Fpt University

FPT POLYTECHNIC

PHẦN 1

LẬP TRÌNH JAVASCRIPT

BÀI 4: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG VÀ MÔ HÌNH BOM

www.poly.edu.vn

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

- Đối tượng là tất cả mọi thứ trong cuộc sống (các đồ vật, sự vật)
  - Ví dụ đối tượng: Quả bóng, cái bàn, ôtô, bông hoa, con người, nhà máy…
- Mỗi đối tượng có đặc tính và hành động riêng
- Ý tưởng chủ đạo của phương thức lập trình hướng đối tượng: Mô phỏng cuộc sống thực trong lập trình
  - Trong cuộc sống có những đối tượng như quả bóng, cái bàn… với các đặc tính và hành động riêng thì lập trình mô phỏng các đối tượng đó với các đặc tính và hành động như thế

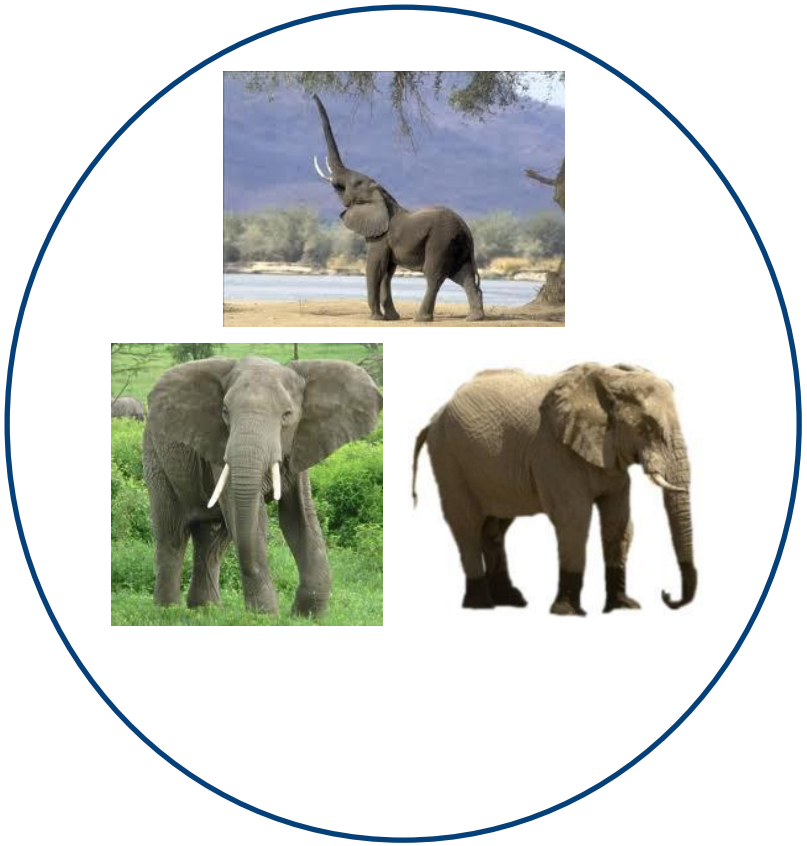
THUỘC TÍNH VÀ PHƯƠNG THỨC

- Trong lập trình: đặc tính được gọi là thuộc tính, hành động được gọi là phương thức

- Mèo có những đặc tính:
  - Màu lông: tam thể
  - Nặng: 2 kg
  - Móng: sắc
- Mèo có những hành động:
  - Bắt chuột
  - Liếm lông

- Voi có những đặc tính:
  - Màu da: nâu
  - Nặng: 2 tấn
  - Vòi: 1m
- Voi có những hành động:
  - Phun nước
  - Ăn cỏ

- Các đối tượng có cùng thuộc tính và phương thức được gom lại thành một lớp
- Hay: Lớp định nghĩa tập hợp các đối tượng có cùng thuộc tính và phương thức



TẠO ĐỐI TƯỢNG

- Cách tạo đối tượng trong javascript:
  - Object literals
  - Object constructor functions

SỬ DỤNG OBJECT LITERALS

- Object literals được biểu diễn bằng dấu phẩy ngăn cách giữa các cặp **name**:**value** nằm trong ngoặc nhọn {}
- Thuộc tính **name** thuộc kiểu dữ liệu dạng chuỗi (strings) và **value** là một trong số bất kỳ kiểu dữ liệu nào của javascript như mảng (arrays), chuỗi (strings), số (number) hay hàm (function)...
- Sử dụng object literals chúng ta có thể gom các giá trị (properties) và phương thức (method) vào cùng nhau giúp code sạch và dễ đọc hơn.

SỬ DỤNG OBJECT LITERALS

- Ví dụ:

```
var alex = {
    idCard: 16323891,
    name: "Alex Smith",
    age: 45,
    isMarried: true,
    childs: [
        {
            name: "James Smith",
            age: 13
        },
        {
            name: "Sarah Smith",
            age: 7
        }
    ],
    buyHouse: function(){
        console.log('already own a house in Pennsylvania');
    }
}
```

SỬ DỤNG OBJECT LITERALS

- Hiển thị giá trị thuộc tính/gọi phương thức

```
> // lấy giá trị của thuộc tính
  console.log(alex.name); console.log(alex.age);
  // thực thi phương thức của đối tượng alex
  alex.buyHouse();
  Alex Smith
  45
  already own a house in Pennsylvania
```

SỬ DỤNG OBJECT LITERALS

- Thay đổi giá trị của thuộc tính

```
// sửa giá trị của thuộc tính
console.log(alex.age);
alex.age = 48;
console.log(alex.age);
// thêm giá trị cho thuộc tính dạng mảng
console.log(alex.childs);
var newChild = {name: "Jim Smith", age: 1};
alex.childs.push(newChild);
console.log(alex.childs);
```

```
45
48
(2) [{…}, {…}]
  0: {name: "James Smith", age: 13}
  1: {name: "Sarah Smith", age: 7}
  2: {name: "Jim Smith", age: 1}
  length: 3
  __proto__: Array(0)
(3) [{…}, {…}, {…}]
  0: {name: "James Smith", age: 13}
  1: {name: "Sarah Smith", age: 7}
  2: {name: "Jim Smith", age: 1}
  length: 3
  __proto__: Array(0)
```

SỬ DỤNG OBJECT LITERALS

- Bên cạnh việc tạo object bằng ký tự {} thì chúng ta có thể tạo bằng câu lệnh new Object().

```
var rex = new Object();
rex.name = 'Rex Kumar';
rex.age = 23;
rex.gender = "male";
rex.goShopping = function(){
    console.log(this.name + ' is going to Super market');
}
console.log(rex);
rex.goShopping();
```

OUTPUT

```
▼ Object i
    age: 23
    gender: "male"
  ▶ goShopping: f ()
    name: "Rex Kumar"
  ▶ __proto__: Object
Rex Kumar is going to Super market
```

SỬ DỤNG OBJECT LITERALS

- Luyện tập: Tạo một đối tượng sinh viên bằng phương pháp sử dụng object literals, đối tượng bao gồm các thuộc tính/phương thức sau
  - ma_sinh_vien
  - ho_va_ten
  - diem
  - ketQua()
    - Nếu điểm >= 5: Đỗ, không thì Trượt

OBJECT CONSTRUCTOR FUNCTIONS

- **Object constructor functions** hay tạo khuôn mẫu một đối tượng bằng cách sử dụng hàm (khởi tạo). Đây là một trong những cách thông dụng nhất để tạo ra một object
- Cách tạo đối tượng bằng Object constructor functions khá giống với các ngôn ngữ backend như Java, C#, PHP,... khác ở chỗ sử dụng function thay cho class.

OBJECT CONSTRUCTOR FUNCTIONS

- Tạo đối tượng sử dụng Object constructor functions

```
function Animal(name, age, gender){
    this.name = name;
    this.age = age;
    this.gender = gender;
    this.run = function(){
        console.log(this.name + " is running!");
    }
}

var duck = new Animal('Donald', 2, 'male');
duck.hairColor = 'yellow';
console.log(duck);
duck.run();
```

OUTPUT

```
▼Animal {name: "Donald", age: 2, gender: "male", hairColor: "yellow", run: f} i
    age: 2
    gender: "male"
    hairColor: "yellow"
    name: "Donald"
  ▶ run: f ()
  ▶ __proto__: Object
Donald is running!
```

Fpt University

FPT POLYTECHNIC

PHẦN 2

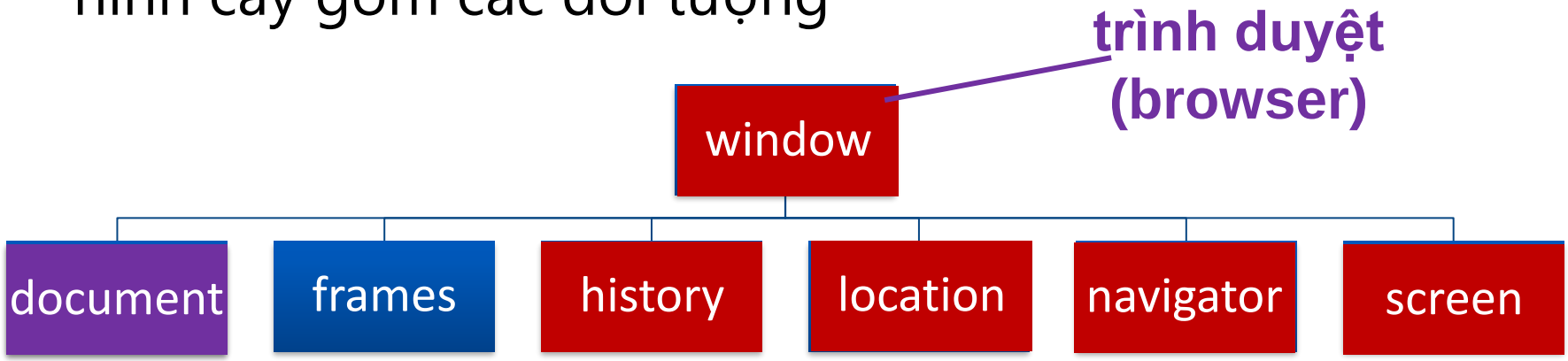
LẬP TRÌNH JAVASCRIPT

BÀI 4: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG VÀ MÔ HÌNH BOM

www.poly.edu.vn

BROWSER OBJECT MODEL

- Browser Object Model là một hệ thống phân cấp hình cây gồm các đối tượng



- Các đối tượng cung cấp thuộc tính và phương thức cho lập trình viên JavaScript
- Đối với mỗi đối tượng, mỗi trình duyệt hỗ trợ các thuộc tính và phương thức khác nhau
  - Hiểu môi trường mà trình duyệt cung cấp để viết mã JavaScript chạy ổn định trên nhiều trình duyệt

ĐỐI TƯỢNG WINDOW

- Window là đối tượng thể hiển cửa số hiển thị hiện tại trên trình duyệt
- Một số phương thức của đối tượng window đã được sử dụng: alert(), prompt(), confirm()
- Các thuộc tính và phương thức của window có thể gọi trực tiếp hoặc thông qua window

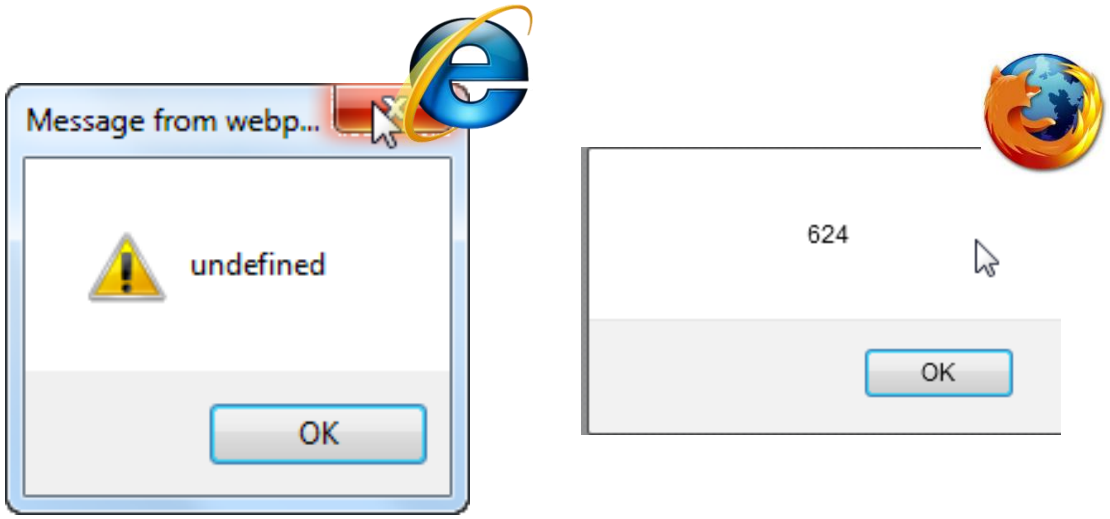
```
alert("Hi")
hoặc
window.alert("Hi")
```

CÁC THUỘC TÍNH CỦA WINDOW

Thuộc tính	Giải thích
closed	Có giá trị là True khi cửa sổ được đóng
defaultStatus	Thiết lập văn bản mặc định trên thanh trạng thái của trình duyệt
name	Thiết lập hoặc trả về tên của cửa sổ
opener	Tham chiếu đến cửa sổ tạo ra cửa sổ hiện tại
status	Thông tin xuất hiện trên thanh trạng thái
innerHeight	Thiết lập hoặc trả về chiều cao phần nội dung của cửa sổ

DEMO SỬ DỤNG THUỘC TÍNH INNERHEIGHT

```
alert(window.innerHeight);
```



- FireFox hỗ trợ thuộc tính này trong khi IE không hỗ trợ
- Tham khảo trang w3school để biết được trình duyệt nào hỗ trợ phương thức và thuộc tính nào

CÁC PHƯƠNG THỨC CỦA WINDOW

Phương thức	Giải thích
focus()	**Chuyển focus** đến cửa sổ
blur()	**Bỏ focus** đến cửa sổ
close()	**Đóng** cửa sổ
open()	**Mở** cửa sổ
print()	Thực hiện chức năng **in**
moveTo()	Sử dụng để **chuyển cửa sổ về vị trí** xác định
resizeTo()	**Thay đổi kích thước** cửa cửa sổ về vị trí xác định

WINDOW.OPEN

- Sử dụng để mở một cửa sổ từ cửa sổ hiện thời

  window.open(url, ten, dactinh)

  - url: url của trang web
  - ten: tên của cửa sổ sẽ mở
  - dactinh: các đặc tính mà cửa sổ được mở sẽ có (mỗi trình duyệt sẽ hỗ trợ một tập các đặc tính riêng)

  ```
  window.open("http://www.google.com.vn/", "timkiem",
  "menubar = yes, width = 800, height = 600")
  ```

Chú ý:
- Chỉ nên sử dụng cách này khi thật cần thiết vì trình duyệt có thể bị disable javascript
- Có thể sử dụng thẻ <a> để thay thế

ĐỐI TƯỢNG SCREEN

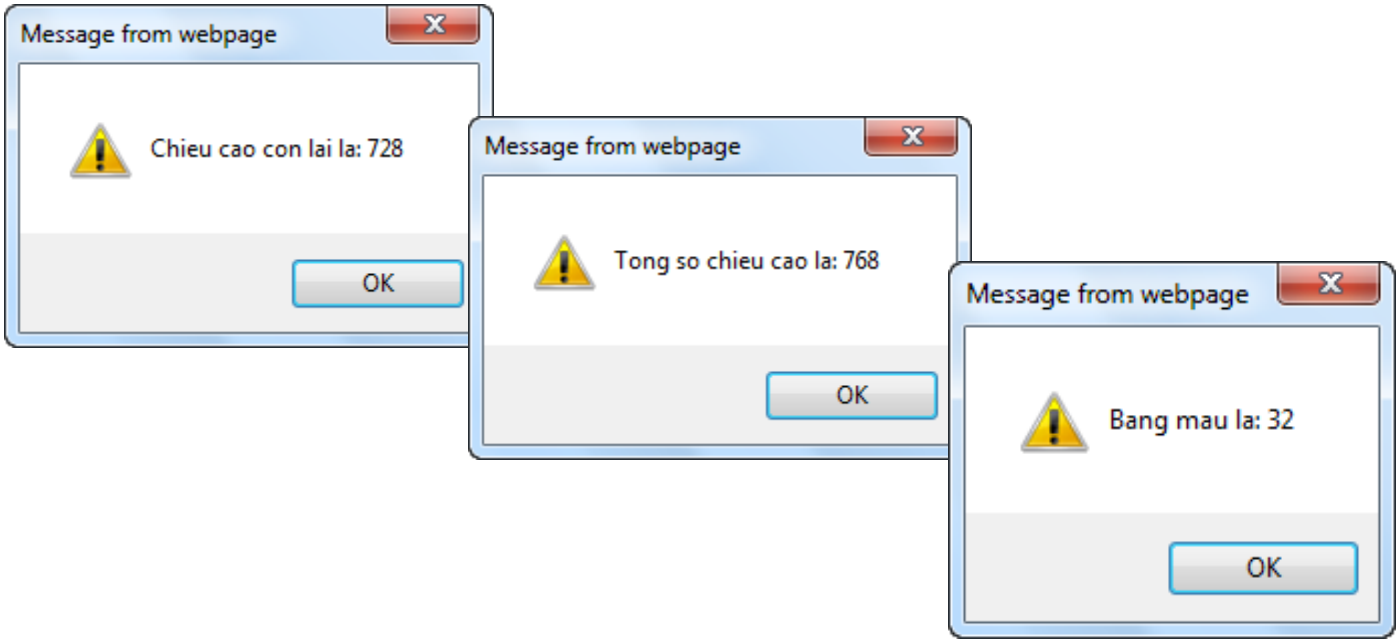
- Mỗi người truy cập sử dụng màn hình có độ phân giải khác nhau, kích thước khác nhau, dải màu khác nhau…
- ➔ Người lập trình phải nắm được thông tin này để hiển thị ảnh phù hợp, hiển thị trang web có kích thước phù hợp…
- Đối tượng screen cung cấp thuộc tính để lấy thông tin về màn hình của người truy cập

ĐỐI TƯỢNG SCREEN

Thuộc tính	Giải thích
availHeight	Trả về **chiều dài** của màn hình (trừ kích thước của window taskbar)
availWidth	Trả về **chiều rộng** của màn hình (trừ kích thước của window taskbar)
height	Trả về **chiều dài** của màn hình
with	Trả về **chiều rộng** của màn hình
pixelDepth	Trả về **độ phân giải** của màn hình
colorDepth	Trả về **bảng màu** để hiển thị ảnh

DEMO THUỘC TÍNH CỦA MÀN HÌNH

```
alert("Chieu cao con lai la: " + screen.availHeight);
alert("Tong so chieu cao la: " + screen.height);
alert("Bang mau la: " + screen.colorDepth);
```



ĐỐI TƯỢNG NAVIGATOR

- Mỗi trình duyệt có cách thức thi hành mã JavaScript riêng
- Có thể cùng thực hiện một chức năng, nhưng đối với từng trình duyệt, cần phải viết các đoạn mã khác nhau
- ➔ Cần biết thông tin về trình duyệt để viết mã JavaScript phù hợp
- Đối tượng Navigator cung cấp các thông tin về trình duyệt

Thuộc tính

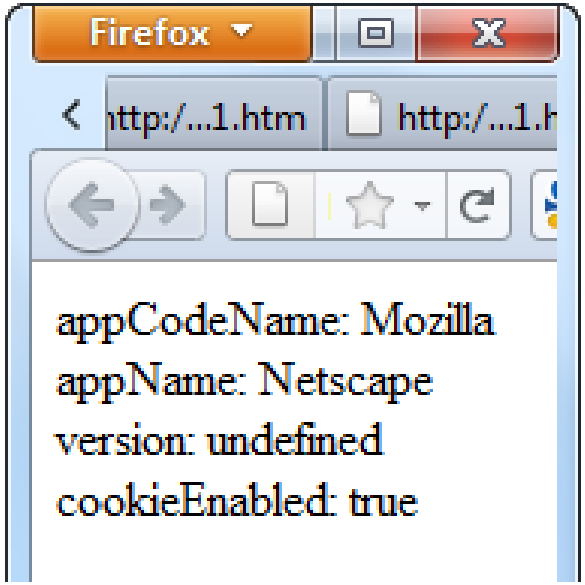
Thuộc tính	Giải thích
appCodeName	Trả về **mã** của trình duyệt
appName	Trả về **tên** của trình duyệt
appVersion	Trả về **phiên bản** của trình duyệt
cookieEnabled	Xác định xem **Cookie** có được bật hay không
platform	Trả về **nền tảng** mà trình duyệt được biên dịch

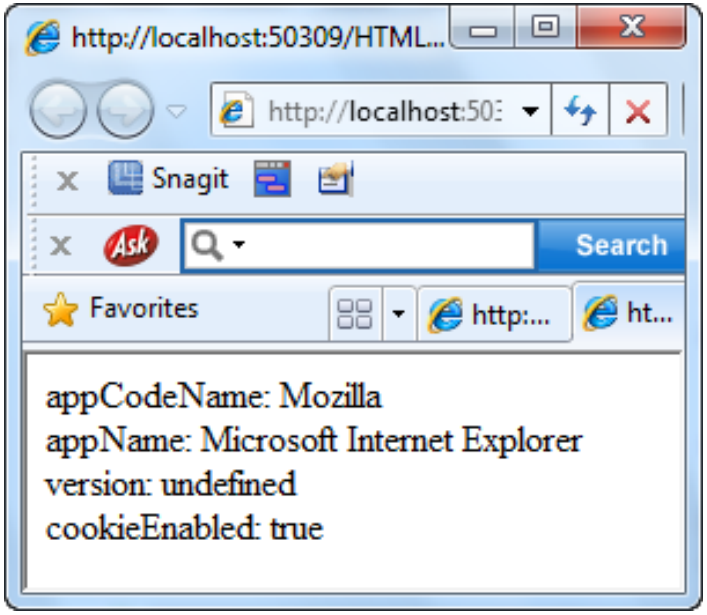
Phương thức

Phương thức	Giải thích
javaEnabled()	Xác định xem trình duyệt có kích hoạt Java hay không

DEMO SỬ DỤNG NAVIGATOR

```
document.write("appCodeName: " + navigator.appCodeName + "<br>");
document.write("appName: " + navigator.appName +"<br>");
document.write("version: " + navigator.version + "<br>");
document.write("cookieEnabled: " + navigator.cookieEnabled);
```





ĐỐI TƯỢNG LOCATION

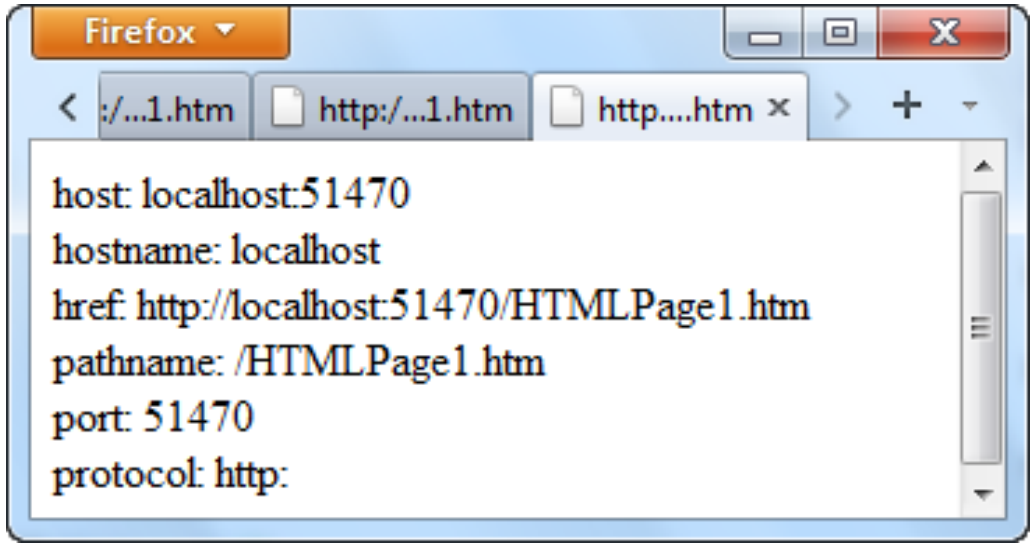
- Quản lý thông tin về URL hiện tại

Thuộc tính	Giải thích
host	Trả về tên **host** và **cổng** của URL
hostname	Trả về tên **host**
href	Trả về toàn bộ **URL**
pathname	Trả về tên **đường dẫn** của URL
port	Trả về **cổng** mà server sử dụng cho URL
protocol	Trả về **protocol** của URL

Phương thức	Giải thích
assign()	**Load** document mới
reload()	**Load lại** document hiện tại

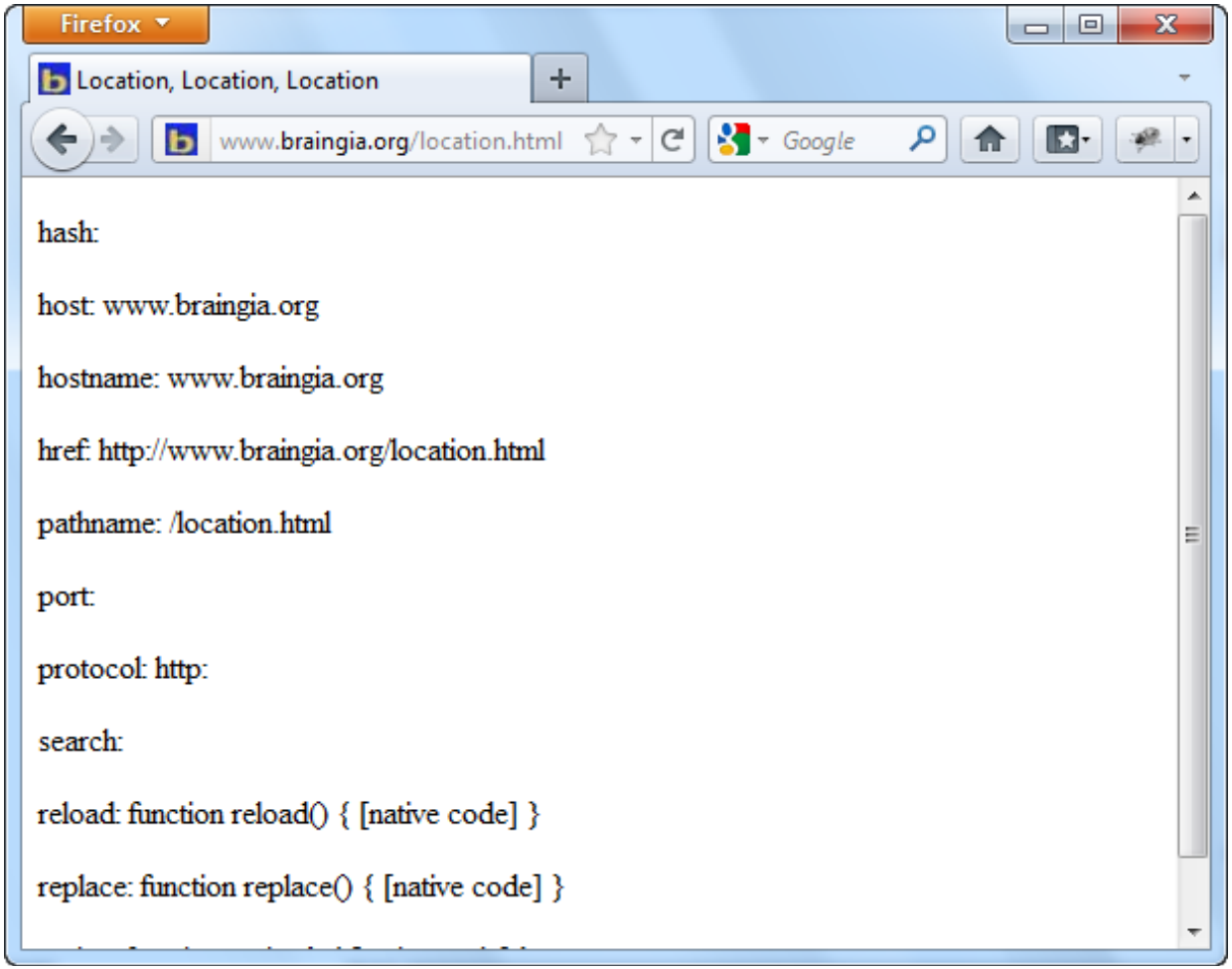
DEMO HIỂN THỊ THÔNG TIN CỦA URL

```
document.write("host: " + location.host + "<br>");
document.write("hostname: " + location.hostname + "<br>");
document.write("href: " + location.href + "<br>");
document.write("pathname: " + location.pathname + "<br>");
document.write("port: " + location.port + "<br>");
document.write("protocol: " + location.protocol + "<br>");
```



DEMO HIỂN THỊ THÔNG TIN CỦA URL

- Vào trang http://www.braingia.org/location.html



DEMO SỬ DỤNG LOCATION ĐỂ MỞ URL MỚI

```
<html >
<head>
<script type="text/javascript">
    function newDoc() {
        window.location.assign("http://www.w3schools.com")
    }
</script>
</head>
<body>
<input type="button" value="Load new document"
onclick="newDoc()" />
</body>
</html>
```

ĐỐI TƯỢNG HISTORY

- Chứa thông tin về các URL được người dùng truy cập

Thuộc tính	Giải thích
length	Trả về số lượng URL trong danh sách History

Phương thức	Giải thích
back()	Load URL **trước** đó trong danh sách History
forward()	Load URL **sau** đó trong danh sách History
go()	Load URL cụ thể từ History

Demo sử dụng History

- Định nghĩa hàm trong thẻ JavaScript

```
function goBack() {
    history.back();
}

function goNext() {
    history.forward();
}
```

- Gọi hàm

```
<p><a href = "#" onclick="goBack()">Back</a></p>
<p><a href = "#" onclick="goNext()">Next</a></p>
```