

BÀI THỰC HÀNH TUẦN 5&6: JavaScript

Mô tả

Mục đích của bài thực hành:

- ✚ Giúp cho người học nắm được kiến thức cơ bản và cách sử dụng JavaScript trong lập trình ứng dụng web.
- ✚ Nắm bắt các kỹ thuật và công nghệ tiên tiến liên quan như jQuery, AJAX.

Nội dung bài thực hành:

- ✚ Phần 1 – Tổng quan
- ✚ Phần 2 – Các đối tượng xây dựng sẵn
- ✚ Phần 3 – Các toán tử và cấu trúc lệnh
- ✚ Phần 4 – Xử lý lỗi
- ✚ Phần 5 – Xử lý sự kiện
- ✚ Phần 6 – Xử lý chuỗi
- ✚ Phần 7 – Xử lý công thức toán học
- ✚ Phần 8 – Xử lý thông số thời gian
- ✚ Phần 9 – Một số hàm thông dụng
- ✚ Phần 10 – Lập trình hướng đối tượng
- ✚ Phần 11 – jQuery
- ✚ Phần 12 – AJAX
- ✚ Bài tập
- ✚ Tài liệu tham khảo

Yêu cầu

- ✚ Đọc và hiểu lý thuyết liên quan.
- ✚ Thực hành và làm bài tập được giao.

Phần 1 – Tổng quan

JavaScript là một ngôn ngữ kịch bản được nhúng trực tiếp vào mã nguồn HTML thông qua một hay nhiều thẻ `<script></script>` hoặc sử dụng gián tiếp thông qua một hay nhiều file JavaScript. Các trình duyệt hỗ trợ JavaScript sẽ thông dịch và thực thi các đoạn mã JavaScript tại client. Khi sử dụng JavaScript, người lập trình có thể đặt script ở trong tab `<head>` của file HTML (đoạn script được thực thi trước khi trang web được hiển thị xong), trong phần `<body>` (đoạn script được thực thi khi trang web được loaded) hoặc trong một file .js bên ngoài như CSS.

JavaScript được sử dụng khá phổ biến trong lập trình ứng dụng web. Một số công dụng của JavaScript được mô tả tóm tắt như sau:

- ✚ Nhúng hoặc thay đổi nội dung vào trong trang HTML.
- ✚ Xử lý sự kiện.
- ✚ Tiền kiểm tra dữ liệu (validation).
- ✚ Kiểm tra các loại trình duyệt: Chrome, Firefox, IE, Opera, Safari, Netscape ...
- ✚ Lưu trữ thông tin ở client.

Phần 2 – Các đối tượng xây dựng sẵn của JavaScript

JavaScript hỗ trợ các đối tượng xây dựng sẵn tương tự như các kiểu dữ liệu trong các ngôn ngữ lập trình khác, sinh viên tự tìm hiểu cách sử dụng các đối tượng này.

- *Array*: được dùng để chứa nhiều giá trị vào một biến đơn
- *Boolean*: chứa giá trị true/false
- *Date*: được dùng để chứa các giá trị date và time
- *Math*: đối tượng Math có thể được sử dụng mà không cần hàm khởi tạo. Các hàm của đối tượng này có thể được dùng để thực hiện các phép tính toán cơ bản như min, max, sin, cos, khai căn...
- *Number*: chứa các giá trị số
- *String*: chứa các giá trị text
- *RegExp*: hỗ trợ thực hiện các chức năng pattern-matching và search-and-replace
- *Global*: các thuộc tính và hàm được khai báo global có thể được sử dụng với bất kỳ đối tượng nào của JavaScript

Phần 3 – Các đối tượng xây dựng sẵn của trình duyệt

Một số đối tượng được xây dựng sẵn trong JavaScript:

Window

Đối tượng window đại diện cho một cửa sổ của trình duyệt. Một vài phương thức cơ bản:

- *Open(URL,name,specs,replace)*: mở một cửa sổ trình duyệt mới. Trong đó:
 - o *URL* là URL của cửa sổ sẽ được mở, nếu URL rỗng thì một trang blank sẽ được mở. (optional)
 - o *Name* xác định thuộc tính target hoặc tên của cửa sổ. Ta có thể xác định mở một cửa sổ mới (`_blank`), load cửa sổ tại parent frame (`_parent`), load tại

trang hiện tại (`_self`) hoặc chỉ định tên cho cửa sổ này (giá trị name xác định) (optional)

- o *Specs* xác định một tập các thuộc tính cho cửa sổ được mở (height, width, fullscreen...)

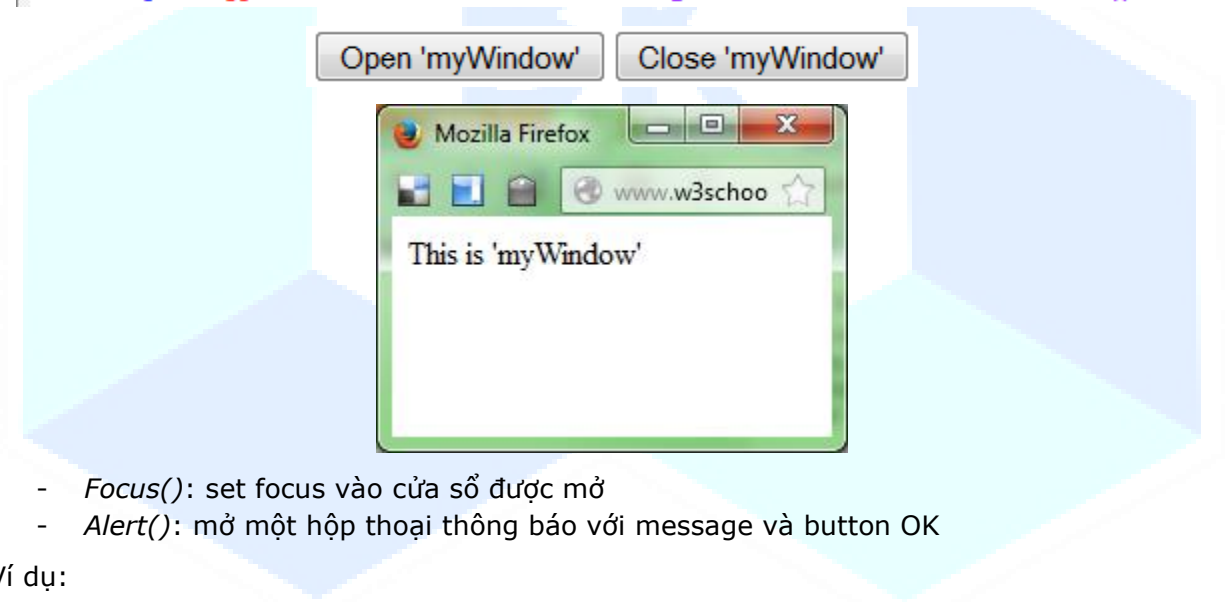
- `Close()`: đóng cửa sổ trình duyệt

Ví dụ:

```
<script>
    function openWin()
    {
        myWindow=window.open("", "", "width=200,height=100");
        myWindow.document.write("<p>This is 'myWindow'</p>");
    }

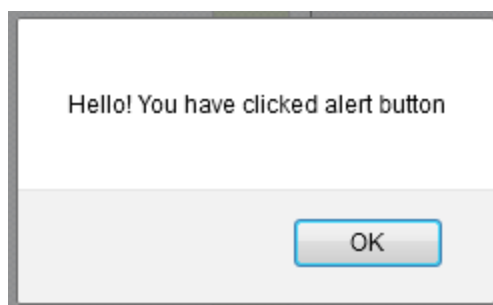
    function closeWin()
    {
        myWindow.close();
    }
</script>
</head>
<body>

<input type="button" value="Open 'myWindow'" onclick="openWin()" />
<input type="button" value="Close 'myWindow'" onclick="closeWin()" />
```



- `Focus()`: set focus vào cửa sổ được mở
- `Alert()`: mở một hộp thoại thông báo với message và button OK

Ví dụ:



- *Confirm()*: tương tự như alert nhưng hàm này sẽ hiển thị thông báo với button OK và Cancel. Hàm sẽ trả về true nếu người dùng chọn OK và false trong trường hợp ngược lại. Ví dụ

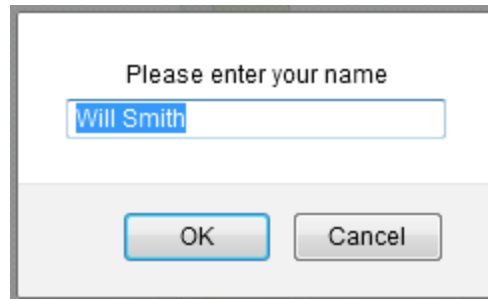
```
<p id="demo"></p>
<script>
function myFunction()
{
var x;
var r=confirm("Press a button!");
if (r==true)
{
x="You pressed OK!";
}
else
{
x="You pressed Cancel!";
}
document.getElementById("demo").innerHTML=x;
}
```

- *Prompt()*: hiển thị hộp thoại thông báo và yêu cầu người dùng nhập dữ liệu vào
- Ví dụ:

```
<p id="demo"></p>
<script>
function myFunction()
{
var x;

var person=prompt("Please enter your name","Will Smith");

if (person!=null)
{
x="Hello " + person + "! How are you today?";
document.getElementById("demo").innerHTML=x;
}
}
</script>
```



Navigator

Đây là đối tượng chứa các thông tin của trình duyệt:

- *CodeName*: tên codename của trình duyệt
- *AppName*: tên trình duyệt
- *CookiesEnabled*: xác định cookie đã được enable cho trình duyệt hay chưa?
- *Platform*: platform mà trình duyệt được hiện thực
- *UserAgent*: thông tin user-agent header mà trình duyệt gửi cho server

Ví dụ:

```
<pre>  
<script>  
document.writeln("CodeName: " + navigator.appCodeName);  
document.writeln("Name: " + navigator.appName);  
document.write("Version: " + navigator.userAgent);  
document.write("Cookies Enabled: " + navigator.cookieEnabled);  
document.write("Platform: " + navigator.platform);  
document.write("User-agent header: " + navigator.userAgent);  
</script>  
</pre>
```

Sinh viên tự thực hành để xem kết quả hiển thị, xem thêm tài liệu [9]

History

Đối tượng history chứa thông tin về các URL mà người dùng đã duyệt qua trên cửa sổ trình duyệt. Đối tượng history là một phần của đối tượng window nên việc truy xuất phải thực hiện qua đối tượng window. Bạn có thể xác định số URL mà người dùng đã duyệt qua (thông qua thuộc tính length) hoặc cung cấp chức năng back, forward và go (load một history page cụ thể). Ví dụ:

```
window.history.back();  
window.history.forward();  
window.history.go(-2);
```

Location

Location chứa các thông tin về URL hiện tại của cửa sổ trình duyệt. Bao gồm các thông tin cơ bản như host, hostname, href, port, protocol, search...

Ví dụ:

```
document.writeln("Hash:" + location.hash);  
document.writeln("Host:" + location.host);  
document.writeln("HostName:" + location.hostname);  
document.writeln("Href:" + location.href);  
document.writeln("PathName:" + location.pathname);  
document.writeln("Port:" + location.port);  
document.writeln("Protocol:" + location.protocol);  
document.writeln("Search:" + location.search);
```

Phần 4 – Các đối tượng HTML DOM

Document Object Model (DOM) là cách thể hiện các thành phần của một trang HTML dưới dạng node:

- Bản thân trang HTML là một document node
- Tất cả elements của HTML là element nodes
- Thuộc tính của các elements này là thuộc tính của node
- Text sử dụng bên trong element là text của node
- Chú thích trong file HTML là comment nodes

Document

Khi một trang web HTML được load vào trình duyệt, trang web này sẽ được biến đổi thành một đối tượng document. Đối tượng document được xem là node cha, cung cấp các thuộc tính và hàm để truy xuất tất cả các node con trong file HTML.

Ví dụ, để lấy một node thông qua thuộc tính id của node đó:

```
<p id="demo">Click the button to change the text in this paragraph.</p>  
<button onclick="myFunction()">Try it</button>  
<script>  
function myFunction()  
{  
document.getElementById("demo").innerHTML="Hello Guys";  
};  
</script>
```

Lấy các node có cùng tên tag là <p>

```
<p id="demo">Click the button to change the text in this paragraph.</p>  
<button onclick="myFunction()">Try it</button>  
<script>  
function myFunction()  
{  
document.getElementsByTagName("P")[0].innerHTML="Hello Guys";  
};  
</script>
```

Tạo một button với text "Click me!"

```
<script>
    var btn=document.createElement("button");
    var t=document.createTextNode("Click me!");
    btn.appendChild(t);
    document.body.appendChild(btn);
</script>
```

Trong ví dụ trên, ta tạo một element theo kiểu button, sau đó tạo một text node có giá trị là text cần hiển thị, text node là con của node button. Sau đó, chèn node button vào node cha, document.

Ngoài, ra ta cũng có thể truy xuất một element thông qua CSS selector của element này

```
<div id="foo\bar"></div>
<div id="foo:bar"></div>

<script>
document.querySelector('#foo\bar')    // Does not match anything
document.querySelector('#foo\\bar')  // Match the first div
document.querySelector('#foo:bar')    // Does not match anything
document.querySelector('#foo\\:bar')  // Match the second div
</script>
```

Phương thức `querySelector` sẽ trả về element đầu tiên có selector trùng với phần mô tả (trả về null trong trường hợp không có element nào thỏa yêu cầu). Lưu ý là phần chuỗi truyền vào cho phương thức phải bỏ qua các ký tự đặc biệt bằng cách thêm vào ký tự `\\`

Events

HTML DOM Events cho phép JavaScript xử lý các sự kiện khác nhau trên các elements của HTML document. Các sự kiện thường được sử dụng kết hợp với hàm; các hàm này sẽ không được thực thi trước khi sự kiện xảy ra. Một số sự kiện điển hình có thể kể đến như sau:

- *onclick*: sự kiện xảy ra khi người dùng click chuột lên element
- *ondblclick*: sự kiện người dùng double-click lên element
- *onmouseover*: khi người dùng trỏ chuột vào một element
- *onkeydown*: sự kiện người dùng đang chọn một phím trên bàn phím
- *onload*: sự kiện xảy ra khi một document, frameset hoặc object được loaded
- *onfocus*: sự kiện xảy ra khi một element của form được focus (áp dụng cho `<label>`, `<input>`, `<select>`, `textarea`, và `<button>`)
- *onselect*: sự kiện khi người dùng chọn một đoạn text trong element `<input>` và `<textarea>`

Sinh viên tự tìm hiểu thêm các sự kiện khác.

Form

HTML DOM Form là đối tượng chứa tất cả các element được định nghĩa trong HTML form. Để lấy được các elements này, ta có thể sử dụng kiểu `elements[]` như ví dụ sau:

```

<form id="frm1">
  First name: <input type="text" name="fname" value="E"><br>
  Last name: <input type="text" name="lname" value="Commerce"><br>
  <input type="submit" value="Submit">
</form>
<p>Get each element in the form:</p>
<script>
  var x=document.getElementById("frm1");
  for (var i=0;i<x.length;i++) //length returns number of elements in the form
  {
    document.write(x.elements[i].value);
    document.write("<br>");
  }
</script>

```

Ngoài ra, đối tượng form cũng cung cấp các thuộc tính và phương thức hỗ trợ (submit, reset), sinh viên tự tìm hiểu phần này.

FileUpload

Đối tượng File được dùng trong phần input của form, cho phép người dùng chọn và upload một file lên server. Một số thuộc tính tiêu biểu của đối tượng này:

- **accept**: biểu diễn danh sách các kiểu file hợp lệ. Tuy nhiên, thuộc tính này không dùng để đánh giá tính hợp lệ của các file này, việc đánh giá nên được thực hiện tại server. Ví dụ, để chỉ định file nên upload theo kiểu audio

```

<form id="form1">
  Select a file to upload:
  <input type="file" id="fname" size="50" accept="audio/*">
</form>

```

- **value**: chứa thông tin đường dẫn hoặc tên file được uploaded. Với các trình duyệt IE, Chrome, Opera, thuộc tính trả về tên file với đường dẫn là fakepath; với trình duyệt Firefox và Safari, thuộc tính trả về tên file vừa được uploaded.

Sinh viên tự tìm hiểu cách sử dụng cho các đối tượng HTML DOM khác.

Phần 3 – Các toán tử và cấu trúc lệnh

STT	Lệnh	Ví dụ
1	Chú thích	// chú thích dòng /*Chú thích Khối*/
2	Khai báo biến Biến toàn cục vs biến cục bộ Phân biệt hoa thường	X = 0; var varname="Hello";
3	Các toán tử số học	x=y+2; x=y-2; x=y*2;

		<pre> x=y/2; //division x=y%2; //modulus (division remainder) x=++y; //increase y, then assign y to x x=y++; //assign y to x, then increase y x--y; //decrease y, then assign y to x x=y--; //assign y to x, then decrease y </pre>
4	Các toán tử luận lý	<pre> var x=5; x==8; //comparision, equal to →false x==5; //→ true x==="5"; //equal value and type → false x===5; //→ true x!=8; //not equal → true x!="5"; //different value and type → true x!==5; //→ false x>8; //greater than → false x<8; //less than → true x>=8; //greater than or equal → false x<=8; //less than or equal → true </pre>
5	Toán tử <code>typeof(x)</code> Cho biết thông tin kiểu dữ liệu của toán hạng	<pre> var x=5; var shape= "round"; document.write(typeof(x)); //number document.write("
" + typeof(shape)); //string document.write("
" + typeof(true)); //boolean document.write("
" + typeof(null)); //object </pre>
6	Toán tử <code>this</code> Chỉ ra đối tượng hiện hành	<pre> function student(id, name){ //thuộc tính cho student this.id = id; this.name = name; } </pre>
7	If...Else	<pre> if (grade < 5){x = "Weak";} else if (grade < 8){x = "Good";} else{x = "Very well";} </pre>
8	Switch	<pre> var day=new Date().getDay(); switch (day){ case 2: x = "Monday"; break; case 7: x = "Saturday"; break; default: x = "Unknown"; } </pre>
9	For Loop	<pre> for (var i=0; i<5; i++){ x = x + "The number is " + i + "
"; } </pre>

10	While Loop	<pre>while (i<5){ x = x + "The number is " + i + "
"; i++; }</pre>
11	Do/While Loop	<pre>do{ x = x + "The number is " + i + "
"; i++; } while (i<5);</pre>
12	Break, thoát khỏi vòng lặp	<pre>for (i=0;i<10;i++){ if (i==3){ break; } x = x + "The number is " + i + "
"; }</pre>
13	Continue, tiếp tục bước lặp tiếp theo của vòng lặp	<pre>for (i=0;i<=10;i++){ if (i==3){ continue; } x = x + "The number is " + i + "
"; }</pre>
14	For/In Loop	<pre>var person={fname:"Van A", lname:"Nguyen", age:25}; for (x in person){ txt = txt + person[x]; }</pre>

Phần 4 – Xử lý lỗi

STT	Lệnh	Ví dụ
1	Try...Catch	<pre>var txt = ""; function message(){ try { adddler("Welcome guest!"); } catch(err) { txt="There was an error on this page.\n\n"; txt+="Click OK to continue viewing this page,\n"; txt+="or Cancel to return to the home page.\n\n"; if(!confirm(txt)) //if user click Cancel { document.location.href="http://www.w3schools.com/"; } }</pre>

		<pre> } } </pre>
2	Throw	<pre> var x=prompt("Enter a number between 5 and 10:",""); try{ if(x>10){throw "Err1";} else if(x<5){throw "Err2";} else if(isNaN(x)) //not a number {throw "Err3";} } catch(err){ if(err=="Err1") { document.write("Error! The value is too high."); } if(err=="Err2") { document.write("Error! The value is too low."); } if(err=="Err3") { document.write("Error! The value is not a number."); } } </pre>

Phần 5 – Xử lý chuỗi

STT	Tên phương thức	Mô tả
1	<string>.charAt(<index>)	Trả về ký tự tại chỉ số <index> trong chuỗi <string>
2	<string>.indexOf(<searchString>[,<index>])	Trả về vị trí xuất hiện đầu tiên của chuỗi <searchString> trong chuỗi <string>. Tham số <index> xác định vị trí bắt đầu tìm kiếm trong chuỗi <string>
3	<string>.substring(<index1>,<index2>)	Trả về chuỗi con trong chuỗi <string> từ vị trí <index1> đến vị trí <index2>
4	<string>.toLowerCase()	Đổi <string> thành chữ thường
5	<string>.toUpperCase()	Đổi <string> thành chữ hoa

Phần 6 – Xử lý công thức toán học

STT	Tên thuộc tính	Mô tả
1	E	Hằng số Euler
2	LN2	logarit tự nhiên của 2

3	LN10	logarit tự nhiên của 10
4	LOG2E	logarit cơ số 2 của e
5	PI	Hằng số pi

STT	Tên phương thức	Mô tả
1	Math.abs (number)	Tính giá trị tuyệt đối
2	Math.ceil (number)	Làm tròn lên đến giá trị nguyên cận trên gần nhất
3	Math.exp (number)	Tính giá trị mũ
4	Math.floor (number)	Làm tròn xuống đến giá trị nguyên cận dưới gần nhất
5	Math.log (number)	Tính logarit
6	Math.max (num1,num2)	Tìm giá trị lớn nhất
7	Math.min (num1,num2)	Tìm giá trị nhỏ nhất
8	Math.pos (base,exponent)	Tính giá trị mũ theo cơ số
9	Math.random (r)	Tính giá trị ngẫu nhiên
10	Math.round (number)	Làm tròn
11	Math.sqrt (number)	Lấy căn bậc 2

Phần 7 – Xử lý thông số thời gian

STT	Tên phương thức	Mô tả
1	getDate()	Lấy giá trị ngày
2	getDay()	Lấy giá trị ngày trong tuần (từ 0 đến 6). Chủ nhật là 0, thứ hai là 1.
3	getHours()	Lấy giá trị giờ
4	getMinutes()	Lấy giá trị phút
5	getSeconds()	Lấy giá trị giây
6	getYear()	Lấy giá trị năm
7	dateVar.toGMTString()	Chuyển đổi sang định dạng chuỗi theo múi giờ GMT

8

dateVar.toLocaleString()

Chuyển đổi sang định dạng chuỗi theo thiết lập cục bộ

Phần 8 – Lập trình hướng đối tượng

Sử dụng các đối tượng được xây dựng sẵn như: String, Array, Object, Date, Boolean, Math, RegExp...

Ví dụ:

```
//Tạo đối tượng student
student = new Object();

//thuộc tính cho student
student.name = "Nguyen Van A";
student.id = 5090xxxx;

//Khai báo các phương thức
student.evaluation = function() {
    this.status='Ready for graduation!';
}
student.renew=function(){
    this.status='Welcome to your school life!';
}

//Kiểm tra
alert(student.name);
student.renew();
alert(student.status);
student.evaluation();
alert(student.status);
```

Tạo lớp đối tượng do người dùng định nghĩa:

```
//Tạo lớp đối tượng student
function student(id, name){

    //thuộc tính cho student
    this.id = id;
    this.name = name;

    //Khai báo các phương thức
    this.evaluation = function(state) {
        if (state==1)
            this.status='Ready for graduation!';
        else
            this.status='Not yet!';
    }
    this.renew=function(state){
        if (state=='new')
            this.status='Welcome to your school life!';
        else
            this.status='Not yet!';
    }
}

//Kiểm tra
var stu = new student(5090xxxx, "Nguyen Van A");
```

```
alert(stu.name);
student.renew();
alert(stu.status);
student.evaluation();
alert(stu.status);
```

Làm việc với prototype (nhằm vào vấn đề bộ nhớ, tốc độ xử lý, làm việc với DOM). Lưu ý là mọi objects được gán với thuộc tính và phương thức prototype được khai báo trước đó.

```
//Tạo lớp đối tượng student
function student(id, name){

    //thuộc tính cho student
    this.id = id;
    this.name = name;
}

//Khai báo các phương thức
student.prototype.evaluation = function(state) {
    if (state==1)
        this.status='Ready for graduation!';
    else
        this.status='Not yet!';
}

student.prototype.renew=function(state){
    if (state=='new')
        this.status='Welcome to your school life!';
    else
        this.status='Not yet!';
}

//Kiểm tra
Var stu = new student(5090xxxx, "Nguyen Van A");
alert(stu.name);
student.renew();
alert(stu.status);
student.evaluation();
alert(stu.status);
```

Sử dụng prototype có thể dễ dàng mở rộng các đối tượng, ngay cả với các đối tượng được xây dựng sẵn:

```
Array.prototype.indexOf=function(obj){
    var result=-1;
    for(var i=0; i < obj.length; i++)
        if(this[i]==obj){
            return i;
            break;
        }
    }
    return result;
}

var ary=new Array();
ary=["one", "two", "three"];
alert(ary.indexOf("one"))
```

Phần 9 – jQuery

jQuery [3,4] là một framework JavaScript có khả năng hỗ trợ đa trình duyệt (cross-browser). Bên cạnh đó, jQuery đơn giản hóa lập trình với JavaScript, tiết kiệm thời gian và công sức lập trình. Sử dụng jQuery cung cấp cho người lập trình một số tiện ích sau:

- Truy cập các phần tử của tài liệu web.
- Thay đổi hình thức/giao diện trang web.
- Thay đổi nội dung trang web.
- Tương tác với người dùng với các hiệu ứng động.
- Tương tác ajax.

STT	Nội dung	Ví dụ
1	Chọn theo tên tag	<code>\$("#p")</code>
2	Chọn theo ID	<code>\$("#author")</code>
3	Chọn theo class	<code>\$(".content")</code>
4	Chọn các phần tử con ta dùng thêm ký hiệu >	<code>\$("#select-id > li")</code>
5	Chọn và loại trừ một số phần tử	<code>\$("#select-id > li:not(.current)")</code>
6	Chọn theo thuộc tính của Tag	<code>\$("#input[value=value']")</code> hoặc <code>\$("#a[href^='mailto:']") [11]</code>
7	Chọn kết hợp	<code>\$("#a[href^=http][href*='zend']") [11]</code>
8	Chọn phần tử theo index trong tập hợp	<code>\$("#select-id > li:eq(2)")</code>
9	Chọn thành phần con đầu tiên	<code>\$("#div:nth-child(1)")</code>
10	Chọn các phần tử li có index là số lẻ	<code>\$("#select-id > li:odd")</code>
11	Chọn theo nội dung bên trong	<code>\$(".content:contains('Example'))"</code>
12	Đặc biệt jquery hỗ trợ việc chọn các thành phần trong Form	<code>:text, :checkbox, :radio, :image, :submit, :reset, :password, :file, :input, :button, :enabled, :disabled, :checked, :selected</code>
13	Chọn các phần tử có liên quan	<code>.next()</code> : chọn phần tử cùng cấp và nằm kế sau. <code>.nextAll()</code> : chọn tất cả phần tử cùng cấp và nằm kế sau. <code>.prev()</code> : chọn phần tử cùng cấp và nằm kế trước. <code>.prevAll()</code> : chọn tất cả phần tử cùng cấp và nằm kế trước. <code>.parent()</code> : chọn phần tử cha.

		.children() : chọn các phần tử con. .find('selector') : tìm phần tử theo 'selector'.
--	--	---

STT	Tên sự kiện	Mô tả
1	.bind()	Gắn các sự kiện với hàm xử lý tương ứng
2	.click()	Bắt sự kiện click tương tự như sự kiện onClick() trong DOM.
3	.hover()	Xử lý 2 sự kiện đưa chuột vào và kéo chuột ra khỏi các phần tử HTML.
4	.load()	Bắt sự kiện load của element
5	.ready()	Chỉ định thực hiện khi DOM được load xong
6	.submit()	Bắt sự kiện submit, thường dùng cho form
7	.unbind()	Ngược lại với .bind()

STT	Hiệu ứng	Mô tả
1	.animate()	Thực hiện một tùy biến chuyển động của tập hợp các thuộc tính CSS
2	.delay()	Thiết lập thời gian trì hoãn thực hiện các function sau nó
3	.fadeIn()	Cho phép các phần tử trong tag hiện một cách từ từ biến thiên theo thời gian đã được thiết lập
4	.fadeOut()	Cho phép các phần tử trong tag ẩn một cách từ từ biến thiên theo thời gian đã được thiết lập
5	.fadeTo()	Điều chỉnh độ mờ của các phần tử trong HTML
6	jQuery.fx.interval	Thiết lập thời gian cho chuyển động (milliseconds)
7	.hide()	Ẩn các phần tử HTML theo thời gian
8	.show()	Hiện các phần tử HTML theo thời gian
9	.stop()	Kết thúc các hiệu ứng chuyển động hiện tại của element được chọn
10	jQuery.fx.off	Vô hiệu hóa tất cả các chuyển động

Phần 10 – AJAX (Asynchronous JavaScript and XML)

AJAX [6] không phải là một ngôn ngữ lập trình mà chỉ là một kỹ thuật giúp cho việc phát triển ứng dụng web tốt hơn. AJAX được sử dụng để trao đổi dữ liệu với server, cập nhật nội dung/dữ liệu một phần của trang web mà không cần phải tải lại toàn bộ trang web.

STT	Tên phương thức	Mô tả	Ví dụ
1	load(<url>, <data>, <callback>)	Load HTML	<pre>\$(document).ready(function(){ \$("button").click(function(){ \$("div").load("demo_ajax_load.txt"); }); });</pre>
2	ajax(<options>)	Xác định các thông số gọi ajax	<pre>\$.ajax({ type: "get" ,dataType: "xml" ,url: categoryurl ,success: function(xml){ alert("xml success!!"); } ,error: function(){ alert("xml error!!"); } });</pre>
3	post(<url>, <data>, <callback>, <type>)	Yêu cầu dữ liệu từ server với phương thức HTTP POST	<pre>\$("button").click(function(){ \$.post("demo_test_post.asp", { name:"Donald Duck", city:"Duckburg" }, function(data,status){ alert("Data: " + data + "\nStatus: " + status); }); });</pre>
4	get(<url>, <data>, <callback>, <type>)	Yêu cầu dữ liệu từ server với phương thức HTTP GET	<pre>\$("button").click(function(){ \$.get("demo_test.asp",function(data,status){ alert("Data: " + data + "\nStatus: " + status); }); });</pre>

Ví dụ, load nội dung của element có id=p1, trong file demo_test.txt vào một thẻ div có id=div1 của trang web:

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script>
[   $(document).ready(function() {
[       $("button").click(function() {
[           $("#div1").load("demo_test.txt #p1");
[       });
[   });
</script>
```

Phần 11 – JSON (JavaScript Object Notation)

JSON [2,13] là cấu trúc để lưu trữ và trao đổi thông tin dạng text, giống như XML, nhưng JSON được đánh giá là nhỏ hơn, nhanh hơn và dễ phân tích hơn XML.

Một số lý do để dùng JSON thay vì XML cho các ứng dụng AJAX:

- Sử dụng XML:
 - o Lấy file XML
 - o Dùng XML DOM để duyệt toàn bộ file
 - o Lấy và lưu giá trị trong XML vào biến
- Sử dụng JSON:
 - o Lấy 1 chuỗi JSON (kiểu string)
 - o Dùng hàm eval() để biến chuỗi JSON thành các đối tượng JavaScript

Một ví dụ của JSON: đối tượng employees được định nghĩa là một mảng có 3 records

```
{
  "employees": [
    { "firstName": "John" , "lastName": "Doe" },
    { "firstName": "Anna" , "lastName": "Smith" },
    { "firstName": "Peter" , "lastName": "Jones" }
  ]
}
```

JSON quy định cấu trúc theo những nguyên tắc sau:

- Dữ liệu là một cặp name – value

Ví dụ: JSON `"firstName" : "John"` , JavaScript `firstName = "John"`

- Các dữ liệu được cách nhau bằng dấu phẩy (,)
- Ngoặc nhọn dùng để phân tách giữa các đối tượng

Ví dụ: `{ "firstName": "John" , "lastName": "Doe" }`

- Ngoặc vuông để định nghĩa array

Vì JSON sử dụng cấu trúc của JavaScript (JSON là một phần trong cấu trúc JavaScript) nên ta không cần dùng phần mềm hỗ trợ khi làm việc với JSON. Ví dụ tạo một array JSON bằng JavaScript như sau:

```
var employees = [
  { "firstName":"John" , "lastName":"Doe" },
  { "firstName":"Anna" , "lastName":"Smith" },
  { "firstName":"Peter" , "lastName":"Jones" }
];
```

Để truy xuất một phần tử trong array trên:

```
employees[0].firstName + " " + employees[0].lastName;
```

JSON thường được dùng để lấy dữ liệu từ server, sau đó chuyển dữ liệu này sang dạng đối tượng JavaScript để xử lý. Như đã trình bày ở trên, hàm eval() của JavaScript được sử dụng để biến đổi chuỗi JSON thành đối tượng JavaScript. Ví dụ như sau:

Khai báo một chuỗi JSON:

```
var txt = '{ "employees" : [' +
  '{ "firstName":"John" , "lastName":"Doe" },' +
  '{ "firstName":"Anna" , "lastName":"Smith" },' +
  '{ "firstName":"Peter" , "lastName":"Jones" } ]}';
```

Biến đổi thành đối tượng JavaScript, lưu ý là chuỗi JSON này phải được đặt trong ngoặc tròn

```
var obj = eval ("(" + txt + ")");
```

Tuy nhiên, hàm eval() của JavaScript có thể thực thi tất cả những đoạn mã dạng script, do đó, điều này sẽ ảnh hưởng đến vấn đề bảo mật của trang web. Ngoài hàm eval(), người dùng có thể sử dụng hàm parse() của JSON cung cấp. Sinh viên tự tìm hiểu phần này.

Bài tập

1. Tạo một form nhập dữ liệu như hình bên dưới

Please fill in the following information		Here is your information after click Review
Last Name*	<input type="text"/>	First name* <input type="text"/>
Email*	<input type="text" value="example@abc.com"/>	
Address	<input type="text"/>	
Sex	<input type="radio"/> Man <input type="radio"/> Woman	
Birth date	<input type="text" value="mm/dd/yyyy"/>	
Job*	<input type="text" value="Student"/>	
You want to pay for*	<input type="text" value="1 semester"/>	
Picture(4x6)*	<input type="button" value="Choose File"/> No file chosen	
<input type="button" value="Review"/>		

Lưu ý:

- Giá trị mặc định của "You want to pay for" là "1 semester", "1 year". Giá trị hiển thị luôn là option đầu tiên.
- Nếu người dùng chọn Job là "Student" thì họ chỉ có thể chọn 2 option thời gian là "1 semester" và "1 year"

Student

1 semester

1 semester

1 year

- Nếu người dùng chọn Job là "Lecturer" hoặc "Other" thì họ chỉ có thể chọn 2 option thời gian là "1 year" và "2 years"

Lecturer

1 year

1 year

2 years

- Sinh viên có thể sử dụng hàm add() và remove() của đối tượng select, tuy nhiên cần lưu ý kiểm tra để tránh add hoặc remove giá trị nhiều lần
- Phần khung bên phải sẽ hiển thị thông tin người dùng đã nhập sau khi click Review. Sinh viên không cần dùng AJAX cho phần hiển thị này

Please fill in the following information		Here is your information after click Review
Last Name*	<input type="text" value="Nguyen"/>	Name: A Nguyen
First name*	<input type="text" value="A"/>	Email: ANguyen@mail.com
Email*	<input type="text" value="ANguyen@mail.com"/>	Address: On the Earth, under the Sun
Address	<input type="text" value="On the Earth, under the Sun"/>	Sex: Man
Sex	<input checked="" type="radio"/> Man <input type="radio"/> Woman	Birth date: 1980-09-26
Birth date	<input type="text" value="09/26/1980"/>	Job: Lecturer
Job*	<input type="text" value="Lecturer"/>	You want to pay for: 2 years
You want to pay for*	<input type="text" value="2 years"/>	Your picture: C:\fakepath\vc_m_s_kf_repr_832x624.jpg
Picture(4x6)*	<input type="button" value="Choose File"/> vc_m_s_kf_re...32x624.jpg	
<input type="button" value="Review"/>		

2. Làm lại bài 1, nhưng sử dụng AJAX và jQuery. Gợi ý:

- Sau khi click button Review, form sẽ gửi một thông điệp POST đến cho server kèm theo các thông tin người dùng đã nhập trên form.
- Server gửi trả file ASP có dạng như hình bên dưới.
- Sau khi nhận dữ liệu trả về từ server, form hiển thị dữ liệu này vào khung bên phải (như ở bài 1)
- Các file html và asp được copy vào cùng một thư mục trong inetpub/, máy tính có cài đặt IIS

```

<%
dim name,email
name=Request.Form("name")
email=Request.Form("email")
address=Request.Form("address")
sex=Request.Form("sex")
birthDate=Request.Form("birthDate")
jobSelect=Request.Form("jobSelect")
timeSelect=Request.Form("timeSelect")
picture=Request.Form("picture")
Response.Write("Name: " & name & "<br/>")
Response.Write("Email: " & email & "<br/>")
Response.Write("Address: " & address & "<br/>")
Response.Write("Sex: " & sex & "<br/>")
Response.Write("Birth Date: " & birthDate & "<br/>")
Response.Write("Job: " & jobSelect & "<br/>")
Response.Write("You want to pay for: " & timeSelect & "<br/>")
Response.Write("Your picture: " & picture & "<br/>")
%>

```

Please fill in the following information		Here is your information after click Review
Last Name*	<input type="text" value="Le"/>	Name: Tui Le
First name*	<input type="text" value="Tui"/>	Email: tui@mail
Email*	<input type="text" value="tui@mail"/>	Address: Unknown!
Address*	<input type="text" value="Unknown!"/>	Sex: Man
Sex	<input checked="" type="radio"/> Man <input type="radio"/> Woman	Birth Date: FF does not support it!
Birth date	<input type="text" value="FF does not support it!"/>	Job: Other
Job*	<input type="text" value="Other"/>	You want to pay for: 1 year
You want to pay for*	<input type="text" value="1 year"/>	Your picture: titleSelect.bmp
Picture(4x6)*	<input type="button" value="Browse..."/> titleSelect.bmp	
<input type="button" value="Review"/>		

3. Làm lại bài tập 7 ở Lab 2, CSS, nhưng dùng jQuery thay vì chỉ dùng HTML và CSS, tạo trang web như sau. Sinh viên có thể sử dụng file ảnh **cat.rar**

Catch me if you can!



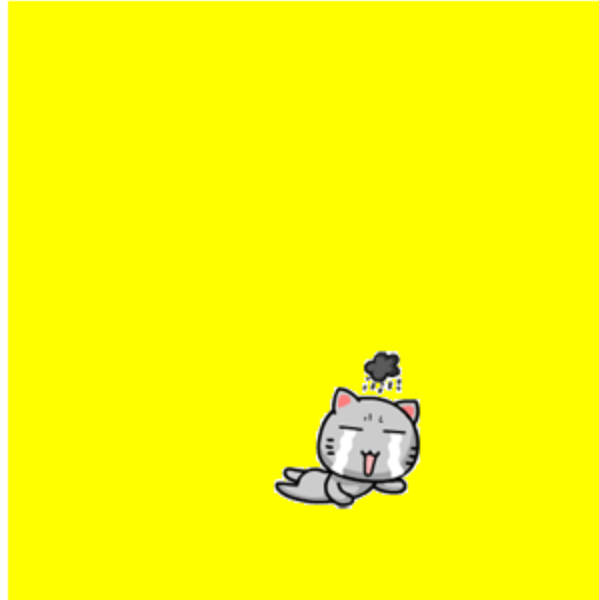
Khi trỏ chuột vào image

Catch me if you can!



Khi trỏ chuột phía trên hình và click chuột

Catch me if you can!



4. To be continued...

Tài liệu tham khảo

- [1] JavaScript W3School: <http://www.w3schools.com/js/default.asp> (2013)
- [2] JSON: <http://www.json.org/> (2013)
- [3] jQuery: <http://jquery.com/> (2013)
- [4] Lập trình jQuery: <http://lanhtv.webchuyennghiep.net/lap-trinh-jquery.html> (2013)
- [5] JavaScript Guide: <http://google-styleguide.googlecode.com/svn/trunk/javascriptguide.xml> (2013)
- [6] AJAX: <http://net.tutsplus.com/tutorials/javascript-ajax/5-ways-to-make-ajax-calls-with-jquery/> (2013)
- [7] jQuery and CSS: <http://stackoverflow.com/questions/15703249/how-to-make-jquery-or-css3-animated-catch-me-if-you-can-button> (2013)
- [8] jQuery change background: <http://stackoverflow.com/questions/7832140/jquery-change-background-image> (2013)
- [9] User agent string: <http://webaim.org/blog/user-agent-string-history/> (2013)
- [10] QuerySelector: <https://developer.mozilla.org/en-US/docs/Web/API/document.querySelector> (2013)
- [11] jQuery Selectors: <http://api.jquery.com/category/selectors/> (2013)
- [12] jQuery AJAX: http://www.w3schools.com/jquery/jquery_ajax_get_post.asp (2013)
- [13] JSON W3school: <http://www.w3schools.com/json/> (2013)
- [14] AJAX example: https://developer.mozilla.org/en-US/docs/AJAX/Getting_Started (2013)