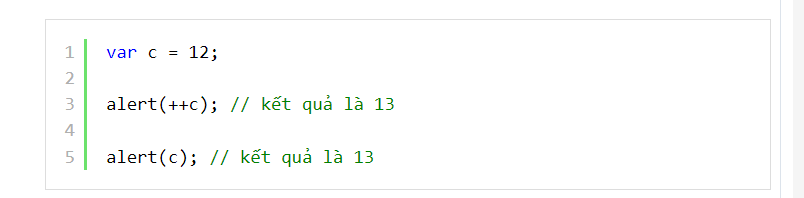
Phép chia (/) và dùng với number

Phép chia lấy phần dư (%)

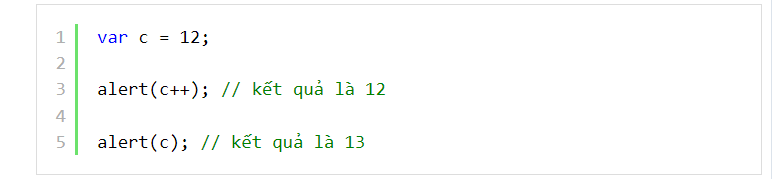
Phép tăng giá trị hiện tại lên 1 đơn vị. Phép này có hai cách sử dụng đó là đặt nó trước biến và đặt nó sau biến

Lưu ý:

Trường hợp đứng trước biến thì nó sẽ tăng trước khi lấy giá trị

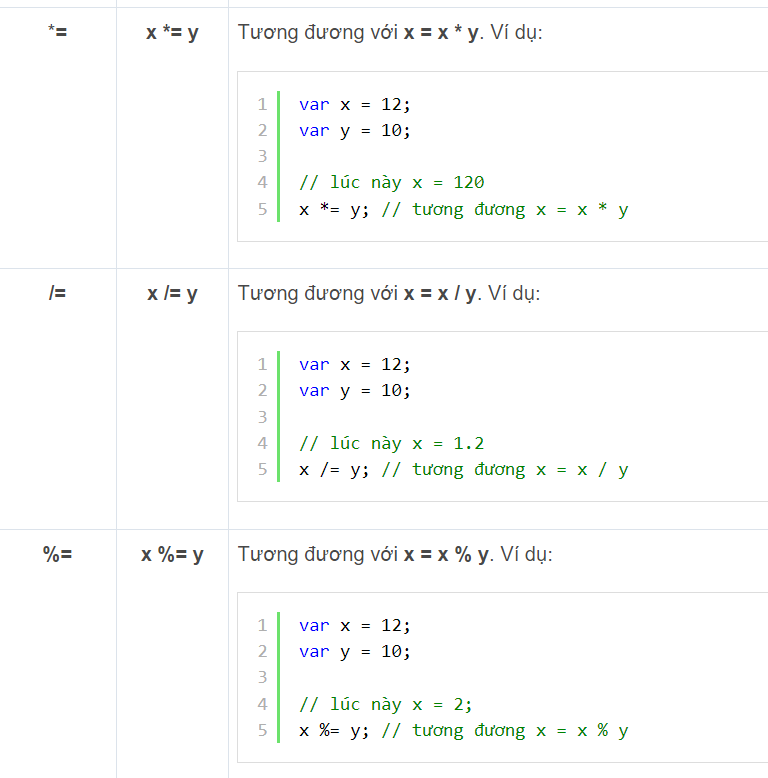


Trường hợp đứng sau biến thì nó sẽ lấy giá trị rồi tăng lên, ví dụ:

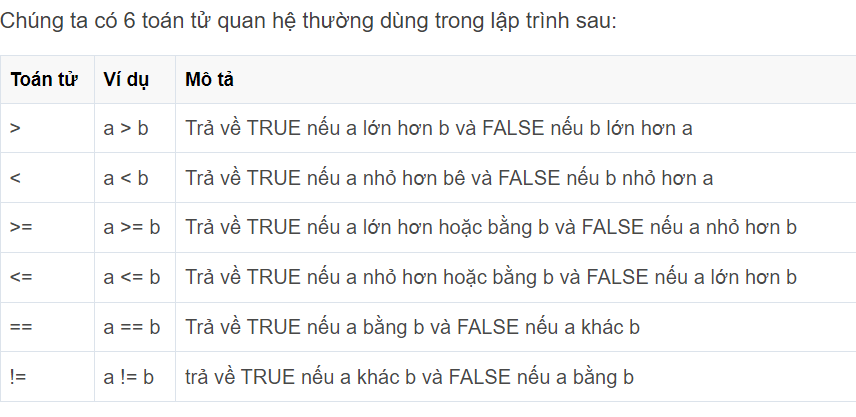


1. **Toán tử gán trong javascript**

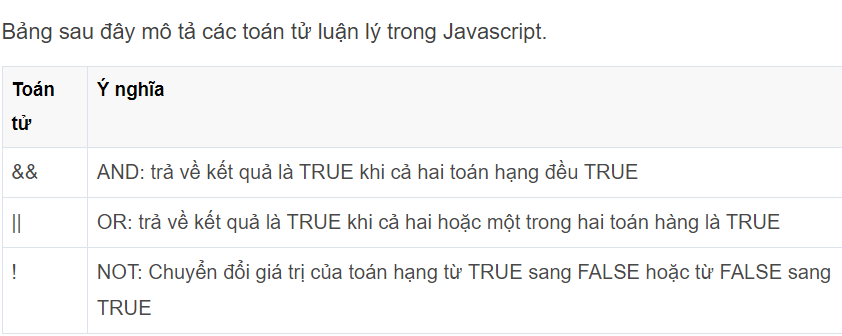




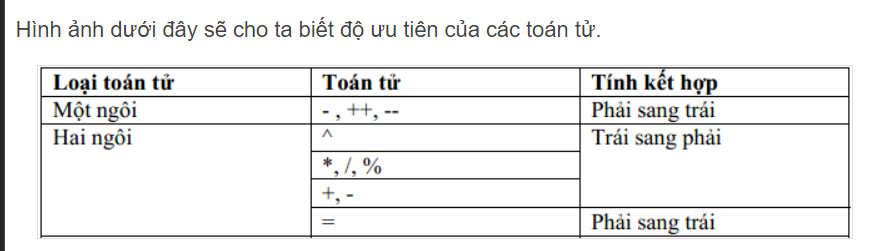
1. **Toán tử quan hệ trong javascript**

****

1. **Toán tử luận lý trong javascript**

****

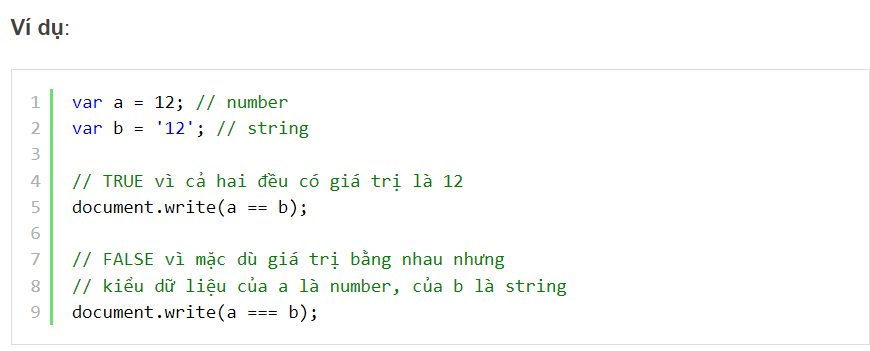
1. **Độ ưu tiên các toán tử trong javascript**

****

1. **Lưu ý với toán tử so sánh bằng trong javascript**

Sự khác nhau giữa hai toán tử == và === là :

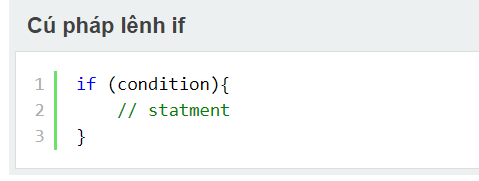
Về nguyên tắc thì cả hai toán tử đều có chung một chức năng là so sánh giá trị của hai biến, tuy nhiên với toán tử == thì chỉ so sánh giá trị mà không so sánh kiểu dữ liệu, còn toán tử === thì có so sánh luôn kiểu dữ liệu



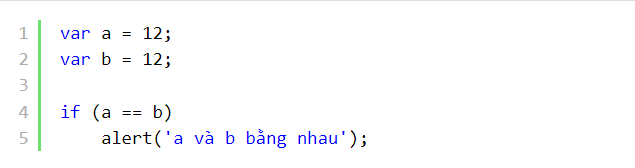
1. **Lời kết**
2. Bài tập
3. Tính giá trị của hai số a và b được nhập vào từ hàm prompt
4. Javascript: sử dụng tất cả các toán tử với hai số a và b
5. Javascript a++ và ++a khác nhau như thế nào

Bài 5: Lệnh if else trong javascript( dùng để rẻ nhánh chương trình)

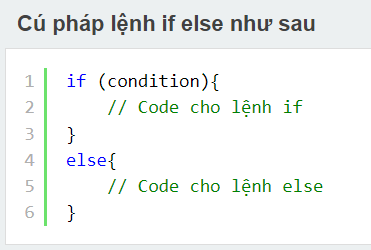
1. Lệnh if trong javascript



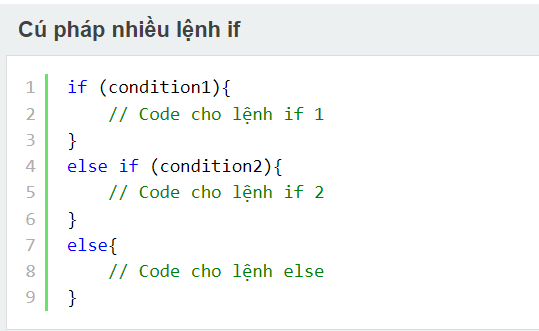
Trường hợp nếu các câu lệnh bên trong là câu lệnh đơn (chỉ có một lệnh) thì ta có thể bỏ cặp dấu ngoặc nhọn. Ví dụ



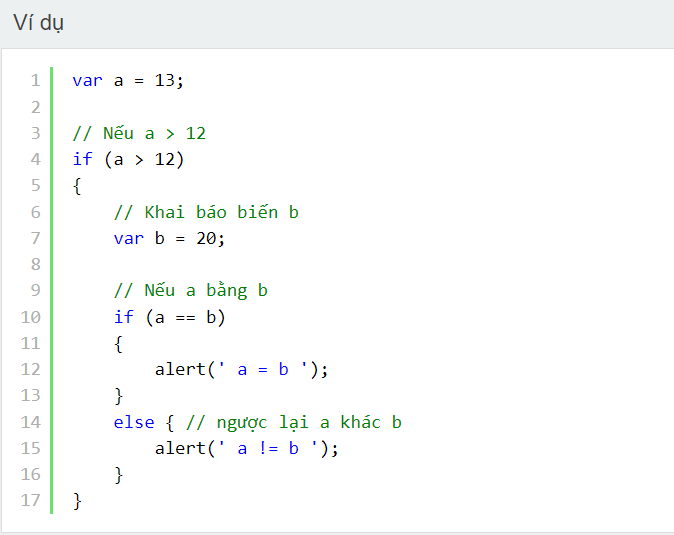
2. Lệnh if else trong javascript



3. Kết hợp nhiều lệnh if else trong javascript



4. Lệnh if else lồng nhau trong javascript



5. Hàm if trong HTML có không

Không có hàm if trong HTML

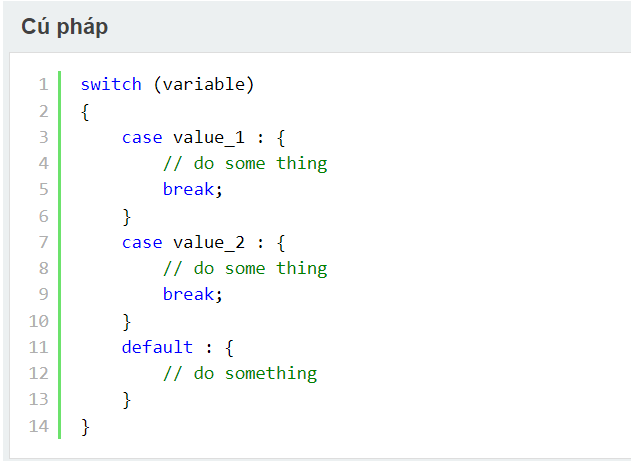
Thay vào đó nếu chúng ta muốn xử lý cách hiển thị HTML thì có thể sử dụng lệnh if else, hoặc sử dụng ngôn ngữ phía server như PHP, ASP, JSP

Bài Tâp:

* Tìm min và max của hai số cho trước bằng Javascript
* Javascript: Thuật toán kiểm tra số chẵn hay lẻ
* Kiểm tra số chẵn hay lẻ bằng Javascript
* Javascript: Kiểm tra một số có chia hết cho n hay không

Bài 6. Lệnh switch case trong Javascript

1. Lệnh swtich case trong Javascript



1. Ví dụ lệnh switch case trong Javascript

Trường hợp không có default

Trường hợp không có break: Nếu không có break thì nó chỉ chạy luôn case đầu tiên phía dưới nó

Trường hợp gom nhóm case

1. Lời kết
2. Bài tập:

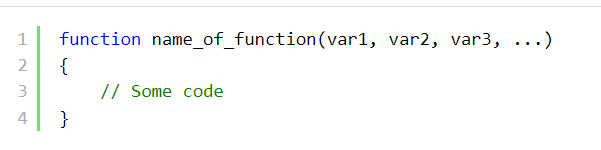
* Tìm thứ trong tuần của ngày hiện tại
* Tìm tên các loại trái cây trong kho hàng
* Chuyển đổi giữa lệnh if else và switch

Bài 7. Function trong Javascript – cách tạo hàm và gọi hàm trong JS

1. Function trong Javascript là gì

Function hay còn gọi là hàm, là tập hợp một đoạn code dùng để xử lý một nhiệm vụ nào đó. Code bên trong function không được biên dịch cho tới khi được gọi đến

1. Một ví dụ về lợi ích của function trong Javascript
2. Cách tạo function trong Javascript



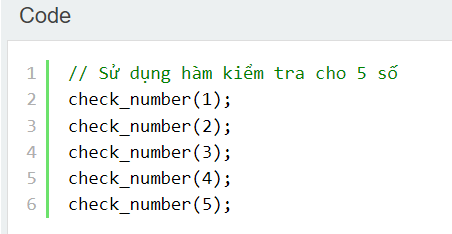
Trong đó:

* function: là từ khóa của javascript nên bắt buộc phải như vậy
* name\_of\_function: là tên của hàm, thông thường tạo những tên có ý nghĩa như find\_max, find\_min,….
* Var1, var2, var3,…. là các tham số truyền vào hàm. Ví dụ viết hàm kiểm tra số chẵn hay lẻ thì ta sẽ có một tham số đó là số cần kiểm tra

Ta cũng có thể tạo một hàm không có tham số

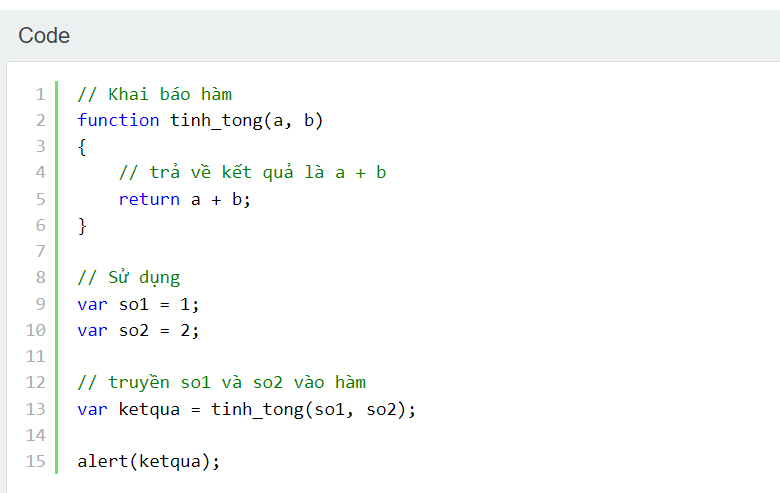
1. Cách gọi hàm trong Javascript

Để gọi hàm thì ta chỉ cần gọi đến tên hàm, sau đó truyền vào các tham số cần thiết

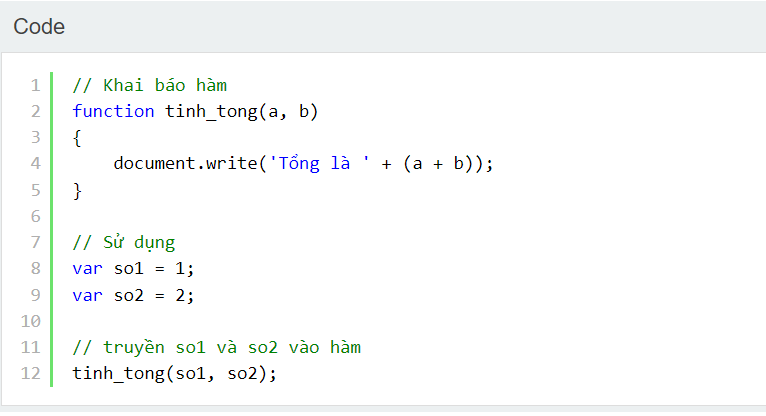


1. Hàm có return và hàm không có return

Hàm có return là hàm có sử dụng từ khóa return để đặt ở cuối hàm với mục đích trả về kết quả để sử dụng tiếp ở những đoạn code bên ngoài



Hàm không có return là hàm không có sử dụng từ khóa return đặt trong hàm.

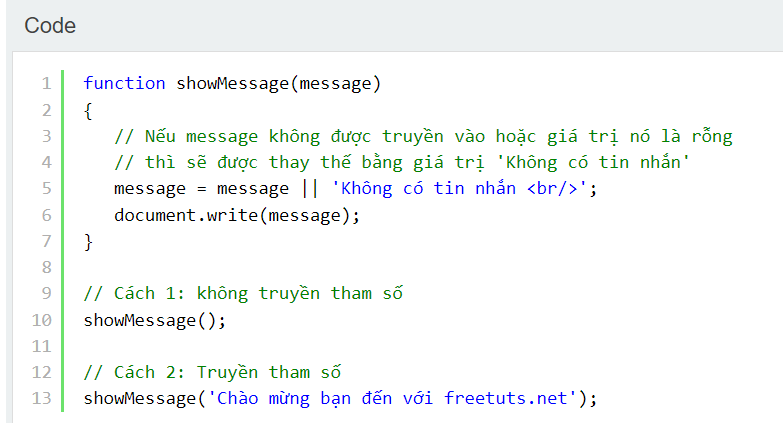


Tùy vào mục đích mà ta có return hay không có return

1. Giá trị mặc định của tham số (default parameter)

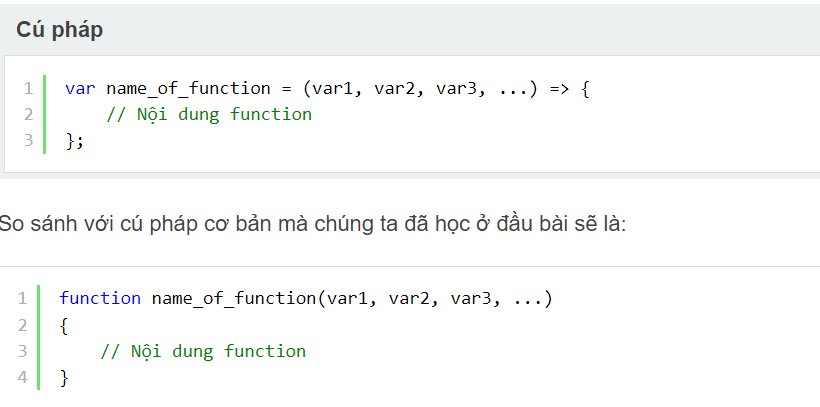
Javascript không có cú pháp gán giá trị mặc định như PHP mà thay vào đó chúng sử dụng toán tử ||

Ví dụ



1. Cách tạo hàm arrow function trong Javascript

Arrow function còn được gọi làm hàm mũi tên



1. Một số ví dụ tạo hàm trong javascript:

Tạo hàm kiểm tra một năm có phải là năm nhuận hay không

1. Các hàm có sẵn trong javascript là gì\*\*

Bài 8: Biến toàn cục và biến cục bộ trong Javascript

1. Biến cục bộ:

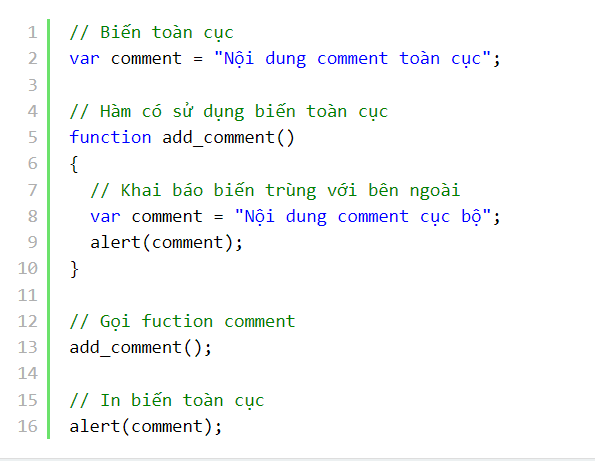
Một biến được gọi là cục bộ khi chúng ta khai báo biến nằm bên trong một hàm cụ thể nào đó, lúc này biến đó sẽ không sử dụng được ở bên ngoài hàm

1. Biến toàn cục

Biến toàn cục là biến mà bạn khai báo bên ngoài và không nằm bên trong một hàm cụ thể nào cả

1. Một số ví dụ biến toàn cục và biến cục bộ

Nếu trong hàm **có sử dụng** từ khóa var để tạo một biến và tên của bến đó **đã tồn tại ở bên ngoài**(toàn cục) thì lúc này bên trong hàm sẽ hiểu là đang sử dụng biến cục bộ nên không ảnh hưởng gì biến bên ngoài cả



Nếu không sử dụng từ khóa var để khai báo biến trong hàm thì JS sẽ sử dụng biến toàn cục nên mọi thay đổi giá trị của bến đó sẽ được lưu lại



1. Lời kết

Biến toàn cục là biến được khai báo bên ngoài hàm và được sử dụng ở tất cả các vị trí

Biến cục bộ là biến khai báo ở trong hàm và chỉ sử dụng được trong nội bộ của hàm đó thôi, sau khi hàm thực thi xong thì ngay lập tức nó sẽ bị xóa khỏi bộ nhớ

**Bài 7: Kiểm tra lỗi Javascript bằng Firebug(Inspect Element)**

Muốn bật firebug thì chúng ta cần click chuột phải, sau đó chọn Inspect Element.

Cách khác có thể ấn tổ hợp phím Ctrl + Shift + I

1. Firebug là gì

Firebug là một addon dành cho các trình duyệt web Firefox và Chrome, giúp lập trình viên có thể xem tất cả các thông tin về website như:

* Xem được cấu trúc HTML và CSS.
* Xem được lỗi Javascript nếu có.
* Sửa các phần tử HTML và CSS
* Xem các request URL gửi lên server.
* Xem các tham số header của các request

2. Inspect Element là gì?

Inspect Element là một tính năng có công dụng giống như Firebug, chỉ có điều nó tích hợp sẵn vào các trình duyệt chứ không cần phải cài đặt trên internet

Để mở cửa sổ Inspect Element thì ta có ba cách:

Nhấn F12

Hoặc ấn tổ hợp phím: Ctrl + Shift + I

Hoặc click chuột phải vào một vị trí trên trang web và chọn Inspect Element



Trong đó các tab:

Element là nơi hiển thị danh sách các thẻ HTML và CSS cho thẻ đó.

Console là nơi hiển thị các lỗi javascript, show kết quả khi chạy lệnh console.log

Source là hiển thị domain các tài nguyên mà trang web hiện tại đang sử dụng

Network là nơi hiển thị danh sách URL mà trang web đang sử dụng

Các tab còn lại khác dạnh cho nhưng ai có kiến thức chuyên sâu hoặc đang viết JS Framework

Bài 9: Hàm console.log trong Javascript(dùng để xem giá trị của biến)

Đây là hàm dùng để in giá trị của một biến ra ngoài ô Console của Inspect Element

Ta có thể áp dụng cho tất cả các loại dữ liệu như number, integer, array, object

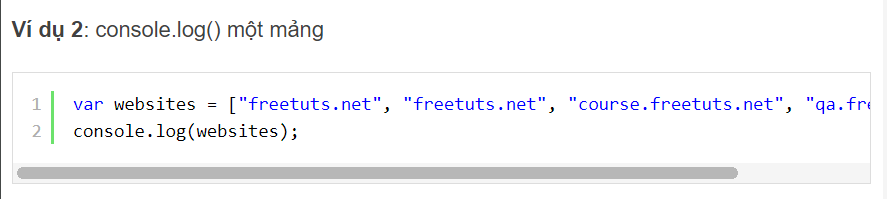
1. Console.log trong Javascript là gì

Console.log là một hàm của Javascript, được sử dụng để debug/ kiểm tra / xem giá trị của một biến trong Javascript. Công dụng của console.log là show ra giá trị của một biến, dù biến đó thuộc kiểu dữ liệu gì đi nữa vẫn được hiển thị ở ô console trong hộp thoại Inspect Element.

2. Cú pháp console.log() trong javascript

Lệnh console.log() có cú pháp là console.log(value) trong đó value là một biến hoặc một giá trị mà chúng ta muốn in ra.







**Bài 10: Hàm setTimeout() và setInterval() trong Javascript**

Hai hàm setTimeout() và setInterval() trong Javascript: đây là hai hàm dùng để hẹn giờ trong JS

***1. Hàm setTimeout() trong Javascript***

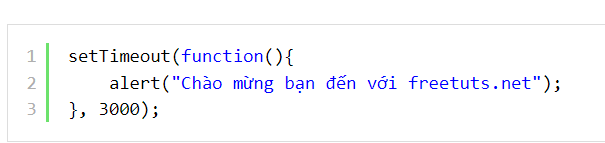
Hàm setTimeout() dùng để thiết lập một khoảng thời gian nào đó sẽ thực hiện một nhiệm vụ nào đó và nó chỉ thực hiện đúng một lần

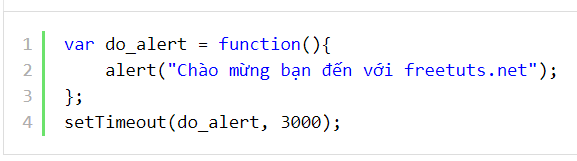
**Cú pháp**: setTimeout(function, time)

Trong đó:

* Function: là nội dung cần thực hiện, đây là một hàm
* Time: là khoảng thời gian bao nhiêu( tính bằng mili giây) thì function sẽ thực hiện

Ví dụ: Sau 3 giây thì xuất hiện câu chào lên màn hình

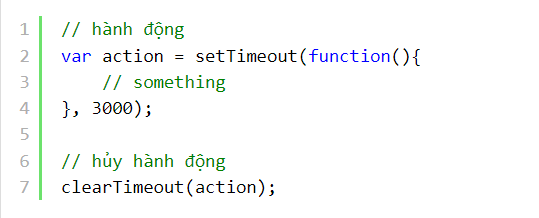


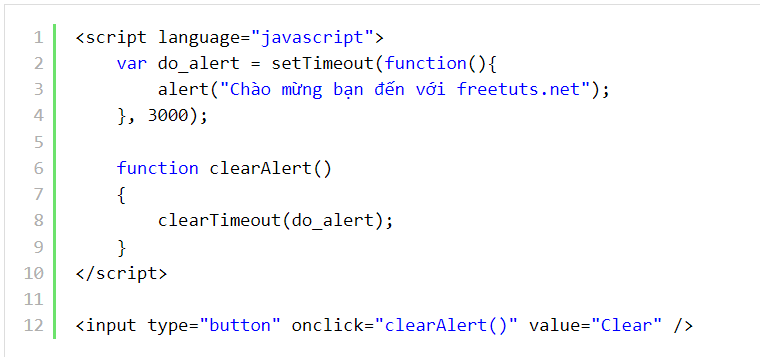


2. Hàm clearTimeout() trong Javascript

Giả sử chúng ta xây dựng chức năng sau 3 giây thì sẽ xuất hiện thông báo, tuy nhiên sau 2 giây chương trình muốn hủy bỏ thì phải làm thế nào? Lúc này chúng ta phải sử dụng hàm clearTimeout()

Tham số truyền vào hàm clearTimeout() là đối tượng setTimeout() nên lúc này chúng ta phải đặt hàm setTimeout() vào một biến cụ thể

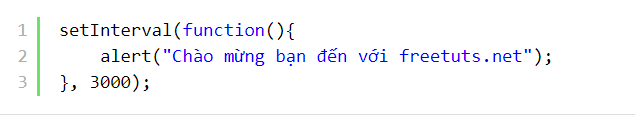




3. Hàm setInterval() trong Javascript

Hàm setInterval() có cú pháp và chức năng giống như hàm setTimeout(), tuy nhiên với hàm setInterval() thì số lần thực hiện là mãi mãi

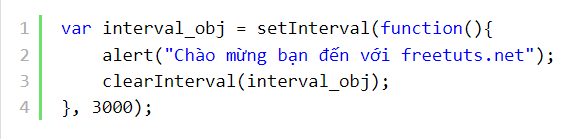
Ví: Cứ sau 3 giây thì xuất hiện câu chào một lần



4. Hàm clearInterval() trong Javascript

Tương tự hàm clearTimeout(), hàm clearInterval() sẽ xóa đi nhiệm vụ mà ta đã thiết lập trong hàm setInterval(), và ta cũng phải đặt setInterval() trong một biến thì mới clear được

Ví dụ: Sử dụng hàm setInterval() để xuất câu chào lên màn hình và số lần xuất hiện chỉ 1 lần day nhất



Trong ví dụ trên nội dung thực hiện là xuất hiện câu chào và ngay lập tức xóa nhiệm vụ của interval luôn nên nó chỉ thực hiện một lần

5. Lời kết:

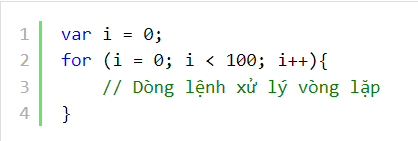
Hàm setTimeout() và setInterval() là đối với hàm setTimeout() thì số lần thực hiện là một lần và đối với setInterval() thì số lần thực hiện sẽ là mãi mãi cho tới khi chúng ta sử dụng hàm clearInterval() để can thiệp vào

**Bài 11: Cách dùng vòng lặp for trong Javascript**

1. Vòng lặp for trong Javascript là gì?

Vòng lặp for trong javascript là vòng lặp được dùng để lặp một mảng hoặc một danh sách nằm trong khoảng (min 🡪 max). Đây là vòng lặp có thứ tự và phải biết trước tổng số lần lặp nên thường dùng trong những bài toán liệt kê và xử lý các phần tử trong danh sách mảng

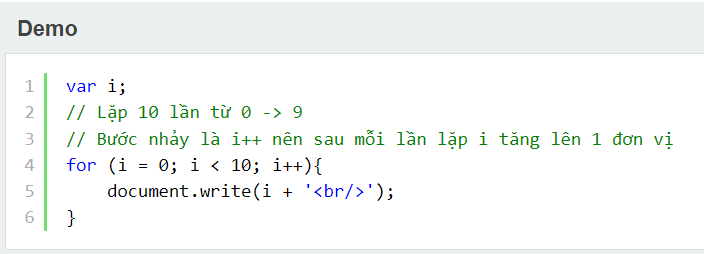
Cấu trúc:

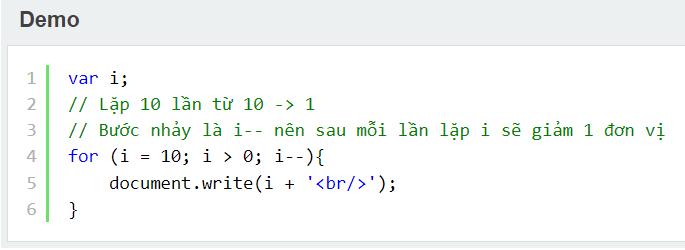


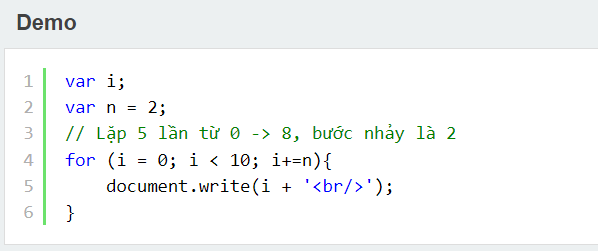
Trong đó:

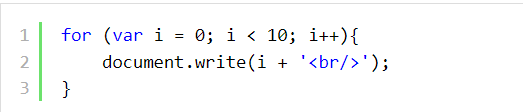
* var i = 0; là khai báo biến điều khiển vòng lặp i
* i = 0 là điểm bắt đầu lặp (lặp từ 0)
* i < 100 là điều kiện dừng vòng lặp, nghĩa là lặp nếu i < 100. Chúng ta có thể dùng một điều kiện bất kì thông qua các toán tử miễn là nó trả về true hoặc false
* i++ là tăng bước nhảy, chúng ta có thể dùng các công thức như i+= 2, i-= 2

2. Cách sử dụng vòng lặp for trong Javascript

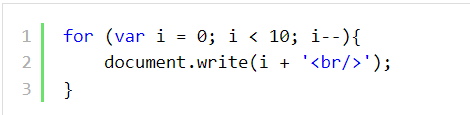




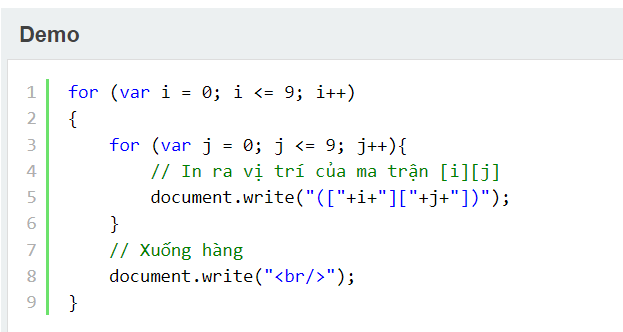




3. Vòng lặp for vô hạn trong Javascript



4. Vòng lặp for lồng nhau trong javascript



Bài tập:

1. Vòng lặp for trong Javascript in ra các số từ 1 – 100

2. Vòng lặp for trong Javascript in ra các số từ 100 – 0

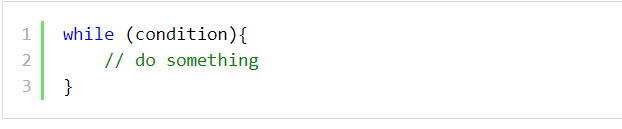
3. Vòng lặp for trong Javascript in ra các số chia hết cho 3 từ 0 – 100

**Bài 12: Vòng lặp while – do while trong javascript**

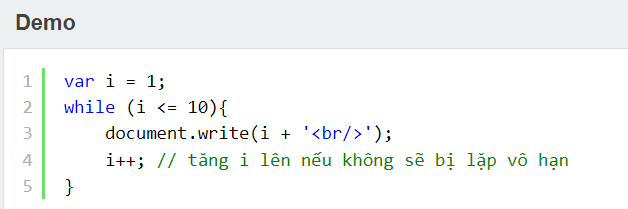
Vòng lặp while và do while dùng để lặp với trường hợp ta không biết chính xác số lần lặp là bao nhiêu và trường hợp điều kiện dừng vòng lặp quá phức tạp, điều này hoàn toàn khác với vòng lặp for. Khi sử dụng vòng lặp while rất dễ bị lặp vô hạn nếu không xử lý logic

**1. Vòng lặp while trong javascript**

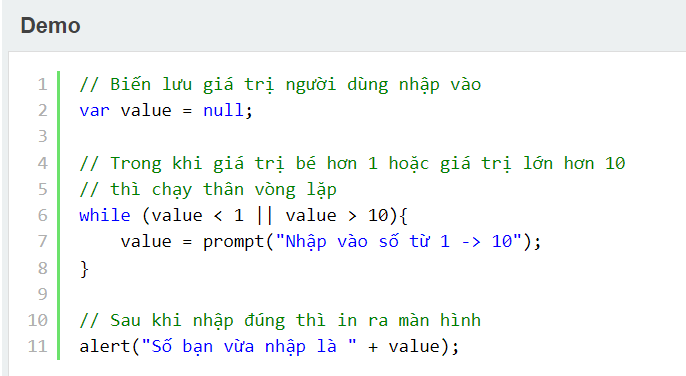
Cấu trúc của vòng lặp while:



Trong đó condition là điều kiện dừng vòng lặp, nếu condition đúng thì vòng lặp sẽ được thực thi cho tới khi condition có giá trị sai. Chính vì vậy nếu condition luôn luôn đúng thì vòng lặp sẽ dẫn tới vô hạn



Lặp với điều kiện phức tạp

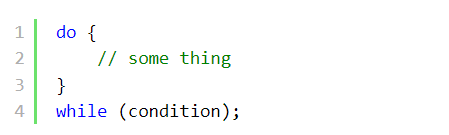


Giải thích:

* Tạo ra biến value ban đầu có giá trị null
* Sử dụng vòng lặp while
* Chừng nào mà value còn nhỏ hơn 1 và lớn hơn 10 thì bắt người dùng nhập lại
* Nếu value > 1 và < 10 thì khi đó hiển thị value ra màn hình

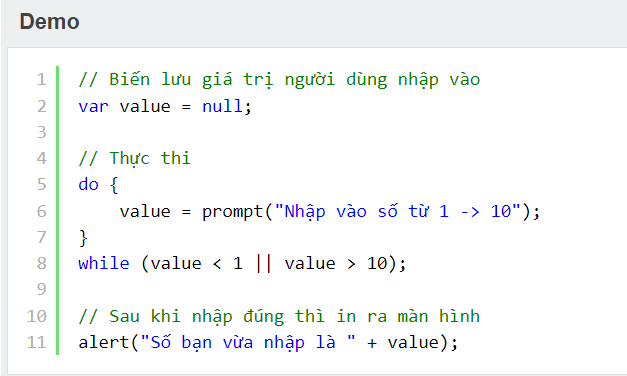
2. Vòng lặp do while trong javascript

Cấu trúc vòng lặp do while



Trong đó condition là điều kiện để dừng vòng lặp

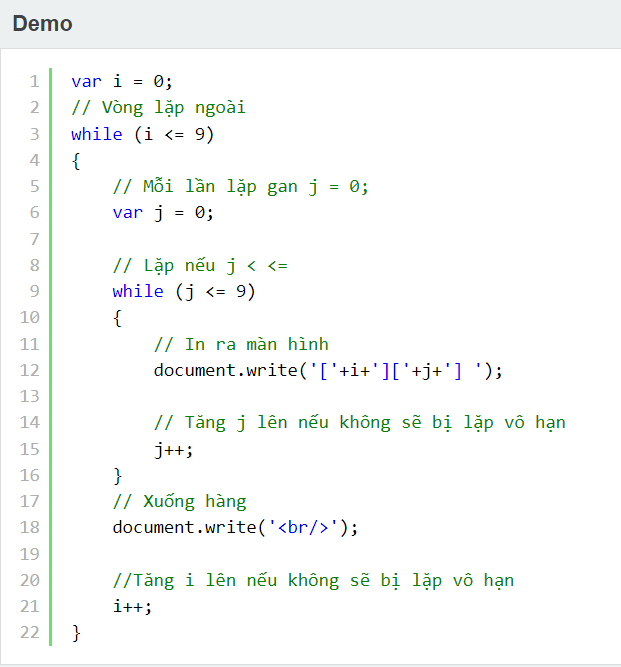
Vòng lặp do while luôn luôn lặp ít nhất là 1 lần



Giải thích:

* Ban đầu tạo ra một biến value có giá trị null
* Sau đó dùng vòng lặp do – while
* Chừng nào value còn < 1 và > 10 thì ta sẽ yêu cầu nhập lại giá trị của value
* Bởi vì value ban đầu khởi tạo là null nên luôn có một lần lặp
* Khi nào điều kiện value là > 1 và < 10 thì sẽ hiển thị ra

3. Vòng lặp while – do while lồng nhau



Bài tập:

*1. Javascript – In các số từ 1 – 100 bằng vòng lặp while và do while*

*2. Javascript – Chuyển bài toán từ while sang do while và ngược lại*

*3. Javascript – giải bài toán bằng vòng lặp for và while*

*4. Tóm tắt ba vòng lặp for – while và do while trong Javascript*

**Bài 13: Lệnh break – continue trong Javascript**

***1. Lệnh break trong Javascript***

Lệnh break có tác dụng dừng vòng lặp cho dù điều kiện của vòng lặp vẫn đang đúng, hay nói cách khác nó thoát khỏi vòng lặp một cách đột xuất và không quan tâm đến điều kiện lặp

Lệnh break có thể sử dụng với mọi vòng lặp như vòng lặp for, while và do while, each,…

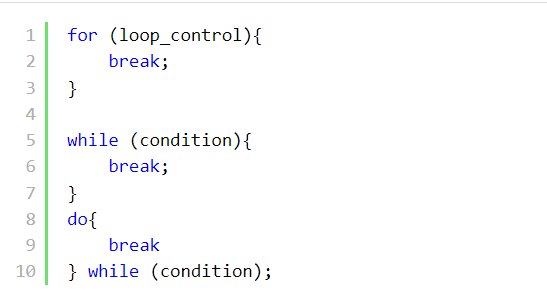
Cú pháp break trong js:

[loop control]

break;

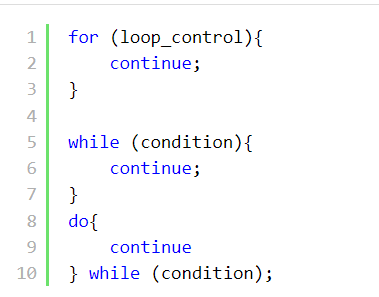
[end loop]

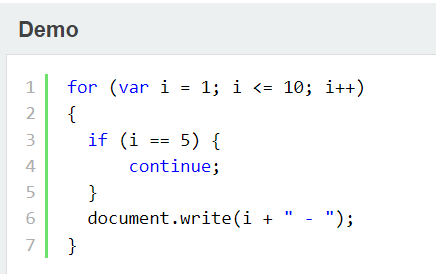
Ráp break vào các vòng lặp:

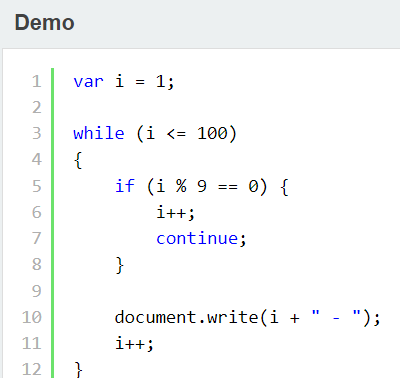


***2. Lệnh continue trong Javascript***

Khác hoàn toàn với lệnh break, lệnh continue trong javascript có tác dụng bỏ qua một bước lặp nào đó, nghĩa là lúc gặp lệnh continue thì tất cả những đoạn code nằm bên dưới sẽ không thực hiện mà nó nhảy qua vòng lặp luôn







***3. Lời kết***

**Bài 15: Xử lý chuỗi( string ) trong Javascript**

1. Chuỗi string trong Javascript là gì?

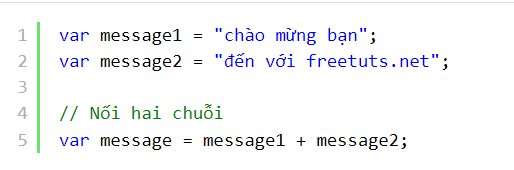
Chuỗi là một đoãn văn bản(text) có thể có một hoặc nhiều ký tự và thông thường chúng ta sẽ lưu trữ nó vào một biến, biến này ta sẽ gọi là biến có kiểu dữ liệu là String(chuỗi). Tất cả các chuỗi đều phải được bao quanh bằng cặp dấu nháy đơn ‘ hoặc nháy kép ”

2. Cách nối chuỗi string trong Javascript

Để nối chuỗi string trong JS thì chúng ta sử dấu + để ghép hai chuỗi hoặc biến kiểu String Object lại với nhau

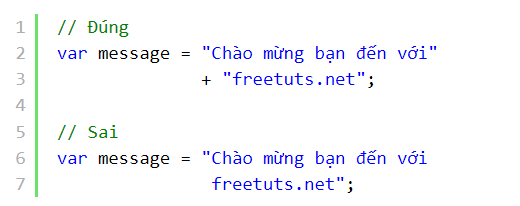
Ví dụ: Ghép hai string lại rồi gán vào biến message





3. Xử lý string trên nhiều dòng trong Javascript

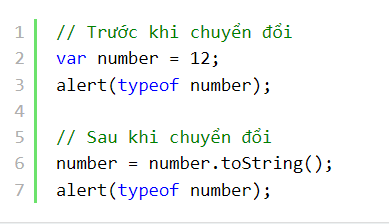
Khi chúng ta muốn Enter xuống hàng một chuỗi trong Javascript thì bắt buộc phải sử dụng dấu + để nối chuỗi, nếu không sẽ bị lỗi cú pháp





4. Ép chuỗi string trong javascript

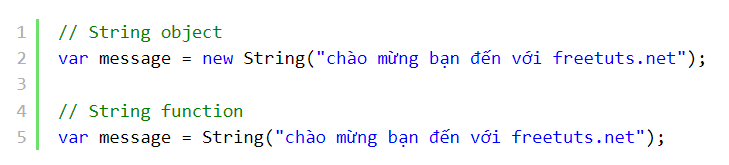
Nếu muốn ép một giá trị nào đó sang kiểu chuỗi thì có thể sử dụng cú pháp string.toString().



Từ khóa typeof vars sẽ trả về kiểu dữ liệu của biến vars

Ngoài các trên ta có thể sử dụng đối tượng String để tạo hoặc ép chuỗi, và vì là đối tượng nên nó phải có từ khóa new đặt đằng trước

Ví dụ: Sử dụng String object hoặc String function.

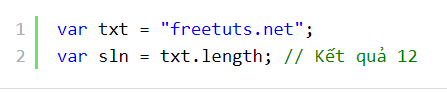


Không nên sử dụng cách này bởi vì chương trình sẽ chạy chậm hơn

5. Một số hàm xử lý string trong Javascript

Đếm số ký tự trong chuỗi

Đếm số ký tự trong chuỗi string thì ta sử dụng thuộc tính length.

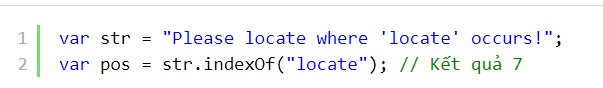


Tìm kiếm trong chuỗi

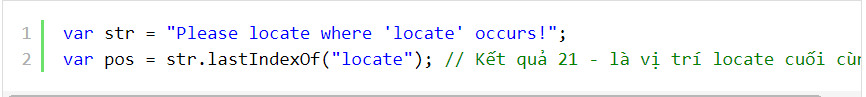
Nếu muốn tìm kiếm một chuỗi bên trong một chuỗi khác thì dùng hàm indexOf

Nếu tìm thấy thì nó sẽ trả về vị trí xuất hiện đầu tiên của chuỗi con

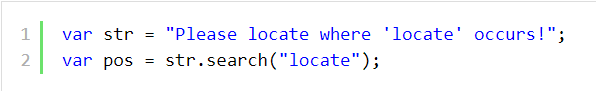
Nếu không tìm thấy thì trả về -1



Nếu muốn tìm vị trí xuất hiện cuối cùng của chuỗi con thì dùng hàm lastIndexOf



Ngoài ra chúng ta có thể sử dụng hàm search, nó có công dụng giống như hàm indexOf



Bài 16: Các hàm xử lý chuỗi trong Javascript(cắt / tách / nối chuỗi…)

1. Các hàm xử lý chuỗi trong Javascript

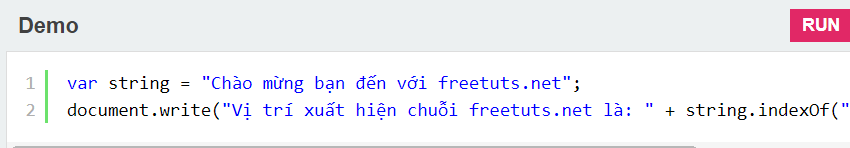
Tìm kiếm chuỗi string trong JS

Chúng ta có ba hàm thường dùng để tìm kiếm chuỗi con trong Javascript như sau:

* indexOf()
* lastIndexOf()
* search()

**Hàm IndexOf()**

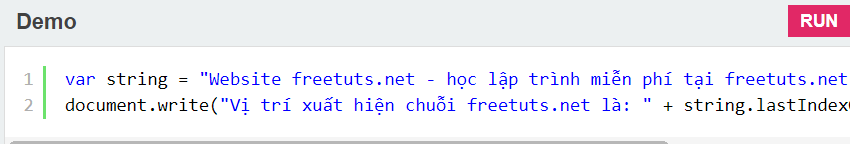
Để tìm kiếm chuỗi con ta sử dụng hàm String.indexOf(str), trong đó str là chuỗi con và String là chuỗi cha. Hàm này sẽ trả về kết quả là vị trí xuất hiện đầu tiên của chuỗi (bắt đầu là vị trí 0), nếu không tìm thấy chuỗi con thì nó sẽ trả về -1



**Hàm lastIndexOf()**

Trường hợp nếu chuỗi con xuất hiện nhiều lần trong chuỗi cha thì kết kết quả cũng trả về vị trí xuất hiện của chuỗi con đầu tiên

Để lấy vị trí của chuỗi con cuối cùng trong chuỗi cha. Ta sẽ sử dụng hàm String.lastIndexOf(str), hàm này sẽ trả về vị trí xuất hiện của chuỗi con cuối cùng và trả về -1 nếu không tìm thấy



**Hàm search()**

String.search(str) trong String là chuỗi cha còn str là chuỗi con

**Cắt chuỗi string trong JS**

Nếu muốn cắt một chuỗi con từ chuỗi cha thì chúng ta có thể sử dụng ba hàm sau

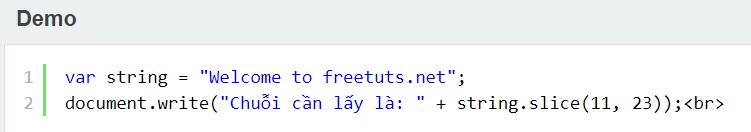
* slice(start, end)
* substring(start, end)
* substr(start, length)

Note: tất cả các vị trí của chuỗi đều bắt đầu từ 0, vì vậy khi tính toán vị trí cẩn thận bị nhầm lẫn

Hàm slice()

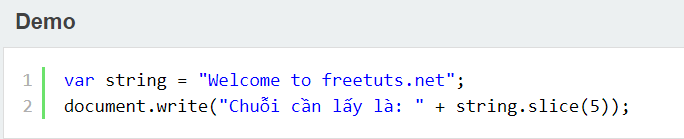
Hàm slice có hai tham số truyền vào:

* strar: vị trí bắt đầu
* end: vị trí kết thúc



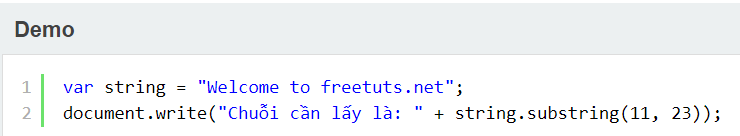
Nếu tham số truyền vào là số âm thì nó sẽ tính ngược lại, nghĩa là nó sẽ đếm từ cuối lên

Nếu chỉ truyền một tham số đầu tiên thì nó sẽ tự hiểu vị trí end là vị trí cuối cùng



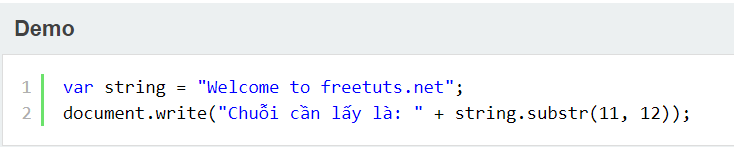
Hàm substring()

Hàm substring có cách sử dụng giống với hàm slice(), tuy nhiên tham số truyền vào substring() phải luôn lớn hơn 0.



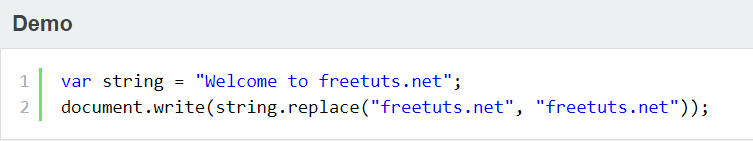
Hàm substr()

Hàm substr() có hai tham số là start và length, trong đó start là vị trí bắt đầu và length là số ký tự muốn lấy bắt đầu từ vị trí start. Nếu ta truyền tham số start là số âm thì nó sẽ tính từ cuối trở lên, còn tham số length phải luôn là số dương.



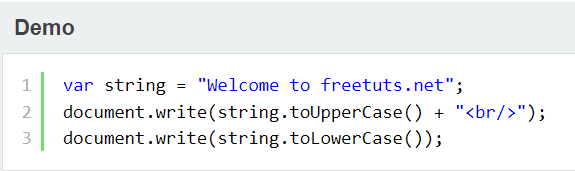
Tìm kiếm và lặp chuỗi trong JS

Để tìm kiếm là và lặp một chuỗi con nào đó thì ta sử dụng hàm replace(str\_find, str\_replace) trong đó str\_find là chuỗi cần tìm và str\_replace là chuỗi sẽ được thay thế chuỗi str\_find



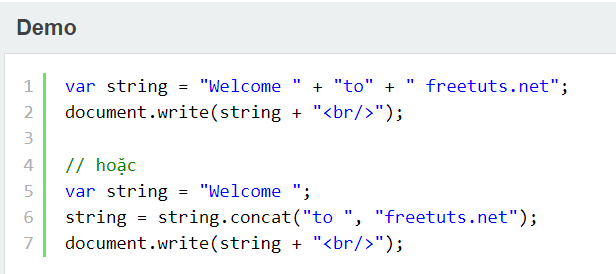
Chuyển thành chữ hoa và chữ thường trong JS

Để chuyển chuỗi thành chữ hoa ta dùng hàm toUpperCase() và chuyển thành chữ thường ta dùng hàm toLowerCase().



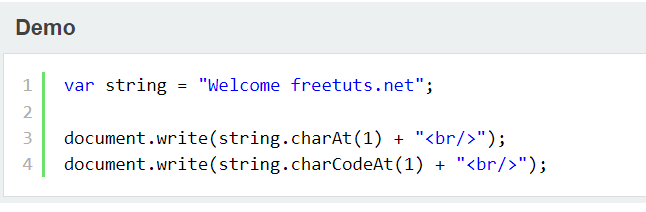
Nối thêm chuỗi

Để nối thêm chuỗi thông thường ta dùng toán tử +, ngoài ra ta có thể dùng hàm concat() để thực hiện nối chuỗi



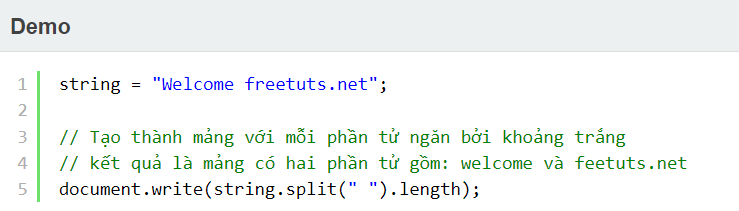
Tìm ký tự hoặc mã ASCII của một ký tự

Để xem ký tự của một vị trí nào đó thì dùng hàm chartAt(), còn xem mã ASCII thì chùng hàm charCodeAt(). Cả hai hàm này đều có tham số truền vào là vị trí muốn xem



Chuyển đổi chuỗi sang mảng JS

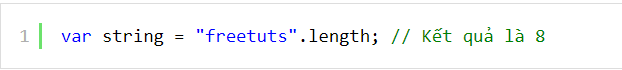
Để chuyển một chuỗi sang mảng thì ta sử dụng hàm split() với tham số truyền vào là ký tự ngăn cách giữa các phần tử



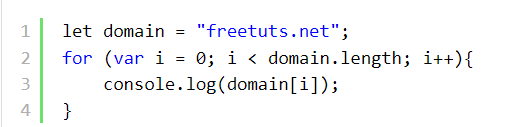
Vẫn còn nhiều hàm khác nữa có thể tham khảo trong bài viết Javascript functions

2. Cách lặp qua từng ký tự chuỗi string trong JS

Trong đối tượng JS có một thuộc tính lưu trữ tổng số ký tự trong chuỗi, đó là length



Trong Js chuỗi được xem là một cấu trúc mảng, vì vậy có thể sử dụng vòng lặp để lặp qua từng phần tử của chuỗi rất dễ dàng



3. Lời kết

Thực tế vẫn còn một thư viện xử lý String trong JS rất hay nữa, đó là sử dụng Regular Expression. Tuy nhiên RegEx thuộc phần kiến thức nâng cao nên học ở phần sau

Bài 17: Khai báo và duyệt mảng trong javascript

1. Mảng trong Javascript là gì

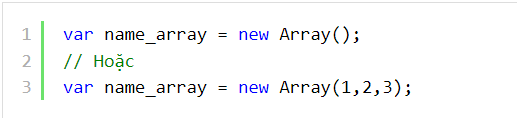
Mảng là một loại dữ liệu dạng tập hợp nhiều phần tử, trong đó mỗi phần tử sẽ được đánh dấu vị trí bằng chỉ mục

Thường sử dụng mảng để lưu trữ dữ liệu dạng danh sách

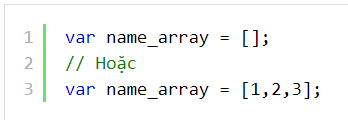
2. Khi bảo mảng trong javascript

Có hai cách khai báo mảng:

1. Sử dụng từ khóa new Array()

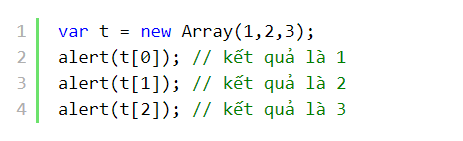


1. Sử dụng cặp dấu ngoặc vuông []



3. Truy xuất các phần tử trong mảng

Cú pháp truy xuất phần tử như sau: tenmang[vị trí]



4. In mảng ra trình duyệt và console

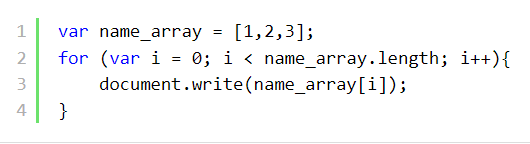
Sử dụng hai hàm join() và console.log() để in toàn bộ giá trị của các phần tử ra màn hình

Hàm join(): tên\_mảng.join()

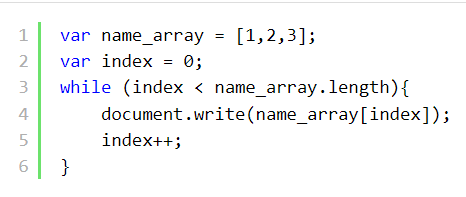
Hàm console.log(): console.log(tên\_mảng)

5. Sử dụng vòng lặp để duyệt mảng trong javascript

Duyệt mảng với vòng lặp for

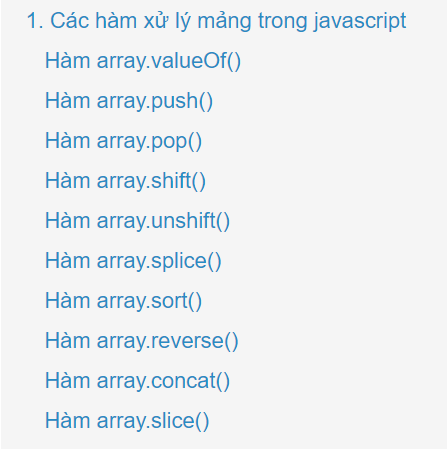


Duyệt mảng với vòng lặp while

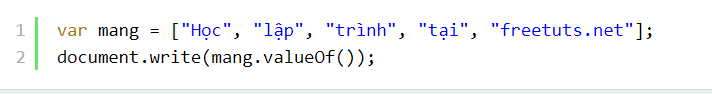


Chú ý: Không khuyến khích duyệt mảng với vòng lặp do while vì vòng lặp do while luôn luôn lặp ít nhất một lần nên trong trường hợp mảng cần lặp rỗng thì sẽ báo lỗi ngay

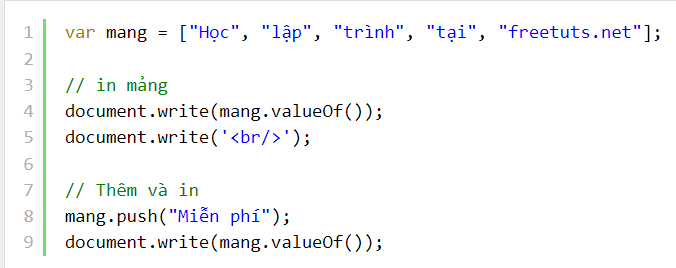
Bài 18: Các hàm xử lý mảng trong javascript hữu ích



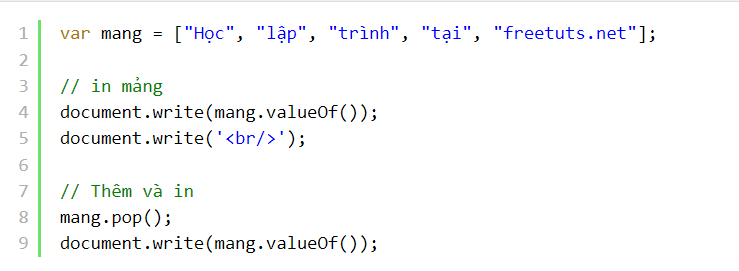
Hàm array.valueOf(): Hàm này có tác dụng tương tự như hàm array.join() có nghĩa là nó sẽ nối các phần tử với nhau vào một chuỗi cách nhau bởi dấu phầy



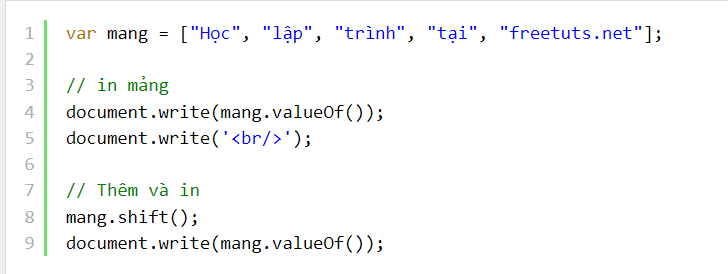
Hàm array.push(): Hàm này sẽ thêm một phần tử vào cuối mảng



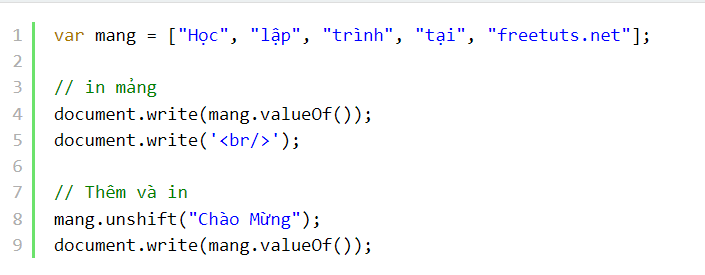
Hàm array.pop(): Ngược với hàm array.push, hàm này có tác dụng xóa đi phần tử cuối cùng trong mảng



Hàm array.shift(): Hàm xóa phần tử đầu tiên của mảng, sau đó dồn các phần tử phía sau xuống một bậc



Hàm array.unshift(): Thêm một phần tử vào vị trí đầu tiên của mảng, đồng thời đẩy các phần tử phía sau lên một bậc:



Hàm array.splice()

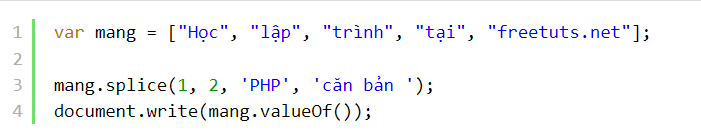
Hàm splice() có ba tham số truyền vào như sau: splice(position\_add, num\_element\_remove, value1, value2, ….)

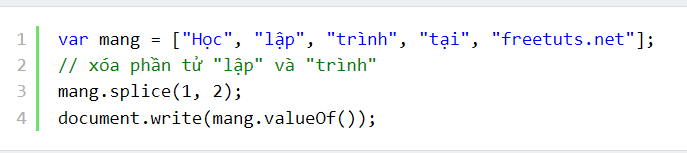
Trong đó:

Position\_add là vị trí sẽ thêm (vị trí đầu tiên là 0)

Num\_element\_remove là số phần tử sẽ xóa (bắt đầu từ position\_add)

Value1, value2, …. Là danh sách các phần tử sẽ được thêm vào sau khi tại vị trí position\_add và sau khi xóa số lượng num\_element\_remove phần tử





Bài 19: Cách để tạo mảng hai chiều trong Javascript

1. Mảng hai chiều trong javascript là gì?

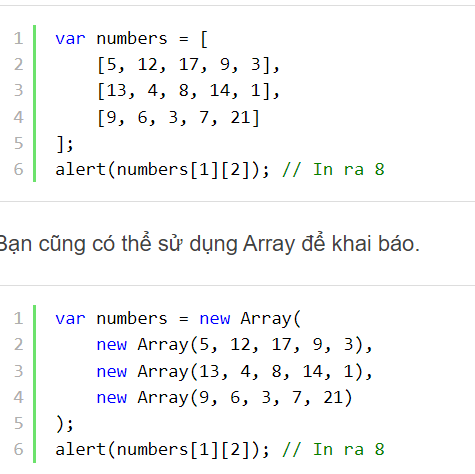
Mảng hai chiều là một loại mảng đặc biệt, nó có cấu trúc như mảng một chiều

Có một điểm khác duy nhất đó là mỗi phần tử của mảng lại là một mảng khác sẽ có hai chỉ mục, đó là vị trí nằm trong hàng và cột

[row][col]

2. Khai báo mảng hai chiều trong javascript

Để khai báo mảng hai chiều trong js thì ta sẽ khai báo một mảng một chiều, sau đó mỗi phần tử của mảng sẽ gán giá trị là một mảng khác



3. Lấy phần tử mảng hai chiều trong javascript

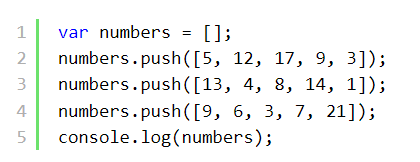
array[row][column]

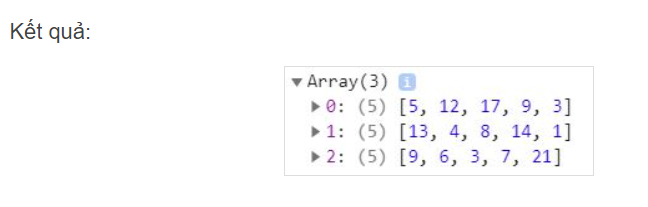
4. Thêm phần tử vào mảng hai chiều trong javascript

Để thêm một phần tử vào mảng thì ta có hai trường hợp, thứ nhất là thêm vào cuối mảng và thứ hai là thêm vào đầu mảng

Thêm vào cuối mảng hai chiều

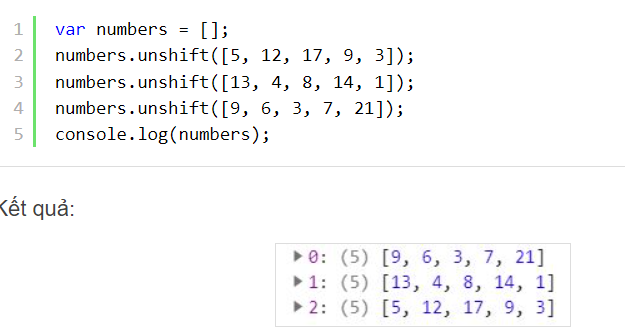
Ta sử dụng hàm push() để thêm một phần tử vào cuối mảng hai chiều





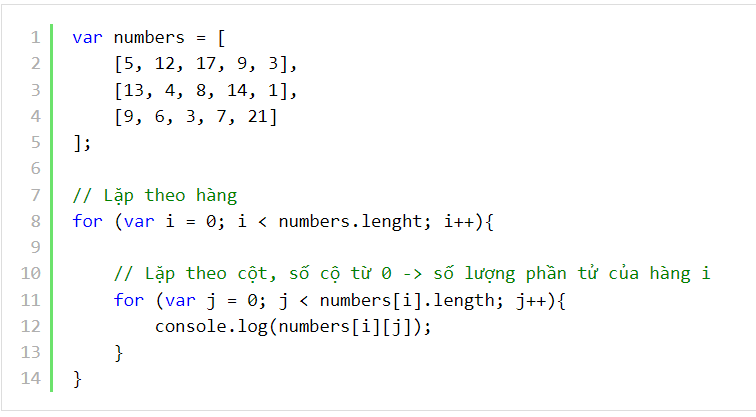
Thêm vào đầu mảng hai chiều

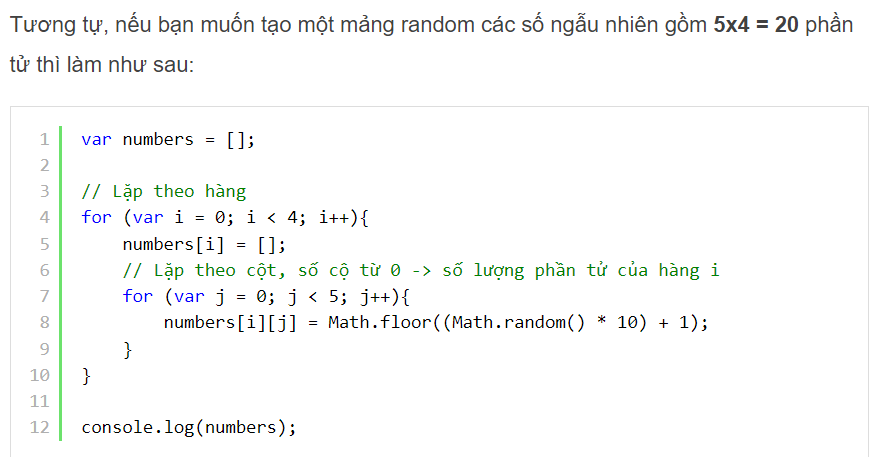
Ta sử dụng hàm unshift() để thêm phần tử vào đầu mảng



Khi sử dụng hàm unshift() thì phần tử thêm đầu tiên sẽ bị dồn về cuối mảng, còn phần tử thêm cuối cùng sẽ nằm ở đầu mảng. Lý do là hàm unshift() sẽ đẩy các phần tử lên để nhường chỗ cho các phần tử cần thêm

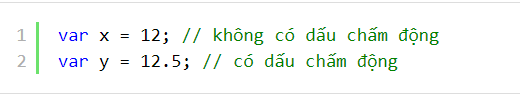
5. Duyệt mảng hai chiều trong javascript

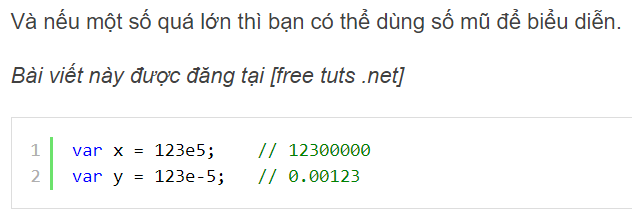




Bài 20: Đối tượng Number trong Javascript(cách tạo và cách sử dụng)

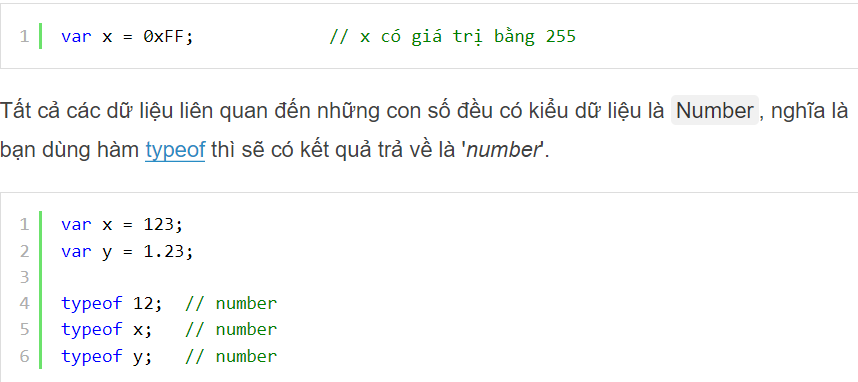
Trong Javascript có một đối tượng thường được dùng để xử lý dữ liệu kiểu number, đó là đối tượng Number. Đối tượng này có hai dạng, đó là số có dấu chấm động và số không có dấu chấm động





Không giống với các ngôn ngữ lập trình khác, các số trong Javascript luôn luôn là 64 bít và kiểu float, vì vậy chúng ta không thể định nghĩa các kiểu dữ liệu như Integer, short, long… Hay nói cách khác, trong Javascript khi làm việc với các chữ số thì chỉ có khái niệm Number

Ngoài ra chúng ta có thể biểu diễn giá trị Number ở dạng nhị phân, thập phân, thập lục phân,…



II. Thao tác với Number trong Javascript

Chuyển Number sang String

Để chuyển một biến đang ở kiểu Number sang kiểu String thì ta sử dụng phương thức number.toString(type), hàm này có một tham số truyền vào là type và đây chính là kiểu dữ liệu mà chúng ta muốn chuyển về, mặc định sẽ là hệ thập phân (10)

Đây là danh sách các hệ cơ số thông dụng:

Hệ nhị phân (2)

Hệ bát phân (8)

Hệ thập phân (10)

Hệ thập lục phân (8)

