



ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA CÔNG NGHỆ THÔNG TIN  
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

## Môn: Thiết kế & Lập trình Web 1

Bài 05:

# Ngôn ngữ JavaScript

Nguyễn Đức Huy  
ndhuy@fit.hcmus.edu.vn

# Nội dung

**1**  
Giới thiệu  
JavaScript

**2**  
Nhúng  
JavaScript  
vào HTML

**3**  
Kiểu dữ liệu  
và cú pháp  
JavaScript

**6**  
Ví dụ ứng  
dụng

**4**  
Xử lý sự kiện  
với JavaScript

**5**  
HTML DOM  
và JavaScript



# Giới thiệu JavaScript



Ngôn ngữ JavaScript



## ● Client-Side Script

- Script được thực thi tại *Client-Side* (trình duyệt): Thực hiện các tương tác với người dùng ( tạo menu chuyển động, ... ), kiểm tra dữ liệu nhập,...

## ● Server-Side Script

- Script được xử lý tại *Server-Side*, nhằm tạo các trang web có khả năng phát sinh nội dung động. Một số xử lý chính: kết nối CSDL, truy cập hệ thống file trên server, phát sinh nội dung html trả về người dùng...





- **JavaScript** Là ngôn ngữ *Client-side script* hoạt động trên trình duyệt của người dùng (*client*)
- **Chia sẻ xử lý** trong ứng dụng web. Giảm các xử lý không cần thiết trên server.
- Giúp **tạo các hiệu ứng, tương tác cho** trang web.





- Khi trình duyệt (*Client browser*) truy cập trang web có chứa các đoạn mã xử lý tại server-side. Server (run-time engine) sẽ thực hiện các lệnh *Server-side Scripts* và trả về nội dung HTML cho trình duyệt.
- Nội dung HTML trả về chủ yếu bao gồm: *mã html, client-script*.



## Nhúng JavaScript vào trang HTML



cuu duong than cong . com

Ngôn ngữ JavaScript

# Nhúng Javascript vào trang web



Định nghĩa Script trực tiếp trong trang HTML:

```
<script type="text/javascript">  
    <!--  
        // Lệnh Javascript  
    -->  
</script>
```

Nhúng sử dụng script cài đặt từ 1 file .js khác:

```
<script src="abc.js"></script>
```





- Web Browser sẽ thực thi **<script>** khi load trang web theo thứ tự từ **trên xuống** dưới.
- Đối với **Source code JavaScript** có thể đặt trong các file **.js** sẽ được **nhúng** vào file HTML trước khi hoạt động.
- Các đoạn **code JavaScript** được Browser xử **cùng thứ tự** với các thẻ **HTML**. Trừ các **hàm (function)** chỉ được thực thi khi **có lời gọi hàm**.

# Nhúng Javascript vào trang web



```
<html>
  <head>
    <script type="text/javascript">
      some statements
    </script>
  </head>
  <body>
    <script type="text/javascript">
      some statements
    </script>

    <script src="Tên_file_script.js">method()</script>

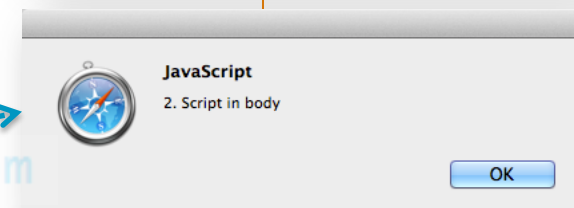
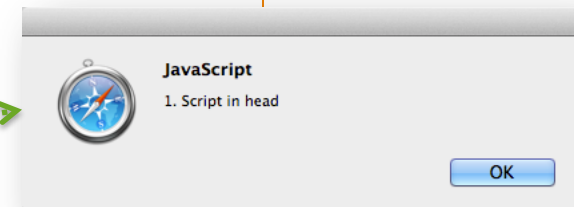
    <script type="text/javascript">
      // gọi thực hiện các phương thức được định nghĩa
      // trong "Tên_file_script.js"
    </script>
  </body>
</html>
```

# Nhúng Javascript vào trang web



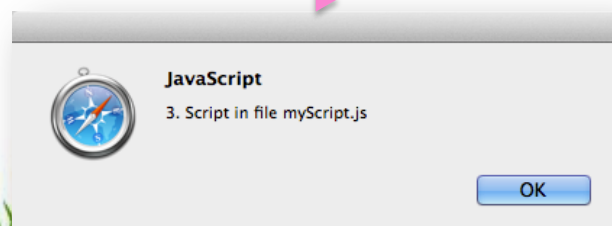
- Đặt giữa tag `<head>` và `</head>`: script sẽ thực thi ngay khi trang web được mở.
- Đặt giữa tag `<body>` và `</body>`: script trong phần body được thực thi khi trang web đang mở (sau khi thực thi các đoạn script có trong phần `<head>`).
- Số lượng đoạn client-script chèn vào trang không hạn chế.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
    <title>Title of website</title>
    <script type="text/javascript">
      alert("1. Script in head");
    </script>
  </head>
  <body>
    <script type="text/javascript">
      alert("2. Script in body");
    </script>
    <script type="text/javascript" src="myScript.js" ></script>
    <script type="text/javascript">
      showMessage();
    </script>
  </body>
</html>
```



File **myScript.js**

```
function showMessage()
{
  alert("3. Script in file myScript.js");
}
```







# Kiểu dữ liệu và cú pháp

## Ngôn ngữ JavaScript



Ngôn ngữ JavaScript

# Biến số trong Javascript



- Cách đặt tên biến

- Bắt đầu bằng một chữ cái hoặc dấu \_
- A..Z,a..z,0..9,\_ : phân biệt HOA, Thường

- Khai báo biến

- Sử dụng từ khóa **var**

Ví dụ: **var count = 10, amount;**

- *Không cần khai báo biến trước khi sử dụng*, biến thật sự tồn tại khi bắt đầu sử dụng lần đầu tiên.
- Biến *không cần khai báo kiểu dữ liệu* vì biến trong javascript *không có kiểu dữ liệu nhất định*

# Kiểu dữ liệu trong Javascript



Kiểu dữ liệu	Ví dụ	Mô tả
Object	<code>var listBooks = new Array(10) ;</code>	Trước khi sử dụng, phải cấp phát bằng từ khóa new
String	<code>"The cow jumped over the moon."</code> <code>"40"</code>	Chứa được chuỗi unicode Chuỗi rỗng ""
Number	0.066218 12	Theo chuẩn IEEE 754
boolean	true / false	
undefined	<code>var myVariable ;</code>	<code>myVariable = undefined</code>
null	<code>connection.Close();</code>	<code>connection = null</code>

***Một biến trong JavaScript có thể lưu bất kỳ kiểu nào***

- Biến tự đổi kiểu dữ liệu khi giá trị mà nó lưu trữ thay đổi

Ví dụ:

```
var x = 10;           // x kiểu Number
x = "hello world !"; // x kiểu String
```

- Có thể cộng 2 biến khác kiểu dữ liệu

Ví dụ:

```
var x;
x = "12" + 34.5; // KQ: x = "1234.5"
```

- Hàm **parseInt(...)**, **parseFloat(...)** : Đổi KDL từ chuỗi sang số.



# Hàm trong Javascript



- Dạng thức khai báo chung:

```
function Tên_hàm(thamso1, thamso2, ...)  
{  
    ...  
}
```

- Hàm có giá trị trả về:

```
function Tên_hàm(thamso1, thamso2, ...)  
{  
    ...  
    return (value);  
}
```

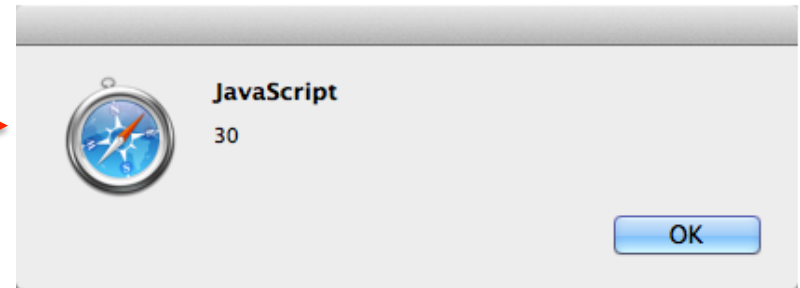


# Hàm trong Javascript



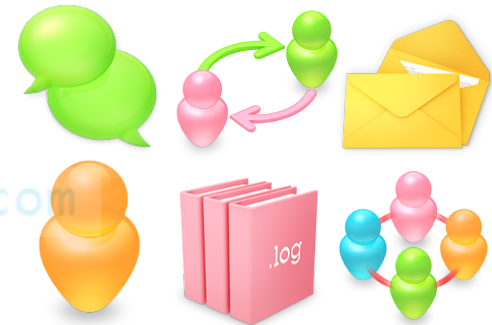
- Ví dụ:

```
function Sum(x, y)
{
    tong = x + y;
    return tong;
}
```



- Gọi hàm:

```
var x = Sum(10, 20);
alert(x);
```



# Các quy tắc chung



- Khối lệnh được bao trong dấu {}
- Mỗi lệnh nên kết thúc bằng dấu ;
- Cách ghi chú thích:
  - // *Chú thích 1 dòng*
  - /\* *Chú thích  
nhiều dòng* \*/



# Câu lệnh if



```
if (condition)  
{  
    statement[s] if true  
}  
else  
{  
    statement[s] if false  
}
```

Ví dụ:

```
var x = 5, y = 6, z;  
if (x == 5)  
{  
    if (y == 6)  
        z = 17;  
}  
else  
    z = 20;
```



# Câu lệnh switch



**switch** (*expression*)

{

**case** *label1* :

*statementlist*

**case** *label2* :

*statementlist*

...

**default** :

*statement list*

}

Ví dụ :

```
var diem = "G";
```

```
switch (diem) {
```

```
    case "Y":
```

```
        document.write("Yếu");
```

```
        break;
```

```
    case "TB":
```

```
        document.write("Trung bình");
```

```
        break;
```

```
    case "K":
```

```
        document.write("Khá");
```

```
        break;
```

```
    case "G":
```

```
        document.write("Giỏi");
```

```
        break;
```

```
    default:
```

```
        document.write("Xuất sắc");
```

```
}
```

# Vòng lặp for



```
for ([initial expression]; [condition]; [update expression])  
{  
    statement[s] inside loop  
}
```

Ví dụ:

```
var myarray = new Array();  
for (i = 0; i < 10; i++)  
{  
    myarray[i] = i;  
}
```

# Vòng lặp while & do ... while



```
while (expression)  
{  
    statements  
}
```

Ví dụ:

```
var i = 9, total = 0;  
while (i < 10)  
{  
    total += i * 3 + 5;  
    i = i + 5;  
}
```

```
do  
{  
    statements  
} while (expression);
```

Ví dụ:

```
var i = 9, total = 0;  
do  
{  
    total += i * 3 + 5;  
    i = i + 5;  
} while (i < 10);
```



# Xử lý sự kiện với JavaScript



Ngôn ngữ JavaScript

# Các sự kiện thông dụng



- Các sự kiện được hỗ trợ bởi hầu hết các đối tượng
  - **onClick**
  - **onFocus**
  - **onChange**
  - **onBlur**
  - **onMouseOver**
  - **onMouseOut**
  - **onMouseDown**
  - **onMouseUp**
  - **onLoad**
  - **onSubmit**
  - **onResize**
  - ...





# Xử lý sự kiện cho các thẻ HTML

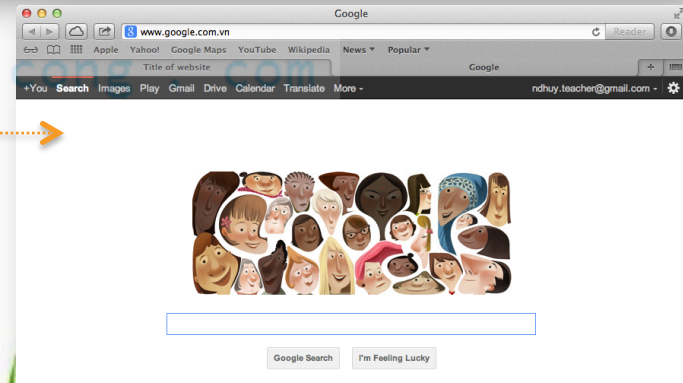
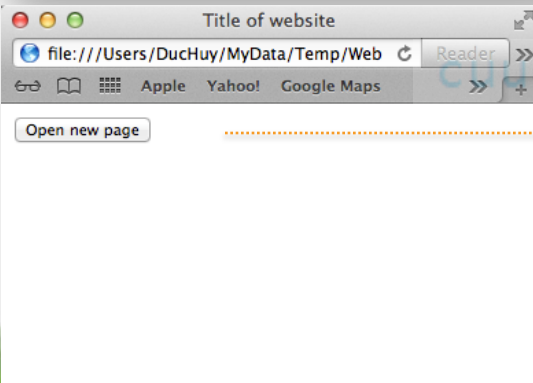


- Cú pháp:

**<TAG** **eventHandler** = **"JavaScript Code"**>

- Lưu ý: Dấu “...” và ‘...’

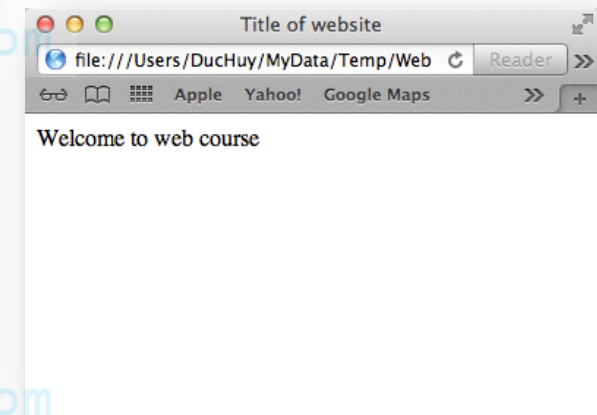
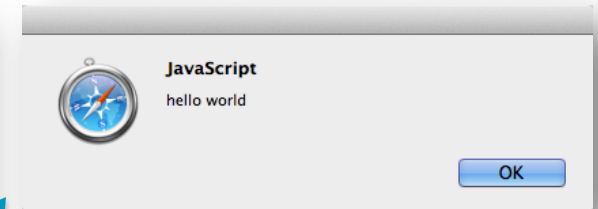
```
<body>
  <input type="button" value="duong than cong . com"
        name="btnClickMe"
        value="Open new page"
        onclick="window.open('http://www.google.com');" />
</body>
```



# Xử lý sự kiện bằng function



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0  
Transitional//EN" "http://www.w3.org/TR/xhtml1/  
DTD/xhtml1-transitional.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml">  
  <head>  
    <meta http-equiv="Content-Type"  
content="text/html; charset=UTF-8" />  
    <title>Title of website</title>  
    <script type="text/javascript">  
      function ShowMessage() {  
        alert("hello world");  
      }  
    </script>  
  </head>  
  <body onload="ShowMessage()">  
    Welcome to web course  
  </body>  
</html>
```



# Xử lý sự kiện bằng thuộc tính



- Gán tên hàm xử lý cho 1 object event  
**object.eventhandler = function\_name;**

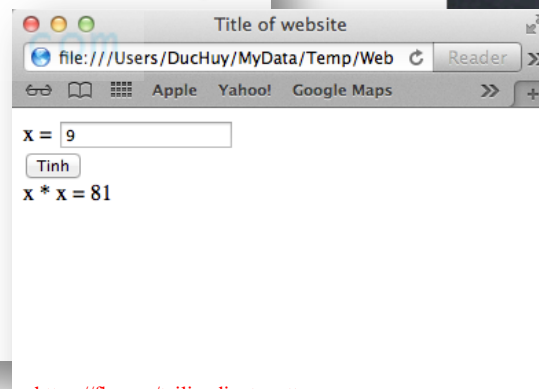
```
<html>
  <head>
    <script language="Javascript">
      function GreetingMessage()
      {
        window.alert("Welcome to my world");
      }

      window.onload = GreetingMessage();
    </script>
  </head>

  <body>
  </body>
</html>
```

# Ví dụ onclick Event

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//
EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
    <title>Title of website</title>
    <script type="text/javascript">
      function TinhToan() {
        var x = frmMain.txtGiaTri.value;
        ketQua.innerHTML = x * x;
      }
    </script>
  </head>
  <body>
    <form name="frmMain">
      x = <input type="text" name="txtGiaTri" />
      <br />
      <input type="button" value="Tinh"
        onclick="TinhToan()" />
      <br />
      x * x = <span id="ketQua"></span>
    </form>
  </body>
</html>
```

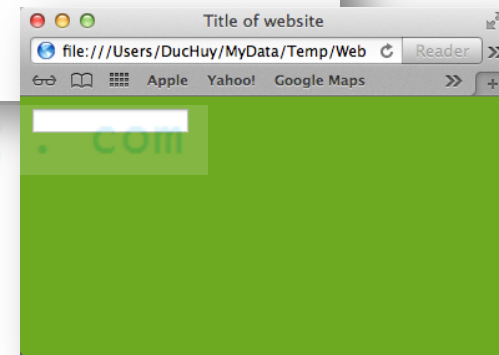
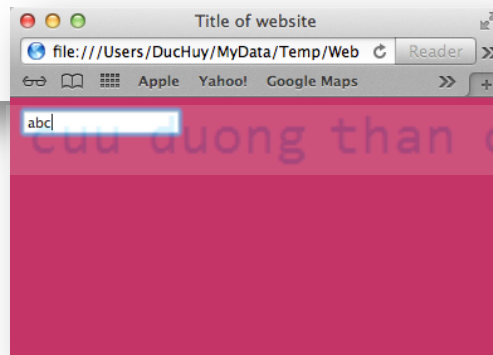


# Ví dụ onFocus - onBlur



- Xảy ra khi một thành phần HTML bị focus (**onFocus**) và mất focus (**onBlur**)

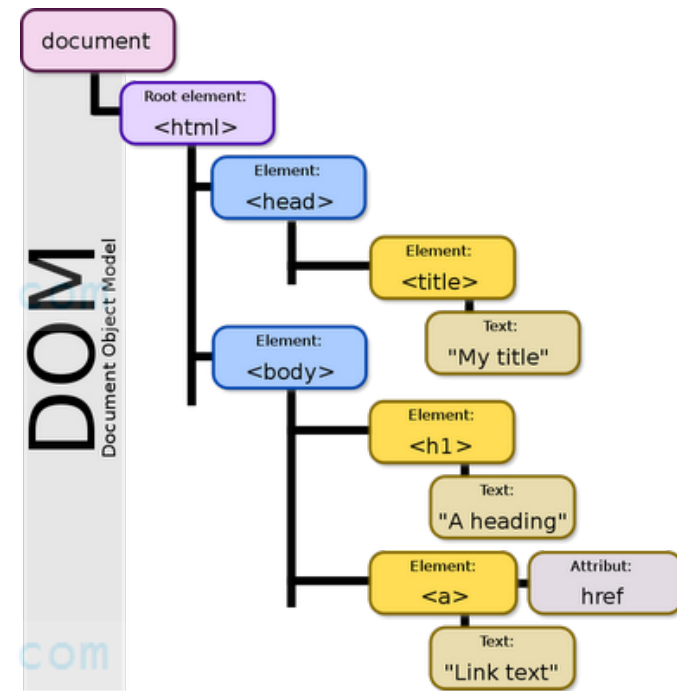
```
<body>
  <form name="frmMain">
    <input type="text"
      onFocus="document.bgColor='#cc3366';"
      onBlur="document.bgColor='#66aa33';"
    />
  </form>
</body>
```







# HTML DOM và JavaScript



Ngôn ngữ JavaScript

# Đối tượng HTML DOM



- **DOM** = Document Object Model
- Là tập hợp các đối tượng HTML chuẩn được dùng để **truy xuất** và **thay đổi thành phần HTML** trong trang web ( thay đổi nội dung tài liệu của trang )
- Một số đối tượng của **DOM**: *window, document, history, link, form, frame, location, event, ...*

# Đối tượng **Window** - DOM



- Là thể hiện của đối tượng **cửa sổ trình duyệt**
- Tồn tại khi mở 1 tài liệu HTML
- Sử dụng để truy cập thông tin của các đối tượng trên cửa sổ trình duyệt (tên trình duyệt, phiên bản trình duyệt, thanh tiêu đề, thanh trạng thái ... )

cuu duong than cong . com

# Đối tượng **Window** - DOM



## ● Properties

- **document**
- event
- history
- location
- name
- navigator
- screen
- status

## ● Methods

- Alert
- Confirm
- Prompt
- Blur
- close
- Focus
- open

# Đối tượng **Window** - DOM



## ● Ví dụ:

```
<html>
```

```
<body>
```

```
<script type="text/javascript">
```

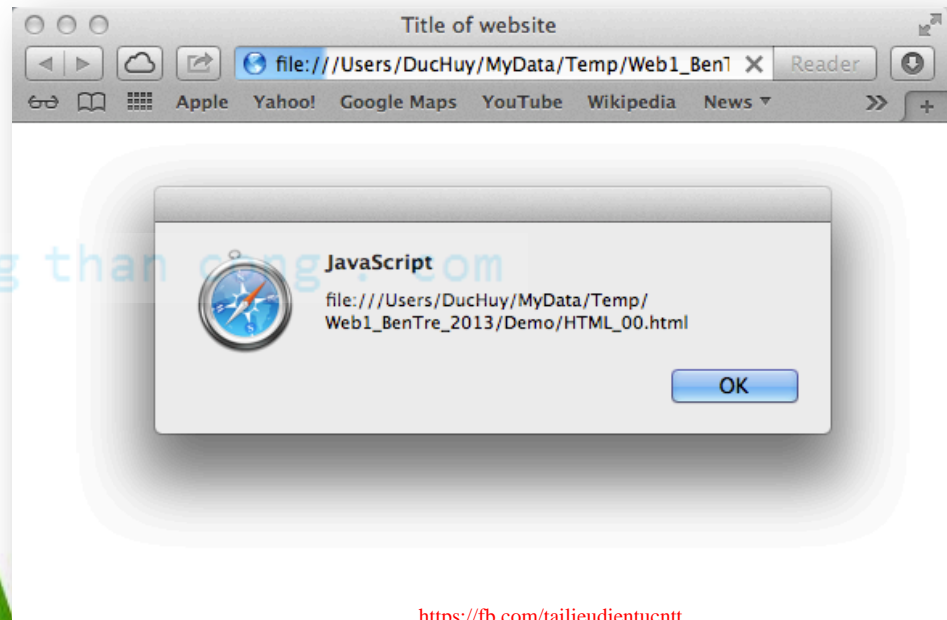
```
var curURL = window.location;
```

```
window.alert(curURL);
```

```
</script>
```

```
</body>
```

```
</html>
```





# Đối tượng **Document** - DOM



- Biểu diễn cho **nội dung trang HTML** đang được hiển thị trên trình duyệt
- Dùng để lấy thông tin về tài liệu, các thành phần HTML và xử lý sự kiện



# Đối tượng **Document** - DOM



## ● **Properties**

- aLinkColor
  - bgColor
  - **body**
  - fgColor
  - linkColor
  - title
  - **URL**
  - vlinkColor
  - forms[]
  - images[]
  - **childNodes[]**
- **documentElement**
  - cookie
  - .....

## ● **Methods**

- close
- open
- **createTextNode(" text ")**
- **createElement("HTMLtag")**
- **getElementById("id")**
- ...

# Đối tượng **Document** - DOM



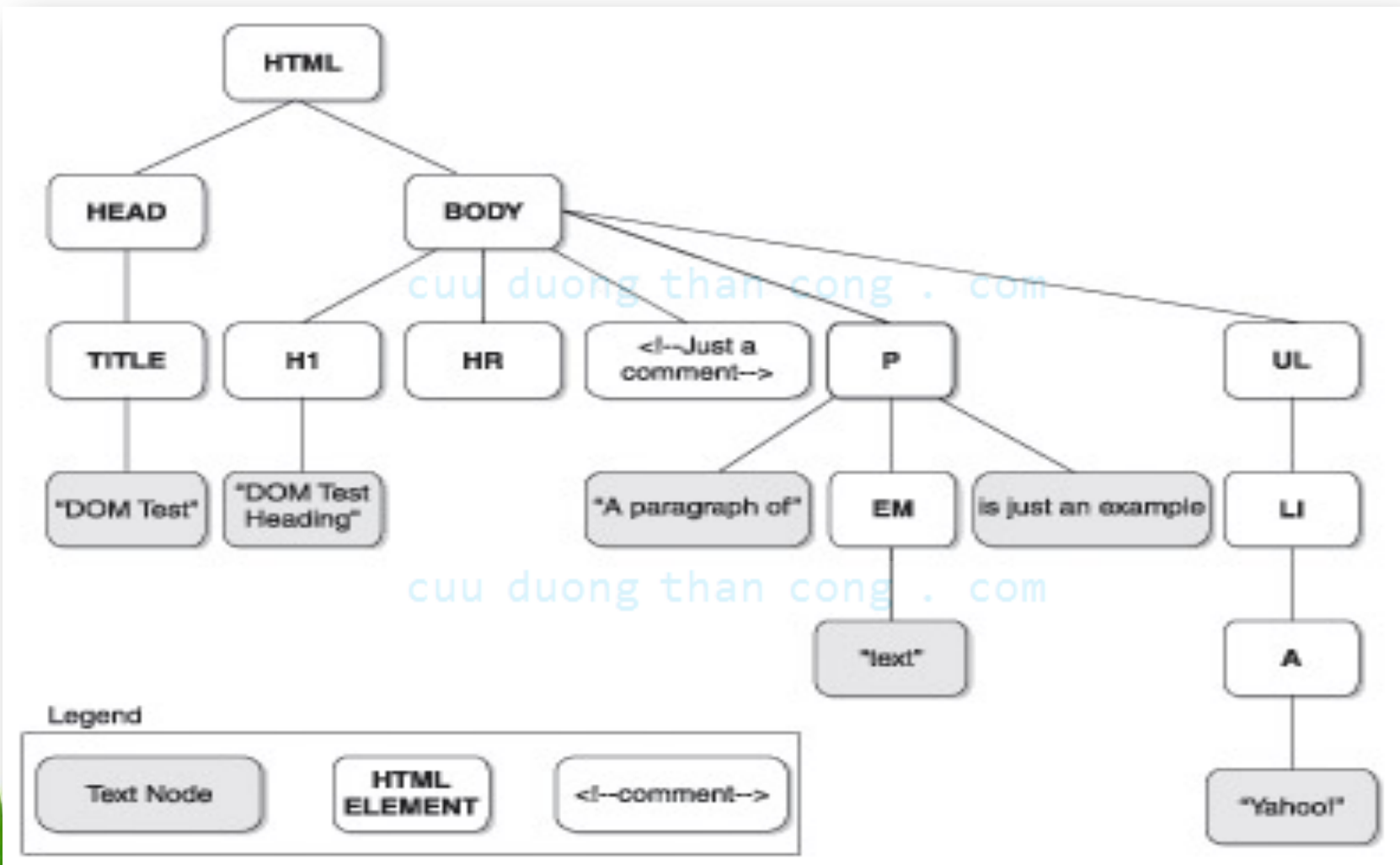
- Biểu diễn nội dung của tài liệu theo cấu trúc cây

```
<html>
  <head>
    <title>DOM Test</title>
  </head>
  <body>
    <h1>DOM Test Heading</h1>
    <hr />
    <!-- Just a comment -->
    <p id="p1" >A paragraph of <em>text</em> is just an example</p>
    <ul>
      <li>
        <a href="http://www.yahoo.com" >Yahoo! </a>
      </li>
    </ul>
  </body>
</html>
```

# Đối tượng **Document** - DOM



- Cấu trúc cây nội dung tài liệu



# Đối tượng **Document** - DOM



## ● Các loại **DOM Node** chính

Node Type Number	Loại	Mô tả	Ví dụ
1	Element	(X)HTML or XML element	<code>&lt;p&gt;...&lt;/p&gt;</code>
2	Attribute	Thuộc tính của HTML hay XML element	<code>align="center"</code>
3	Text	Nội dung chứa trong HTML or XML element	<code>This is a text fragment!</code>
8	Comment	HTML comment	<code>&lt;!-- This is a comment --&gt;</code>
9	Document	Đối tượng tài liệu gốc, thường là element nằm ở cấp cao nhất trong cây cấu trúc của tài liệu	<code>&lt;html&gt;</code>
10	DocumentType	Định nghĩa loại tài liệu	<code>&lt;!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd"&gt;</code>



# Đối tượng **Document** - DOM



- **getElementById** ( *id1* )

Trả về node có giá trị thuộc tính **id** = *id1*

Ví dụ:

```
//<p id="id1" >  
//      some text  
//</p>
```

Text Node

```
var node = document.getElementById("id1");  
var nodeName = node.nodeName; // p  
var nodeType = node.nodeType; // 1  
var nodeValue = node.nodeValue; // null  
var text = node.innerText; // some text
```

# Đối tượng **Document** - DOM



- **createElement** (*nodeName*)

Cho phép tạo ra 1 node HTML mới tùy theo đối số *nodeName* đầu vào

cuu duong than cong . com

Ví dụ:

```
var imgNode = document.createElement("img");  
imgNode.src = "images/test.gif";
```

```
// 
```

# Đối tượng **Document** - DOM



## ● **createTextNode** (*content*)

Ví dụ:

```
var textNode = document.createTextNode("New text");  
var pNode = document.createElement("p");  
pNode.appendChild(textNode);
```

```
// <p>New text</p>
```

cuu duong than cong . com

# Đối tượng **Document** - DOM



- **appendChild** (**newNode**)

Chèn node mới **newNode** vào cuối danh sách các node con của một node.

Ví dụ:

```
//<p id="id1">  
//      some text  
//</p>  
var pNode = document.getElementById("id1");  
var imgNode = document.createElement("img");  
imgNode.src = "images/test.gif";  
pNode.appendChild(imgNode);
```

```
//<p id="id1" > some text  /</p>
```

# Đối tượng **Document** - DOM



## ● **innerHTML**

Chỉ định nội dung HTML bên trong một node.

Ví dụ:

```
//<p id="para1" >cuu duong than cong . com  
//  some text  
//</p>
```

```
var theElement = document.getElementById("para1");  
theElement.innerHTML = "Some <b> new </b> text";
```

```
// Kết quả :      cuu duong than cong . com  
// <p id="para1" >  
// Some <b> new <b/> text  
// </p>
```

# Đối tượng **Document** - DOM



## ● **innerText**

Tương tự *innerHTML*, tuy nhiên bất kỳ nội dung nào đưa vào cũng được xem như là text hơn là các thẻ HTML.

Ví dụ:

```
var theElement = document.getElementById("para1");  
theElement.innerText = "Some <b> new </b> text";
```

```
// Kết quả hiển thị trên trình duyệt  
// bên trong thẻ p: "Some <b> new </b> text"
```



6

## Ví dụ Ứng dụng

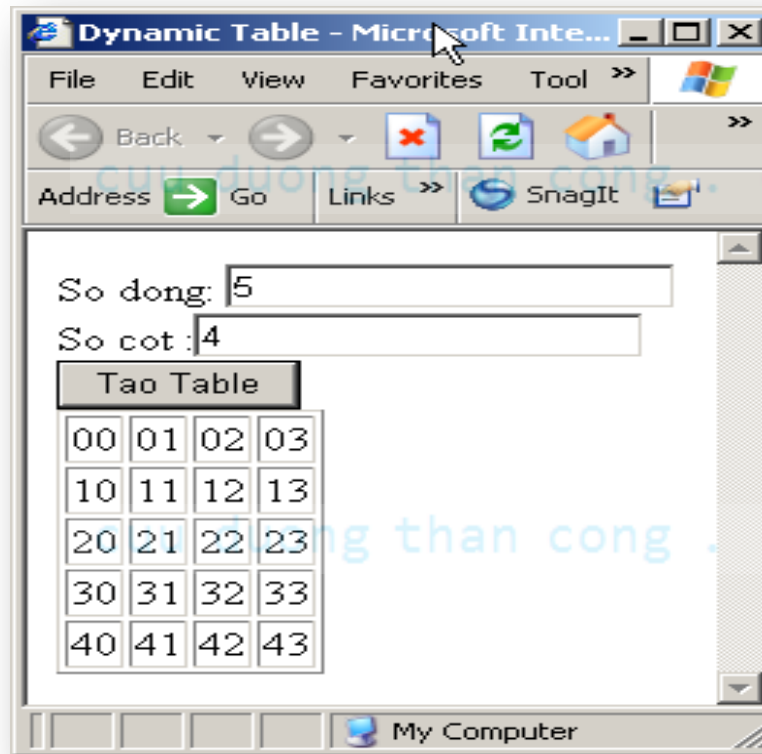


Ngôn ngữ JavaScript

# Ví dụ Dynamic Table



- Viết trang web cho phép tạo table có số dòng, số cột do người dùng nhập vào.



# Ví dụ Dynamic Table



`<Table>`

```
<Tr>  
  <td> 12 </td>  
  <td> 13 </td>  
</Tr>
```

```
<Tr>  
  <td> 21 </td>  
  <td> 22 </td>  
</Tr>
```

`</Table>`



# Ví dụ



body

Table

TBody

<Table>

<Tbody>

<Tr>

<td> 12 </td>

<td> 13 </td>

</Tr>

<Tr>

<td> 21 </td>

<td> 22 </td>

</Tr>

</Tbody>

</Table>

Tr

Td

Td

Tr

Td

Td

# Ví dụ Dynamic Table



- `Document.createElement(...)` :Tạo một đối tượng thẻ DOM HTML
- `Object.appendChild(...)`: Thêm một đối tượng thẻ DOM HTML như là nút con.

cuu duong than cong . com

# Ví dụ Dynamic Table



```
function CreateTable(divTable)
```

```
{
```

```
    var tagTable = document.createElement("table");  
    tagTable.border = 1;
```

```
    var tagTBody = document.createElement("tbody");  
    tagTable.appendChild(tagTBody);
```

```
    var nDong = txtSoDong.value;  
    var nCot = txtSoCot.value;
```

```
    for (i=0; i<nDong; i++)  
    {
```

```
        var tagTR = document.createElement("tr");  
        for (j=0; j<nCot; j++)  
        {
```

```
            var tagTD = document.createElement("td");  
            var textNode = document.createTextNode(i+" "+j);  
            tagTD.appendChild(textNode);
```

```
            tagTR.appendChild(tagTD);
```

```
        }
```

```
        tagTBody.appendChild(tagTR);
```

```
    }  
    divTable.appendChild(tagTable);
```

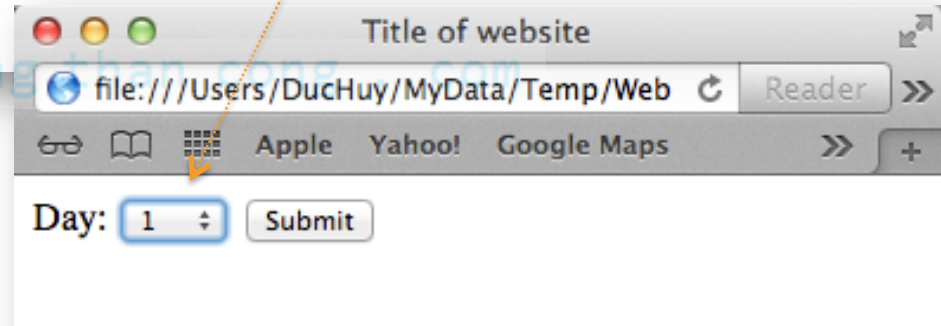
```
}
```



# Ví dụ Tạo ComboBox bằng JavaScript



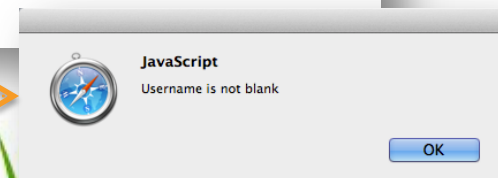
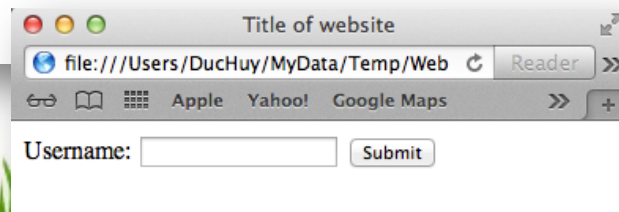
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Title of website</title>
  </head>
  <body>
    <form name="frmMain" action="#" method="get">
      Day:
      <select name="cmbDay">
        <script type="text/javascript">
          for(var i = 1; i <= 31; i++){
            document.writeln("<option value='" + i + "'>" + i + "</option>");
          }
        </script>
      </select>
      <input type="submit" value="Submit" />
    </form>
  </body>
</html>
```

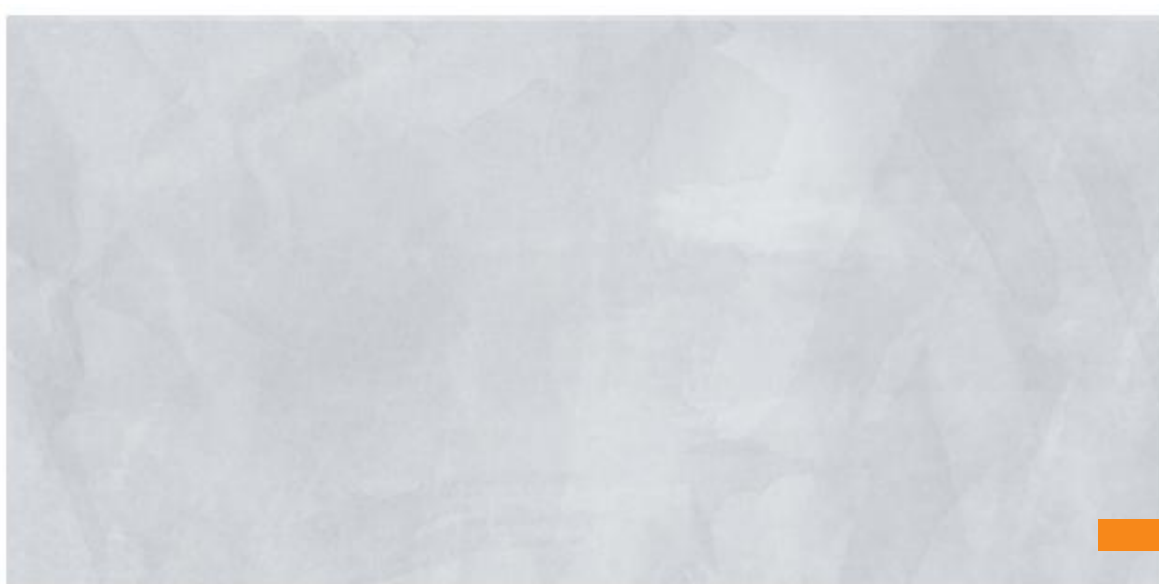


# Ví dụ Kiểm tra tính nhập liệu trước khi Submit



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Title of website</title>
    <script type="text/javascript">
      function CheckInput(){
        var us = document.getElementById("txtUS");
        if(us.value == ""){
          alert("Username is not blank");
          us.focus(); //Đưa con trỏ điều khiển đến TextBox Username
          return false;
        }
        return true;
      }
    </script>
  </head>
  <body>
    <form name="frmMain" action="#" method="get" onsubmit="return CheckInput()">
      Username: <input type="text" name="txtUS" id="txtUS" />
      <input type="submit" value="Submit" />
    </form>
  </body>
</html>
```





cuu duong than cong . com

# Ngôn ngữ JavaScript

cuu duong than cong . com





# Câu hỏi ?