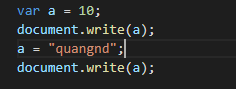
# Biến – Toán tử

## Biến



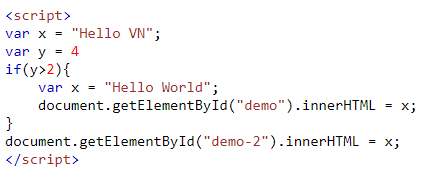
Kết quả:



Biến trong phụ thuộc vào giá trị mà biến đó nhận tại thời điểm xử lý lệnh. Một biến có thể nhận nhiều kiểu dữ liệu khác nhau.

### Var

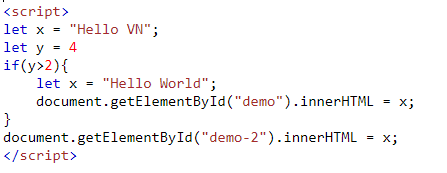
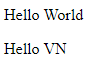
* Khi điều kiện xảy ra var sẽ thay đổi hoàn toàn biến được khai báo ban đầu.



* Với biến var ta sẽ không thể biết biến bị thay đổi khi nào nếu như ta có hàng triệu dòng code

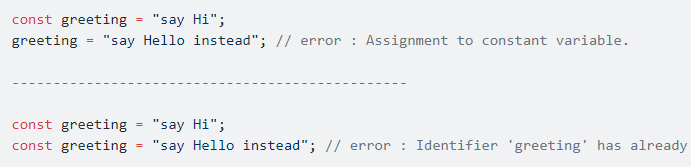
### Let

* Khi điều kiện xảy ra let sẽ chỉ cập nhật lại biến đó trong điều kiện thỏa mãn và khi thoát khỏi điều kiện nó không thể thay đổi hoàn toàn biến đó với giá trị ban đầu

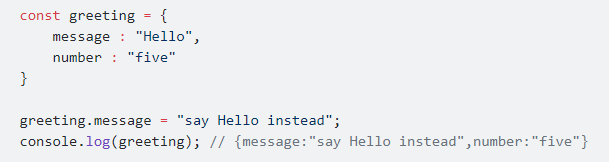
 

### Const

* Với kiểu biến là kiểu nguyên thủy thì ta không thể thay thế hay cập nhật giá trị mới



* Với kiểu biến là kiểu dữ liệu đối tượng thì ta chỉ có thể cập nhật giá trị cho thuộc tính của biến đó



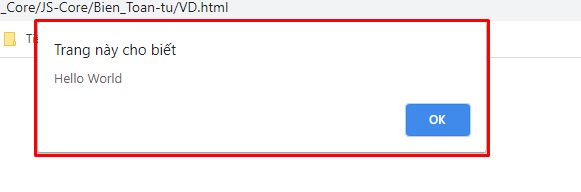
## Toán tử

Giống với cách khai báo toán tử ở các ngôn ngữ lập trình khác

# Các hàm cơ bản thường dùng

## Alert

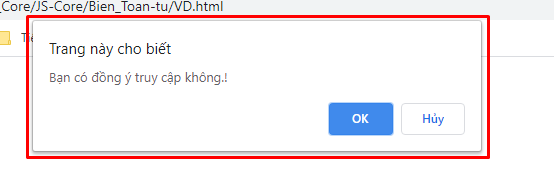




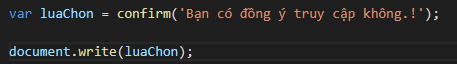
* Ứng dụng: Thông báo lỗi cho người dùng, thông báo cho người dùng chú ý hơn

## Confirm





* Ứng dụng: Thông báo cho người dùng lựa chọn
* Để lấy được thông tin lựa chọn của người dùng thông qua hàm confirm ta sẽ làm như sau:



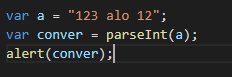
* Nếu chọn đồng ý thì trên tab website sẽ hiện thị là true ngược lại sẽ hiện false:

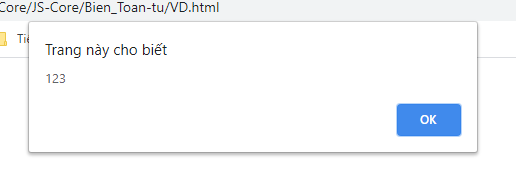


* Như vậy ta có thể dùng biến này để có thể xét các trường hợp của nó.

## ParseInt

* Cắt số nguyên đầu tiên của chuỗi



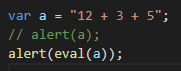


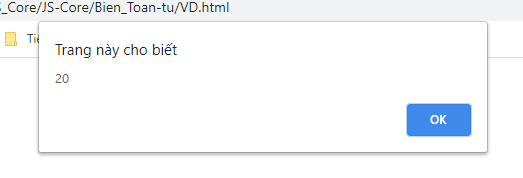
## ParseFloat

* Cũng giống như parseInt nhưng là dùng cho số thực

## Eval

* Biến chuỗi thành biểu thức toán học
* Eval có thể biến chuỗi thành các mã js

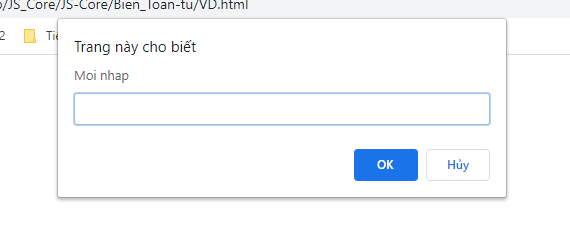




## Prompt

* Xuất hiện form cho phép người dùng nhập dữ liệu





## Set timeout

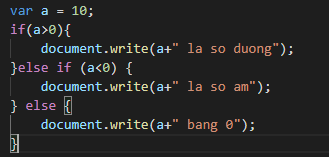
* Cho thực hiện câu lệnh sau một khoảng thời gian được tùy chỉnh

## Set interval

* Thực hiện câu lệnh liên tục và cách nhau một khoảng thời gian

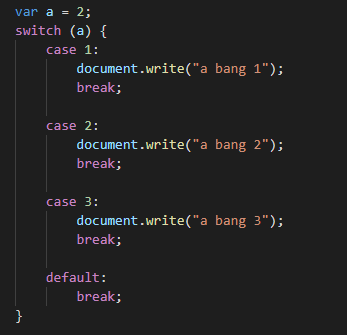
# Biểu thức điều kiện

## If else





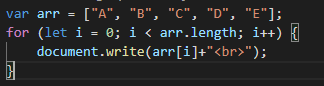
## Switch case





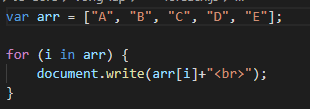
# Vòng lặp

## For

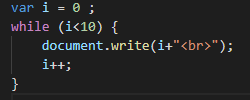




## Foreach

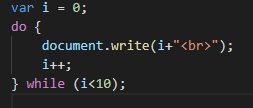
  


## While





## Do while

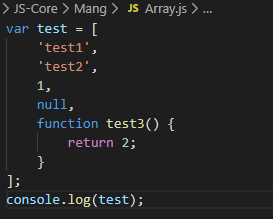


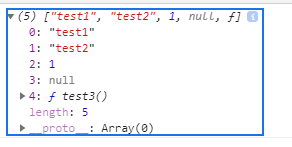


# Mảng

* Mảng là một biến có nhiều giá trị.
* Giá trị trong một mảng có thể là tập hợp gồm nhiều kiểu dữ liệu
* Chỉ mục trong mảng sẽ được đánh tự động

## Tạo mảng





## Truy xuất mảng

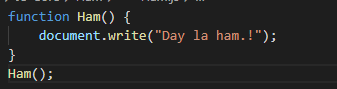




# Hàm

* Hàm là một khối mã và chỉ làm việc cụ thể

## Tạo hàm

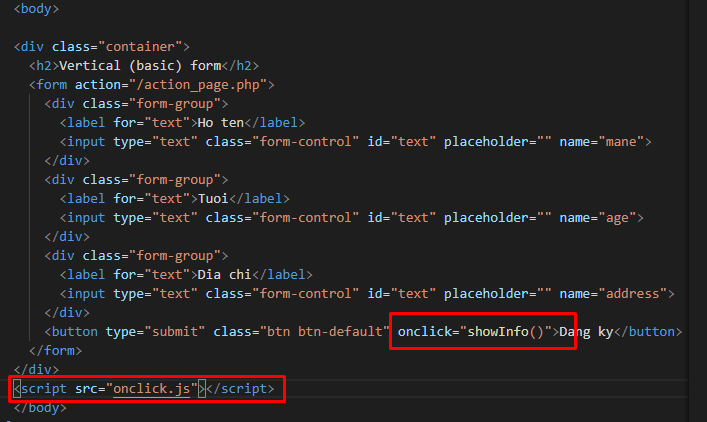


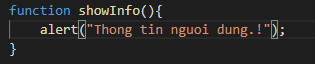


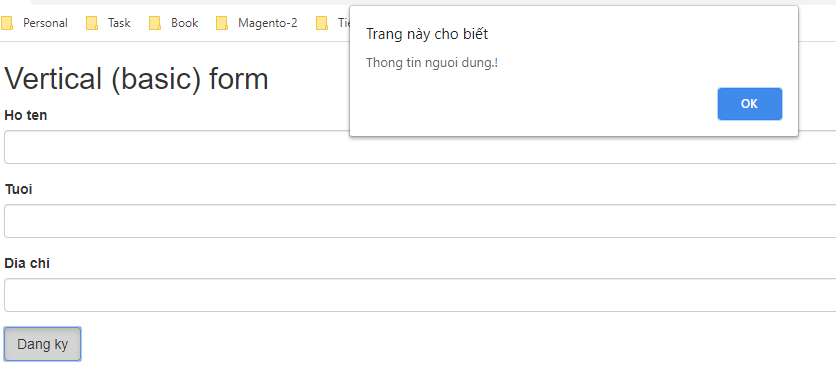
# Sự kiện

## Onclick

* Bắt sự kiện click chuột



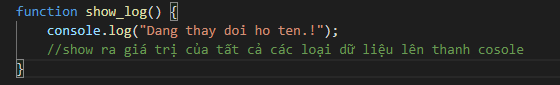


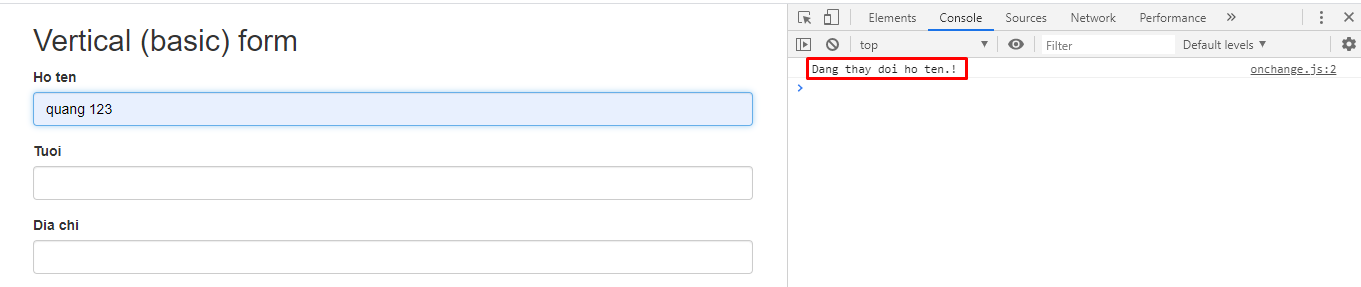


## Onchange

* Bắt sự kiện thay đổi giá trị textfield



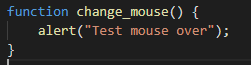
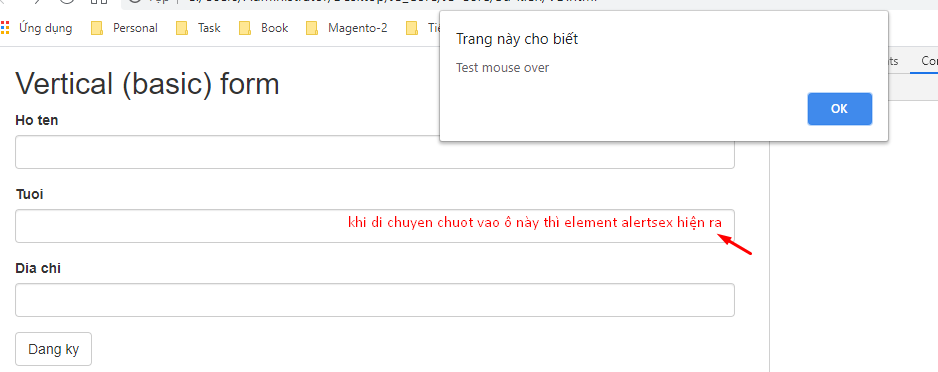




## Onmouseover

* Bắt sự kiện khi di chuyển chuột đến



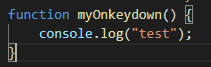
## Onmouseout

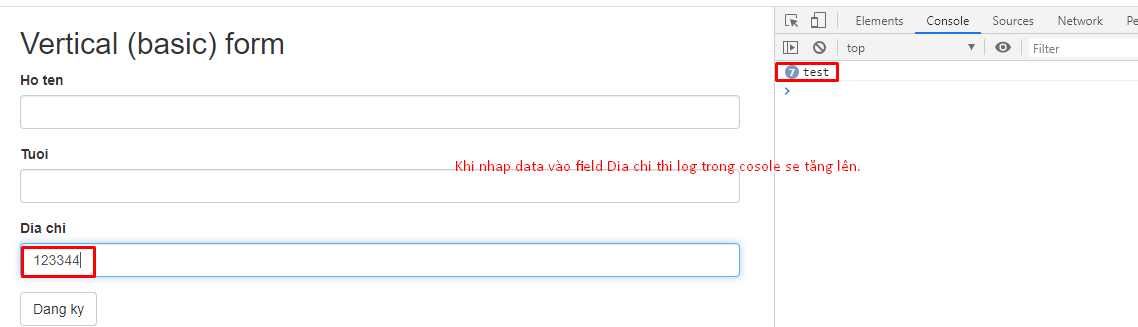
* Bắt sự kiện khi di chuyển chuột đi. Khai báo giống như Onmouseout

## Onkeydown

* Đếm số lượt click, thêm text vào field, …

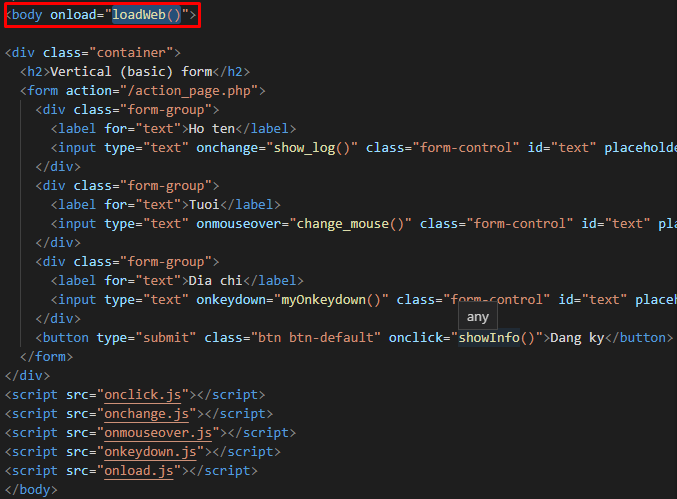


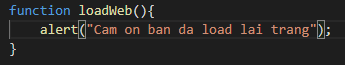


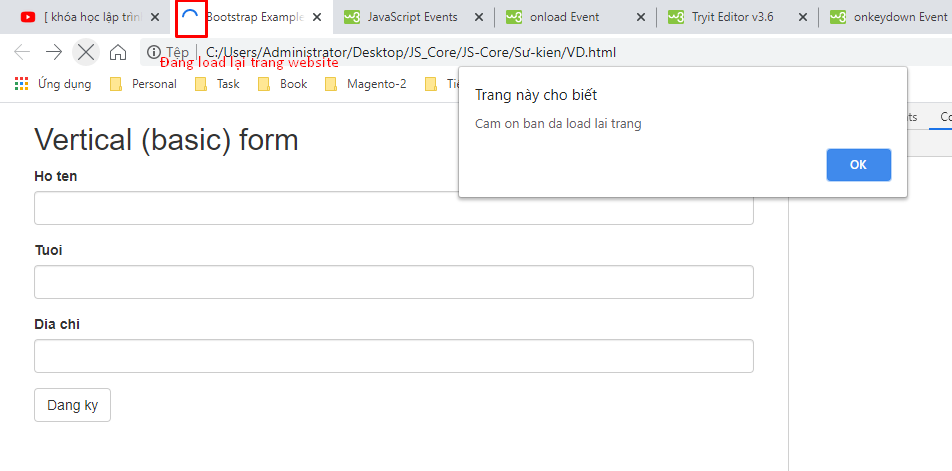


## Onload

* Sự kiện được thực hiện khi ta load lại trang website







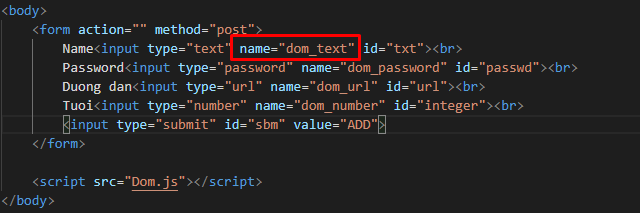
# DOM

* Nếu muốn truy xuất bất cứ yếu tố nào trong trang HTML thì ta dùng tới DOM trong Javascript

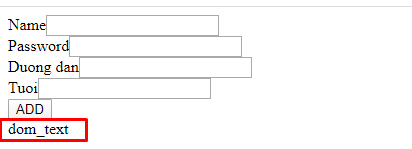
## Các thẻ tìm kiếm phần tử HTML

### Documment.getElementById(“id”)

* Tìm kiếm 1 element HTML theo id. Nó sẽ trả về giá trị là tên của thuộc tính đó

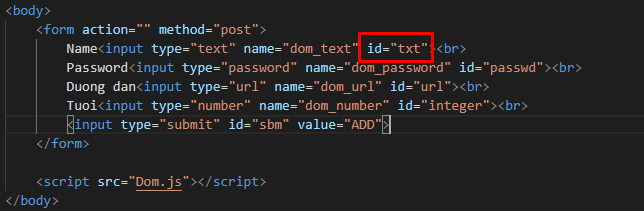




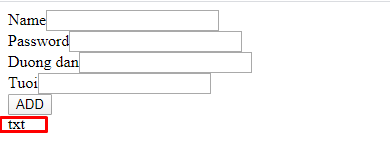


### Documment.getElementByTagName(“tag\_name”)

* Tìm kiếm các element được khai báo theo tên. Nó sẽ tìm kiếm tất cả các thẻ có cùng tên

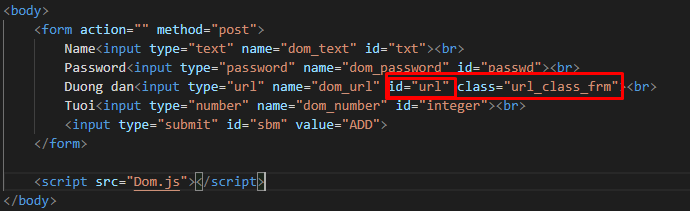




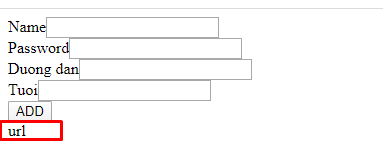


### Documment.getElementByClassName(“class\_name”)

* Tìm kiếm các element được khai báo theo tên class. Nó sẽ tìm kiếm tất cả các thẻ có cùng tên class



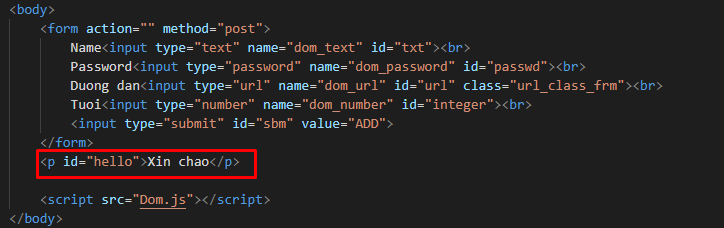




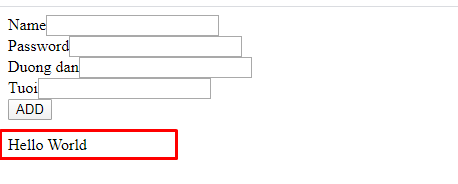
## Các thẻ thay đổi giá trị HTML

### innerHTML

* Thẻ này dùng để thay đổi giá trị của thẻ HTML

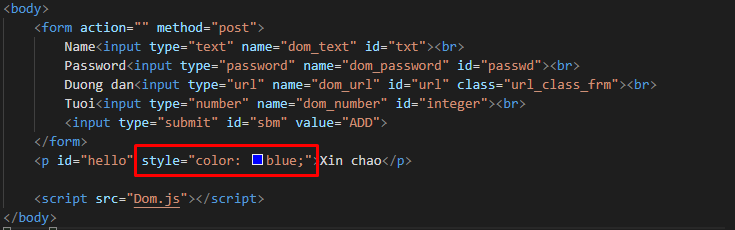




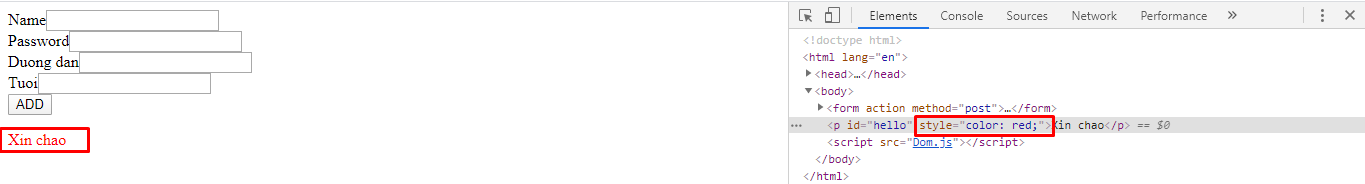


### Attribute

* Thẻ này dùng để thay đổi giá trị của thuộc tính

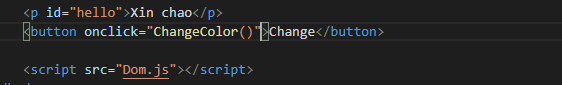


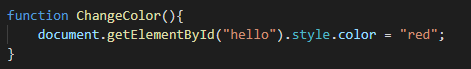




### Style.property

* Thẻ này dùng để thay đổi style của thẻ html





Trước khi nhấn buttom

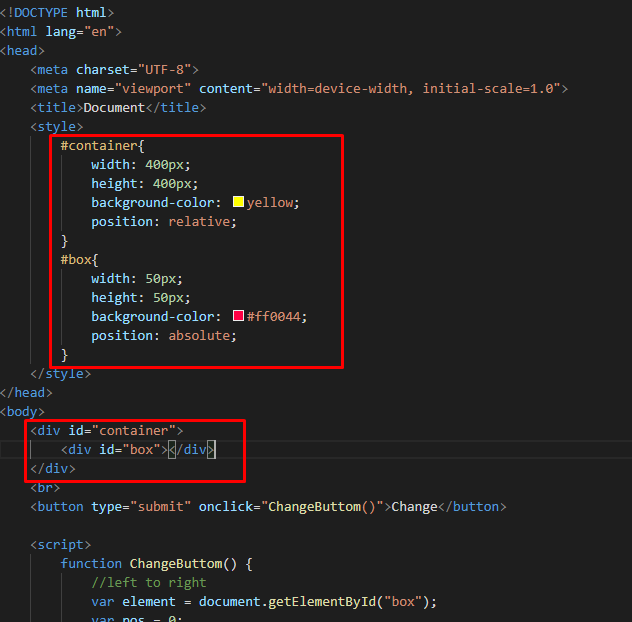


Sau khi nhấn buttom



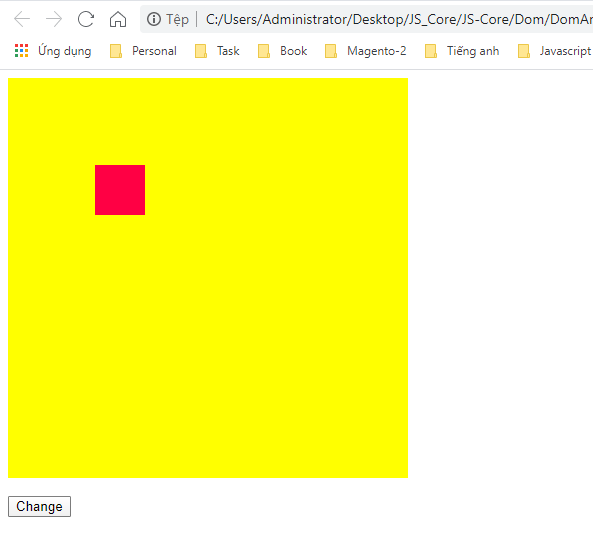
## Một vài ví dụ khác

### DOM Animations

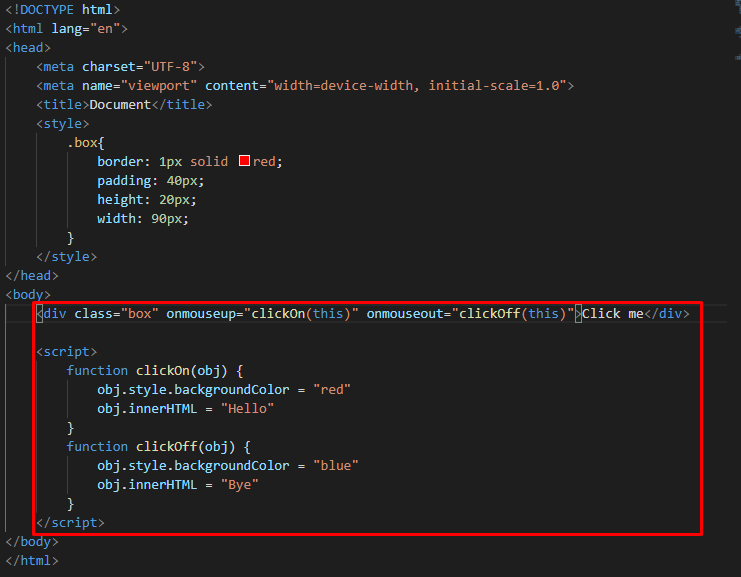




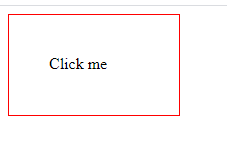
* Sau khi nhấn vào change thì box màu đỏ sẽ di chuyển từ góc trên bên trái xuống góc dưới bên phải



## Từ khóa this



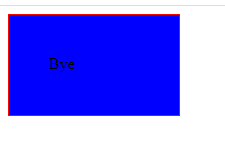
* Khi mới vào lần đầu tiên



* Khi click vào



* Khi chi chuột ra ngoài sau khi click



# Browser BOM

* 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 sẽ không có tiêu chuẩn nào cả.

## JS Window

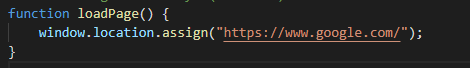
### Screen

* Screen.width, screen.height : Trả về giá trị chiều rộng, chiều cao màn hình của máy khách
* Screen.availWidth, Screen.availHeight : Trả về giá trị chiều rộng, chiều cao màn hình của máy khách khi đã trừ đi thanh taskbar

### Window.location

* Window.location.href : Hiển thị địa chỉ url
* Window.location.hostname : Hiển thị tên trang website
* Window.location.pathname : Hiển thị đường dẫn file của trang html
* Window.location.protocol : Hiển thị tên giao thức của website (http or https)
* Window.location.assign(“url”) : Nhúng đường dẫn load page





|  |  |
| --- | --- |
|  | Khi bấm vào buttom google nó sẽ chuyển sang trang google.com |
|  | |

### Window.history

### Window.navigator

### Window.alert

* Cách hoạt động giống với hàm alert()
* Ngoài ra còn có các hàm: window.confim(), window.prompt()

### Timing Event

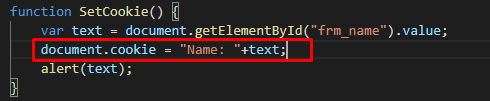
* Cho phép chúng ta thiết lập thời gian cho website
* Window.setTimeout(giaTri, time) : Xác định thời gian để giá trị được thực thi rồi dừng lại. Có hai biến được truyền vào hàm:
  + Giá trị: Giá trị được truyền vào để thực thi
  + Thời gian: Thời gian từ lúc chạy browse đến lúc thực thi câu lệnh
* Window.clearTimeout(giaTri) : Thực hiện dừng lệnh setTimeout\
* Window.setInterval(giaTri, time): Giống như setTimeout là cũng xác định thời gian để giá trị được thực thi nhưng sẽ lặp lại hành động sau một khoảng thời gian đã thiết lập
* Window.clearInterval (giaTri) : Tương tự như hàm clearTimeout nhưng được áp dụng để dừng hàm setInterval

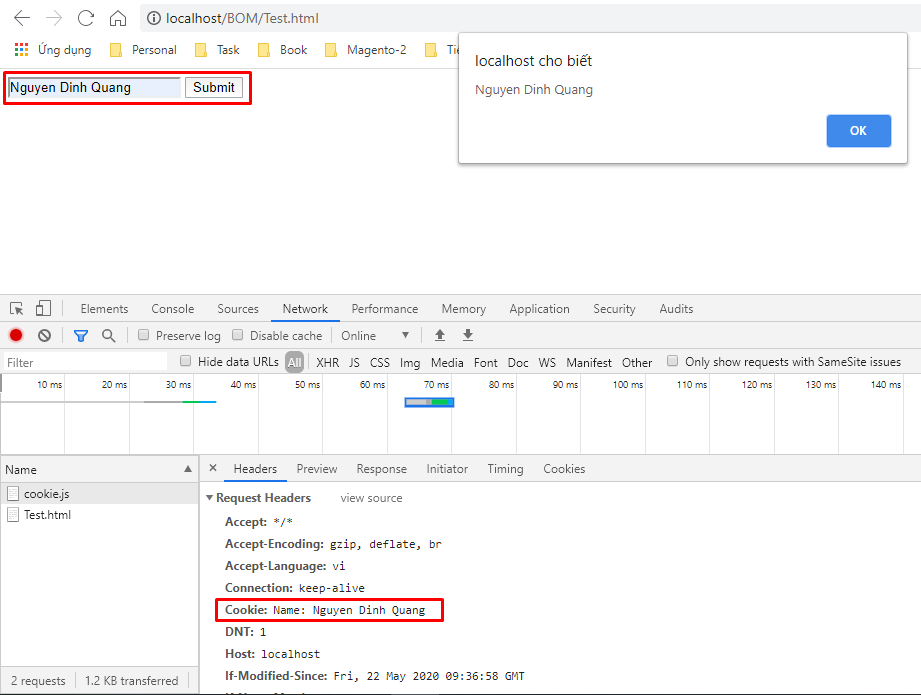
## Cookies

### Định nghĩa

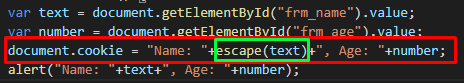
* Cookie cho phép lưu thông tin người dùng trong trang web
* Khi người dùng truy cập trang web thông tin của họ sẽ được lưu lại và khi họ quay trở lại vào lần sau thì cookie sẽ nhớ thông tin của họ
* Điều này giúp trang web được xử lý nhanh hơn và người dùng sẽ không phải khai báo thông tin của mình nhiều lần

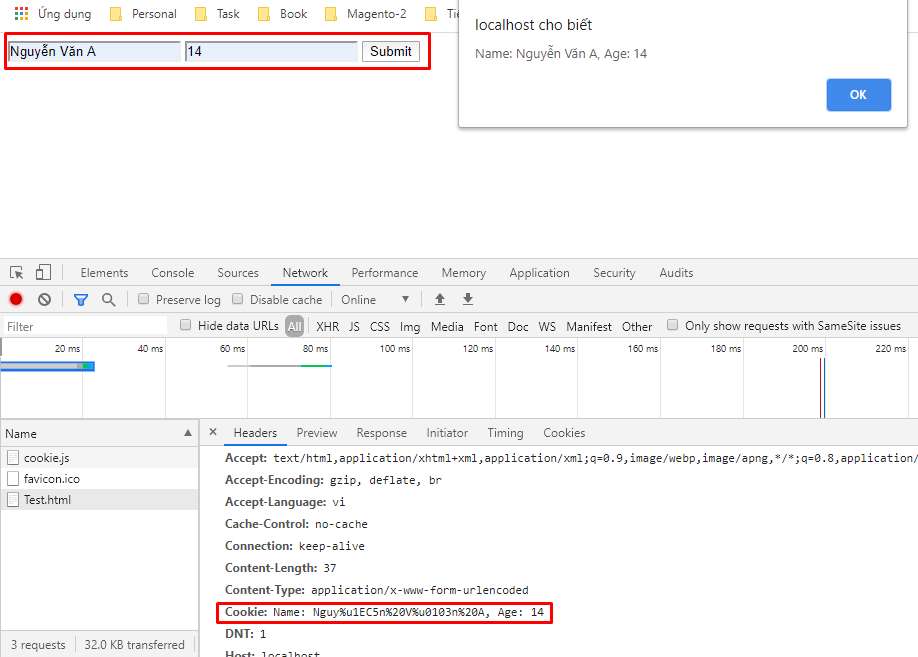
### Ghi cookie



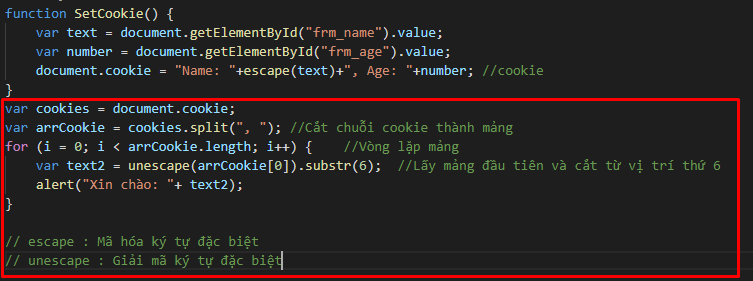


* Để lưu cookie những dữ liệu kiểu chữ tiếng việt hoặc khoảng trắng ta cần mã hóa chúng

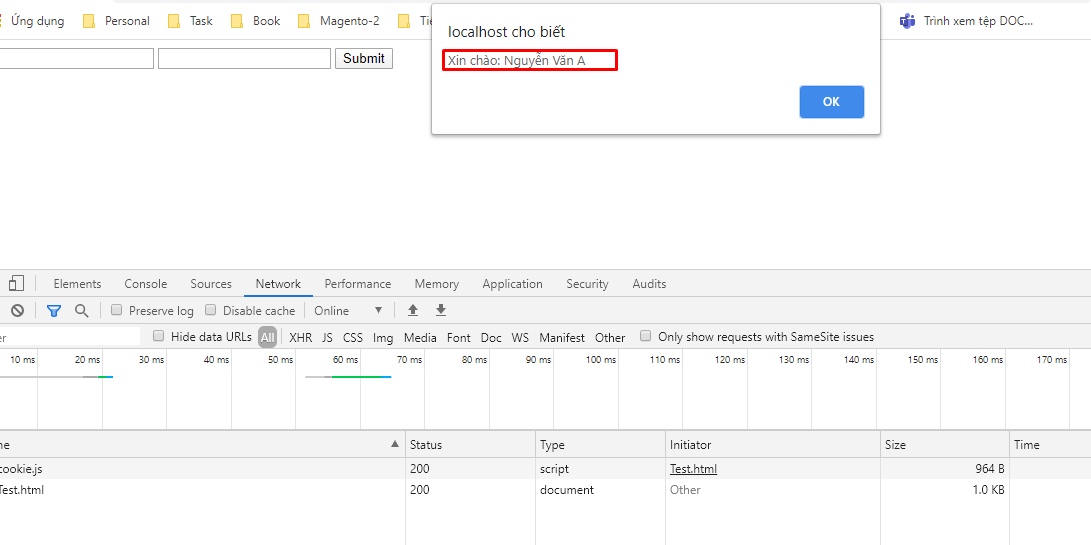




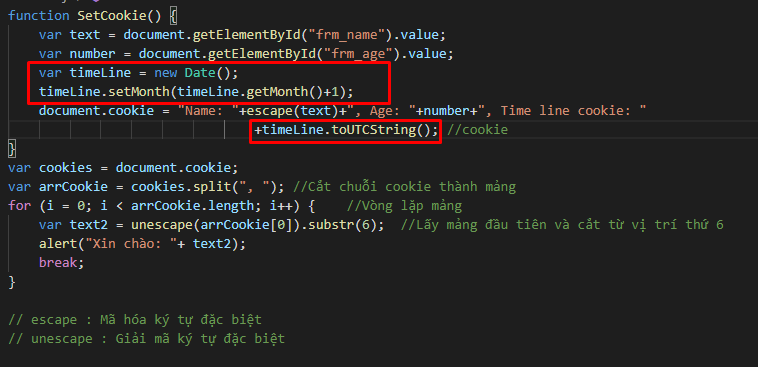
### Đọc cookie

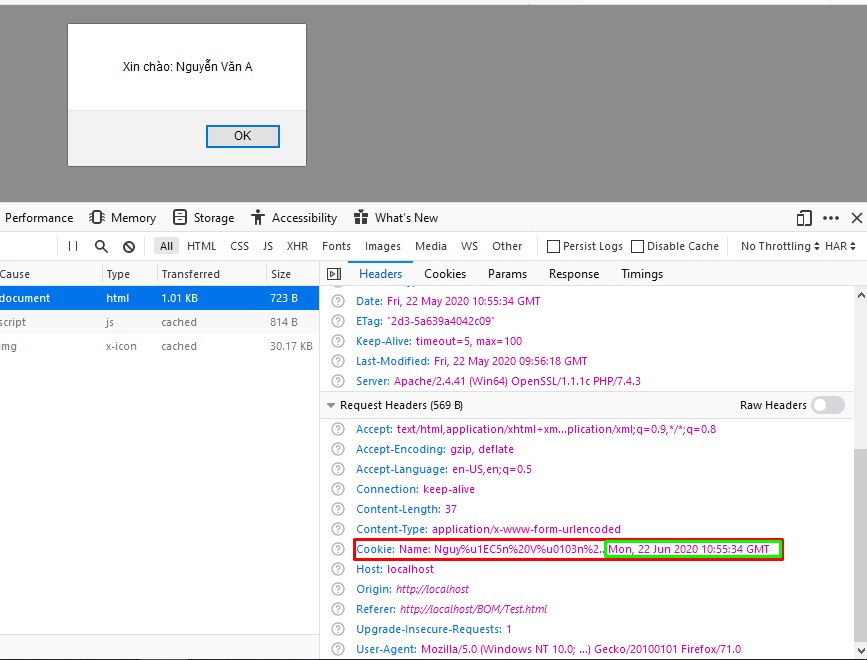


* Khi load lại trang web thì ta có:



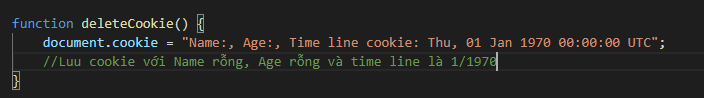
### Thiết lập thời gian sống



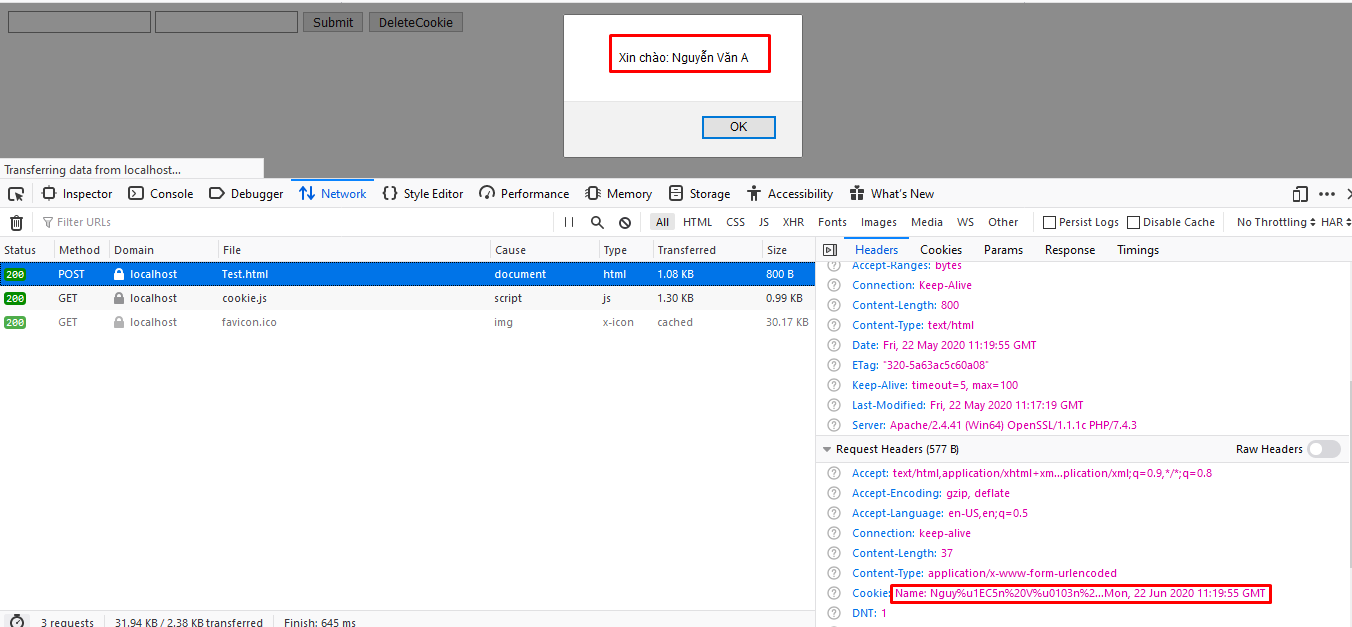


* Khi ta thiết lập lại ngày cho máy tính thì cookie sẽ ko còn tồn tại.

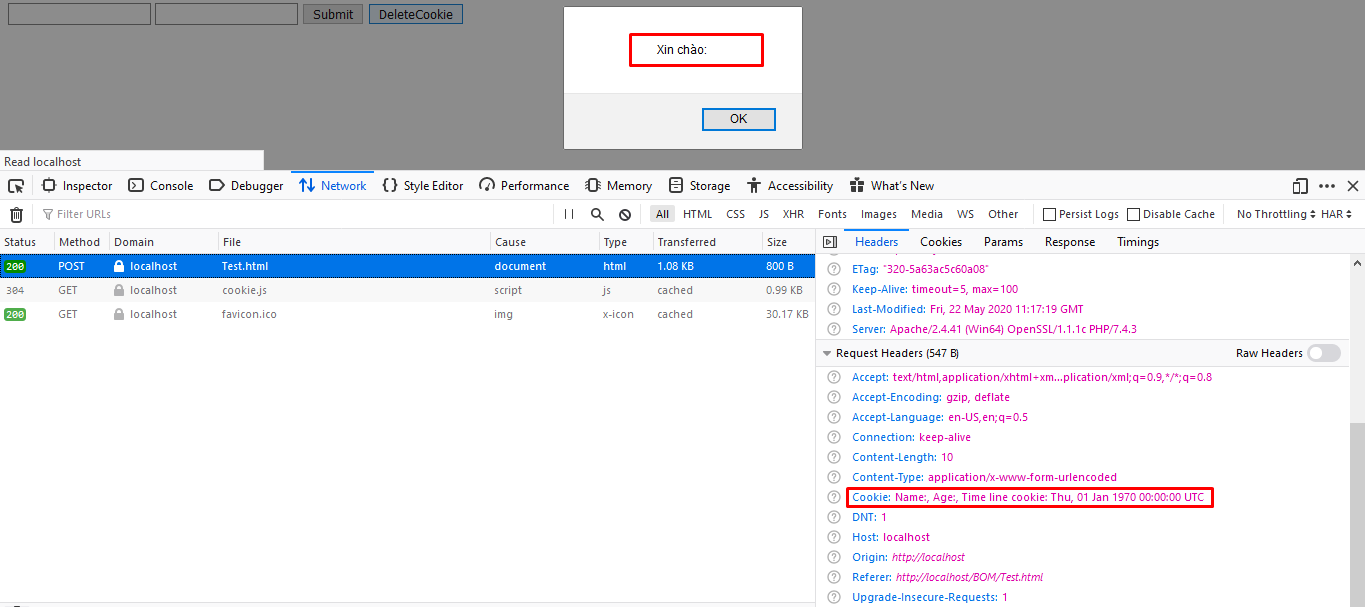
### Xóa cookie



* Khi chưa xóa:



* Khi đã xóa

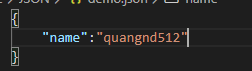


# JSON

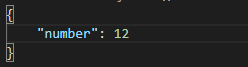
* Javascript Object Notation là một cú pháp để lưu trữ và trao đổi dữ liệu. JSON là văn bản được viết bằng ký hiệu đối tượng JavaScript.

## Cú pháp

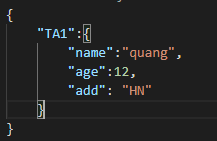
* Chuỗi: Phải được đặt trong dấu ngoặc kép



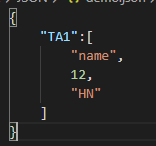
* Số: Không cần phải có dấu ngoặc



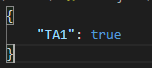
* Một đối tượng:



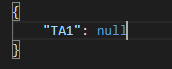
* Một mảng



* Boolean



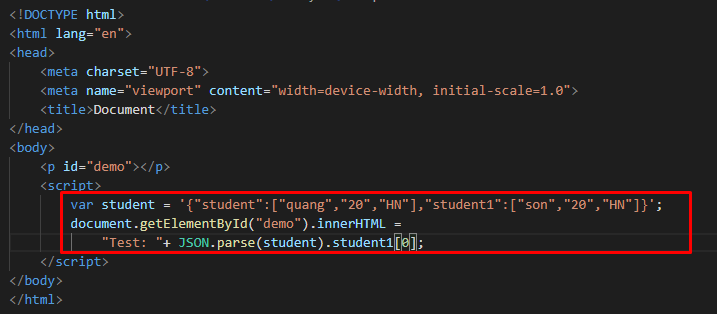
* Null



## Các hàm xử lý JSON

### JSON.parse()

* Dùng để chuyển đổi văn bản thành đối tượng trong Javascript





### JSON.Stringify()

* Hàm chuyển đổi object javascript thành kiểu JSON

