

# Phát triển ứng dụng web



Khoa Công nghệ thông tin  
Bộ môn Công nghệ phần mềm

# Javascript

Nguyễn Huy Khánh  
nhkhanh@fit.hcmus.edu.vn

# Mục tiêu



- Hiểu được ý nghĩa của Javascript trong trang web.
- Biết nguyên tắc và cú pháp khi lập trình Javascript.
- Biết cách sử dụng Javascript để lập trình và xử lý sự kiện trong trang web.

# Nội dung



Giới thiệu về Javascript

Nhúng Javascript vào trang web

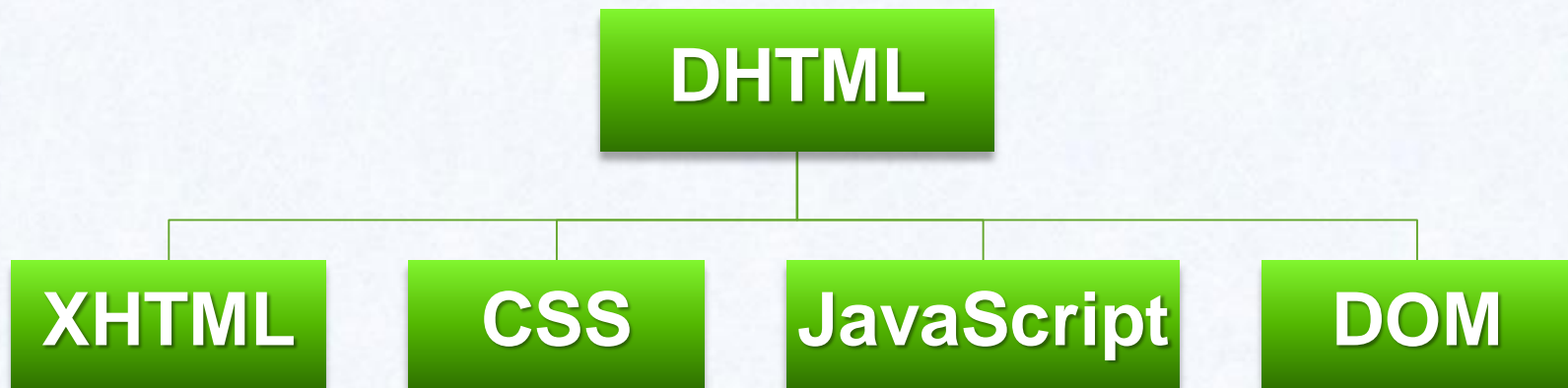
Kiểu dữ liệu & cú pháp trong Javascript

Xử lý sự kiện

DOM HTML với Javascript

# Dynamic HTML (DHTML)

- Cho phép trang web có thể tương tác và thay đổi tùy theo hành động của người dùng.
- $\text{DHTML} = \text{HTML} + \text{CSS} + \text{JavaScript}$





# Ý nghĩa Javascript

- HTML: xác định nội dung trang web thông qua các thẻ ngữ nghĩa (heading, paragraph, list...).
- CSS: xác định luật hay định dạng để thể hiện tài liệu HTML
  - Font chữ
  - Nền (màu, hình ảnh, ...)
  - Vị trí và kích thước
- Javascript: xác định các hành động
  - Tương tác thông qua các hành động của người dùng, xử lý sự kiện, ...

# Giới thiệu về 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.

# Giới thiệu về 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...

# Giới thiệu về Javascript

- 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.



# Nội dung



Giới thiệu về Javascript

Nhúng Javascript vào trang web

Kiểu dữ liệu & cú pháp trong Javascript

Xử lý sự kiện

DOM HTML với 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="xxx.js"></script>
```

# 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.

# VD: Nhúng Javascript vào trang web

```
<html>
```

```
<body>
```

```
document.write("Hello world!");
```

```
<script type="text/javascript">  
    document.write("Hello world!");  
</script>
```

```
</body>
```

```
</html>
```





# Nội dung



Giới thiệu về Javascript

Nhúng Javascript vào trang web

Kiểu dữ liệu & cú pháp trong Javascript

Xử lý sự kiện

DOM HTML với 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** để giới hạn phạm vi biến trong function

Ví dụ: **var count=10, amount;**

- Sử dụng từ khóa **let** để giới hạn phạm vi biến trong code block
- Không cần khai báo biến trước khi sử dụng, biến thật sự tồn tại khi bắt đầu sử dụng lần đầu tiên và có phạm vi toàn cục.

# Phạm vi của biến

- Phạm vi của biến được giới hạn trong `function`.
- Tất cả các biến được khai báo bên ngoài `function` có phạm vi toàn cục.
- Tất cả các biến toàn cục là con của đối tượng `window`.
- Nếu một biến được gán khi chưa được khai báo thì biến đó sẽ có phạm vi toàn cục.

```
function a(){  
    if (1==1){  
        var x = 10;  
    }  
    console.log(x);  
}
```

```
function a() {  
    x = 10;  
}  
a();  
console.log(x);
```

# 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 <b>new</b>
String	<code>"The cow jumped over the moon."</code> <code>"40"</code>	Chứa được chuỗi unicode Chuỗi rỗng <code>"</code>
Number	<code>0.066218</code> <code>12</code>	Theo chuẩn IEEE 754
boolean	<code>true / false</code>	
undefined	<code>var myVariable ;</code>	<code>myVariable = undefined</code>
null	<code>connection.Close();</code>	<code>connection = null</code>

1 Biến trong javascript có thể lưu bất kỳ kiểu dữ liệu nào.



# Đổi kiểu dữ liệu

- 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.

JavaScript Coercion

# 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);
```

# 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** *label* :

*statementlist*

**case** *label* :

*statementlist*

...

**default** :

*statement list*

}

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

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

Ví dụ:

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



# Vòng lặp do.. while

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

Ví dụ:

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

# Nội dung



Giới thiệu về Javascript

Nhúng Javascript vào trang web

Kiểu dữ liệu & cú pháp trong Javascript

Xử lý sự kiện

DOM HTML với 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 1:

`<tag eventHandler = "JavaScript Code">`

- Ví dụ:

`<body>`

`<input type="button" name="Button1"`

`value="OpenSesame!"`

`onclick="window.open('mydoc.html');">`

`</body>`



# Xử lý sự kiện bằng function

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

<body onload="GreetingMessage () ">
</body>
```

# Xử lý sự kiện bằng thuộc tính

- Gán tên hàm xử lý cho 1 object event

`object.eventhandler = function_name;`

- Ví dụ:

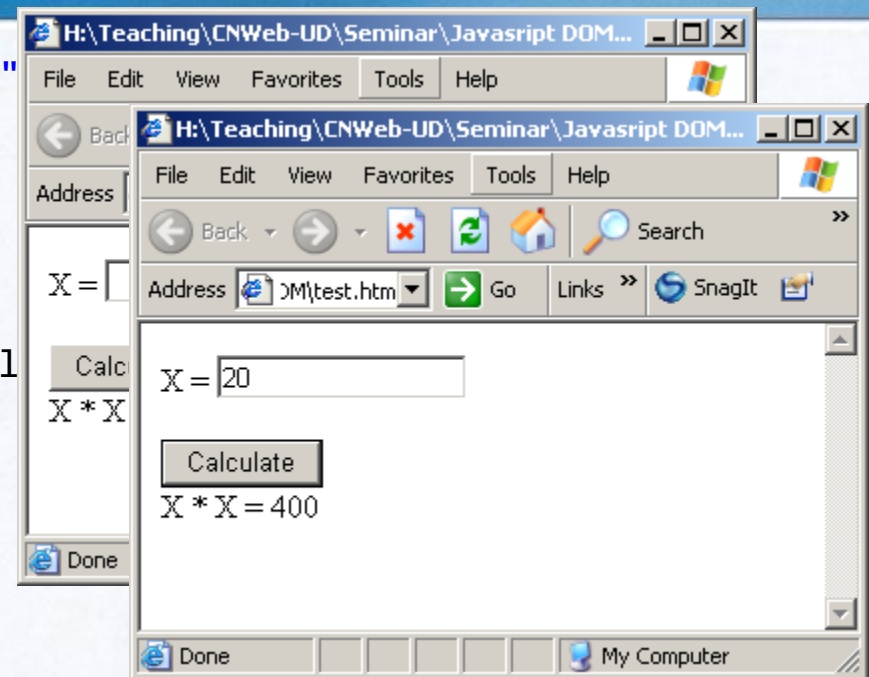
```
<head>
  <script language="Javascript">
    function GreetingMessage ()
    {
      window.alert("Welcome to my world");
    }

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

<body>
</body>
```

# Ví dụ: onclick Event

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Untitled Page</title>
  <script type="text/javascript">
    function compute(frm) {
      var x = frm.expr.value;
      document.getElementById("result")
    }
  </script>
</head>
<body>
  <form name="frm">
    X =
    <input type="text" name="expr" size="15">
    <br />
    <br />
    <input type="button" value="Calculate" onclick="compute(this.form)">
    <br />
    X * X = <span id="result"></span>
  </form>
</body>
</html>
```



# Nội dung



Giới thiệu về Javascript

Nhúng Javascript vào trang web

Kiểu dữ liệu & cú pháp trong Javascript

Xử lý sự kiện

DOM HTML với 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... )

# 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">
```

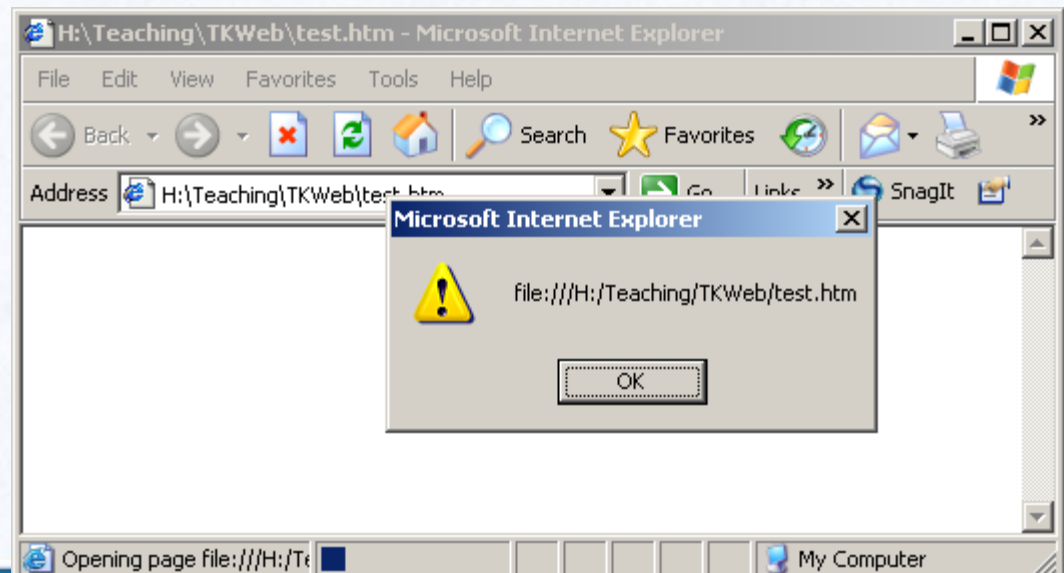
```
      var curURL = window.location;
```

```
      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")
- ...

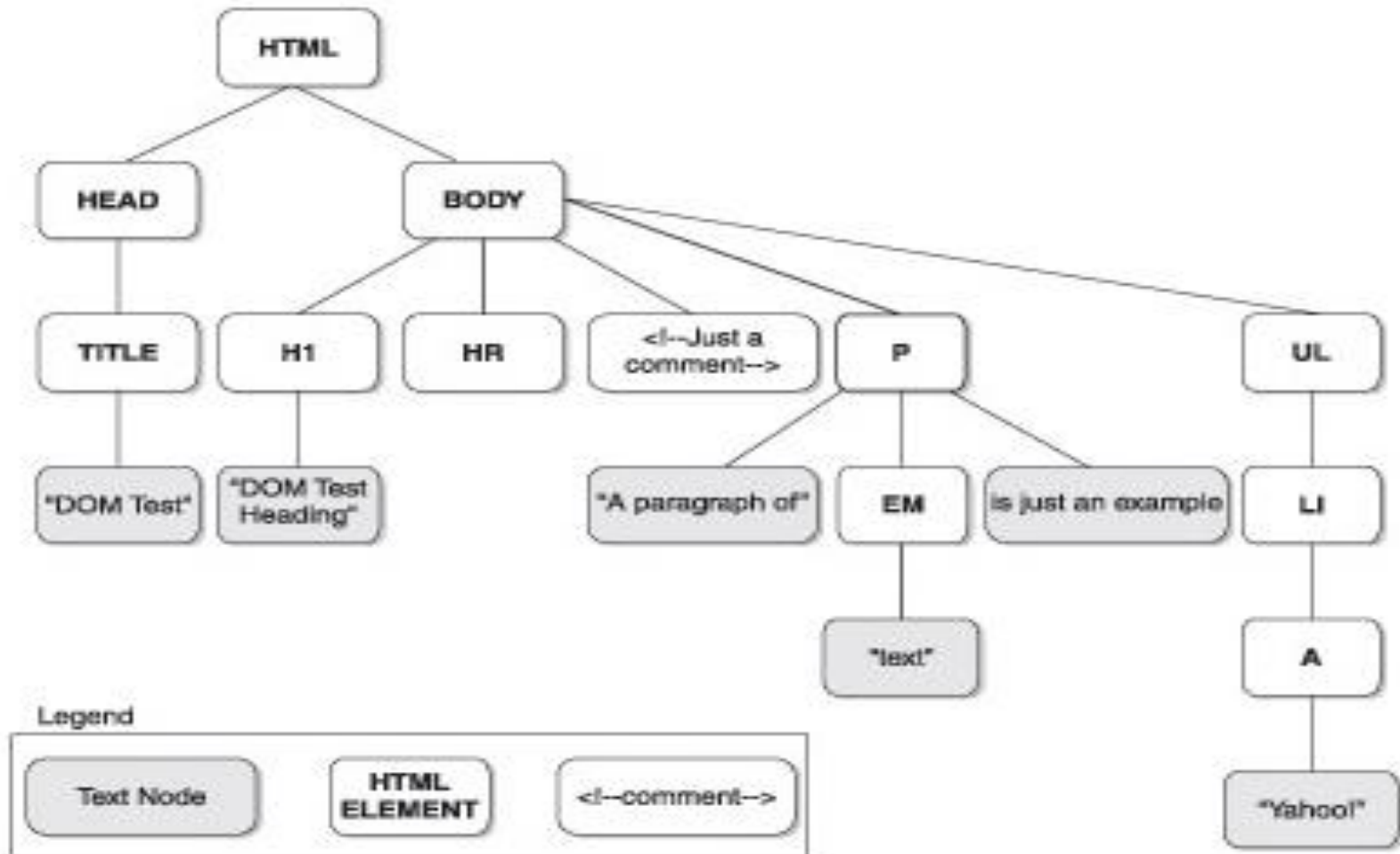
# Đối tượng Document - DOM

- 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





# 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

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

```
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

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

# Đối tượng Document - DOM

- createTextNode ( content )

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

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



# Đối tượng Document - DOM

- appendChild ( newNode )

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

```
//<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.

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

```
// Kết quả :  
// <p id="para1" >  
// Some <b> new <b/> text  
// </p>
```

# Đối tượng Document - DOM

- innerText

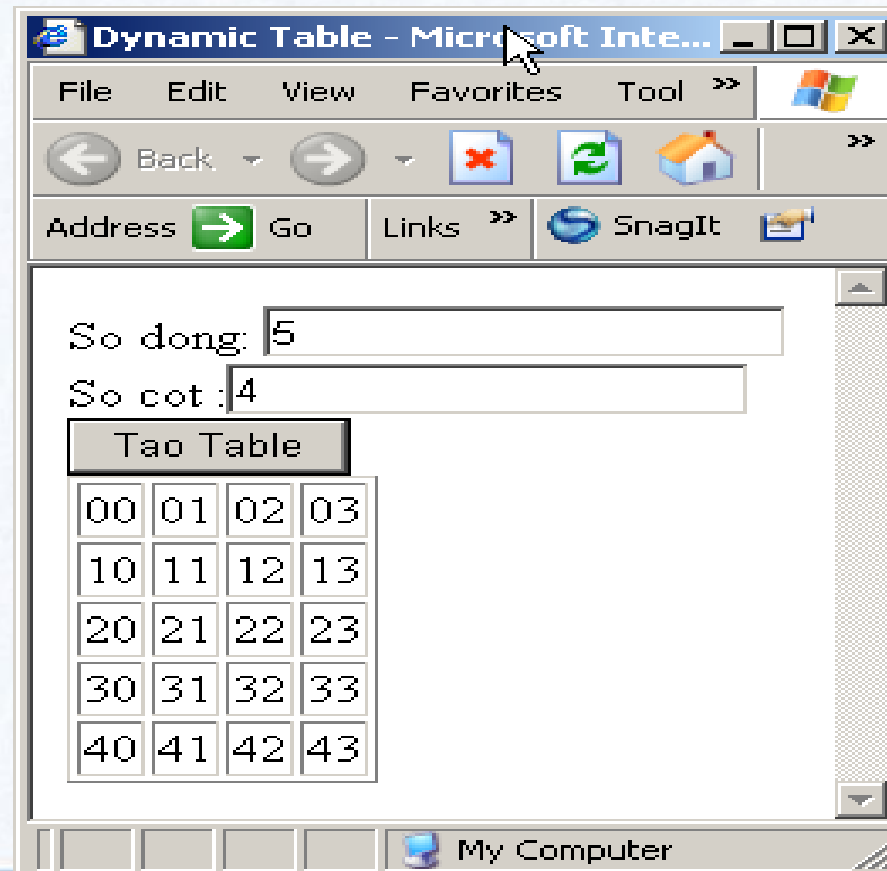
Tương tự innerHTML, tuy nhiên bất kỳ nội dung nào đưa vào cũng được xem như là text hơn là các thẻ HTML.

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"
```

# Ví dụ: Dynamic table

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





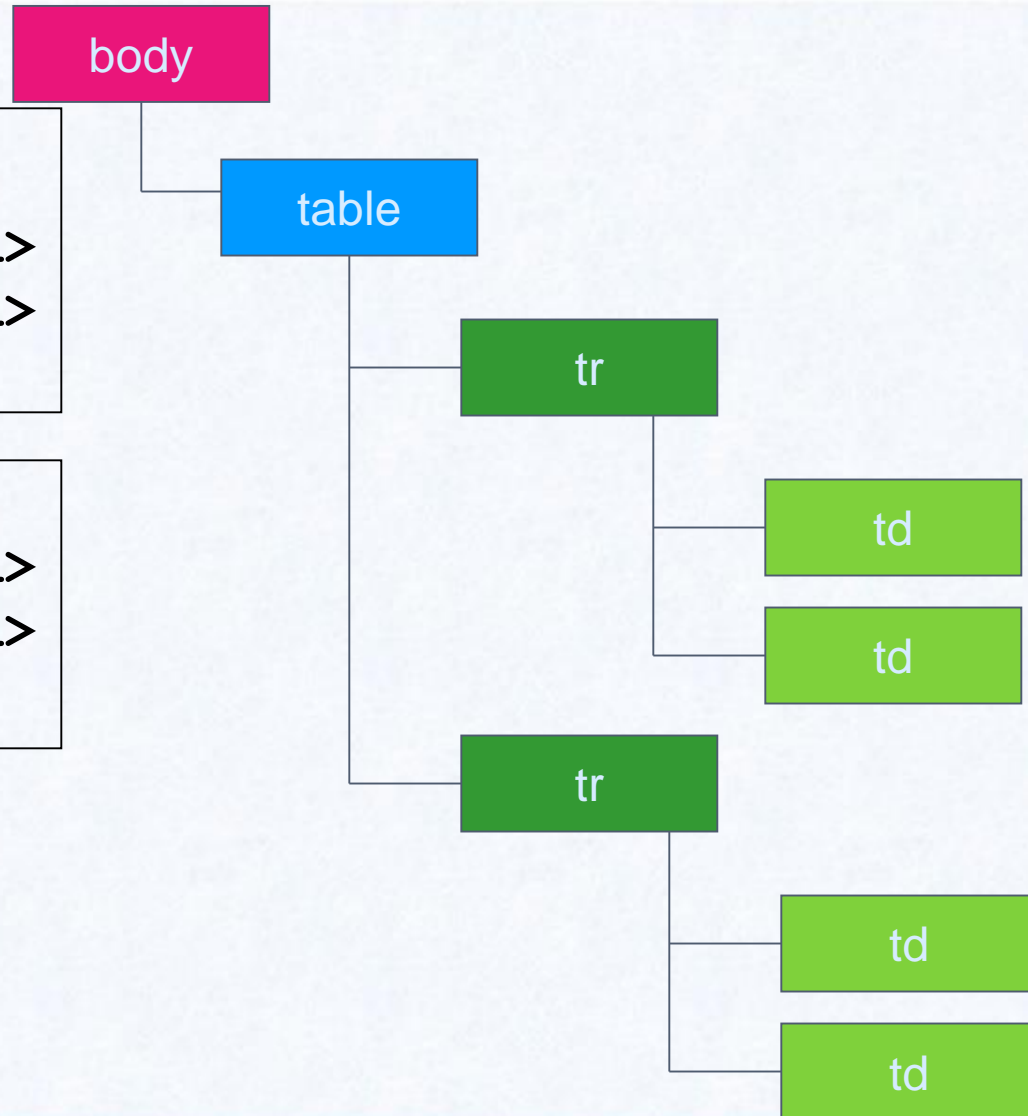
# Ví dụ: Dynamic table

`<table>`

```
<tr>  
  <td> 12 </td>  
  <td> 13 </td>  
</tr>
```

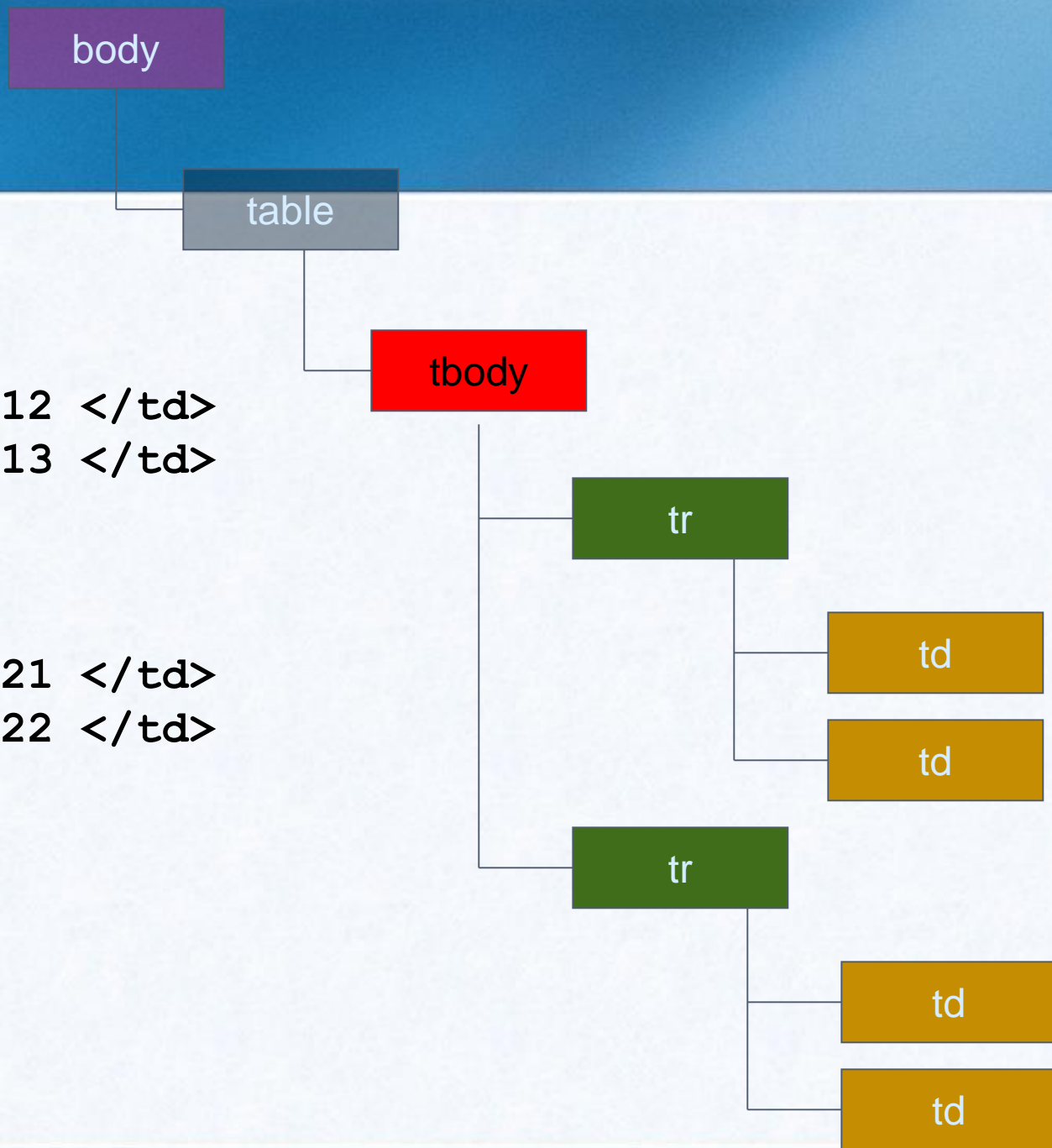
```
<tr>  
  <td> 21 </td>  
  <td> 22 </td>  
</tr>
```

`</table>`



# Ví dụ:

```
<table>  
  <tbody>  
    <tr>  
      <td> 12 </td>  
      <td> 13 </td>  
    </tr>  
    <tr>  
      <td> 21 </td>  
      <td> 22 </td>  
    </tr>  
  </tbody>  
</table>
```



# 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.

# 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);  
}
```



# Bài tập cá nhân



- Dùng HTML, CSS và Javascript để làm trang web như sau:

## Bé tập tính

Số thứ nhất	<input type="text"/>	<input type="radio"/> Cộng	<input type="radio"/> Trừ
Số thứ hai	<input type="text"/>	<input type="radio"/> Nhân	<input checked="" type="radio"/> Chia
Kết quả	<input type="text"/>	<input type="button" value="Tính"/>	
Thông báo	Giá trị nhập ở ô <i>Số thứ nhất</i> không phải là số		

# Bài tập cá nhân



- Các đối tượng được canh thẳng theo cột, chỉ có thể chọn một trong bốn phép tính (2 điểm)
- Viết mã Javascript để khi bấm vào nút tính, kết quả của phép tính đang chọn sẽ được điền vào ô kết quả. (4 điểm)
- Vùng thông báo sẽ hiển thị thông báo lỗi khi
  - Di chuyển khỏi ô nhập *Số thứ nhất* hoặc *Số thứ hai* mà dữ liệu không phải là số thực.
  - Bấm nút *Tính* mà
    - chưa chọn phép tính.
    - chưa điền đủ hai số hợp lệ để thực hiện phép tính (4 điểm)

# Tham khảo thêm



- [http://www.w3schools.com/js/js\\_object\\_prototypes.asp](http://www.w3schools.com/js/js_object_prototypes.asp)
- <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Closures>
- [http://www.w3schools.com/js/js\\_function\\_closures.asp](http://www.w3schools.com/js/js_function_closures.asp)
- [http://localjs.org/docs/en/tutorial\\_js4cpp.php](http://localjs.org/docs/en/tutorial_js4cpp.php)
- [http://en.wikipedia.org/wiki/Comparison\\_of\\_JavaScript\\_frameworks](http://en.wikipedia.org/wiki/Comparison_of_JavaScript_frameworks)
- <http://www.netmagazine.com/features/essential-javascript-top-five-mvc-frameworks>