Bài 6 – Phần 2/2

HTML DOM với Javascript

Khoa CNTT – ĐH.KHTN

Nội dung

- Giới thiệu về HTML DOM
- Thuộc tính (Property) và Phương thức (Method)
- Xử lý sự kiện (Event)

Nội dung

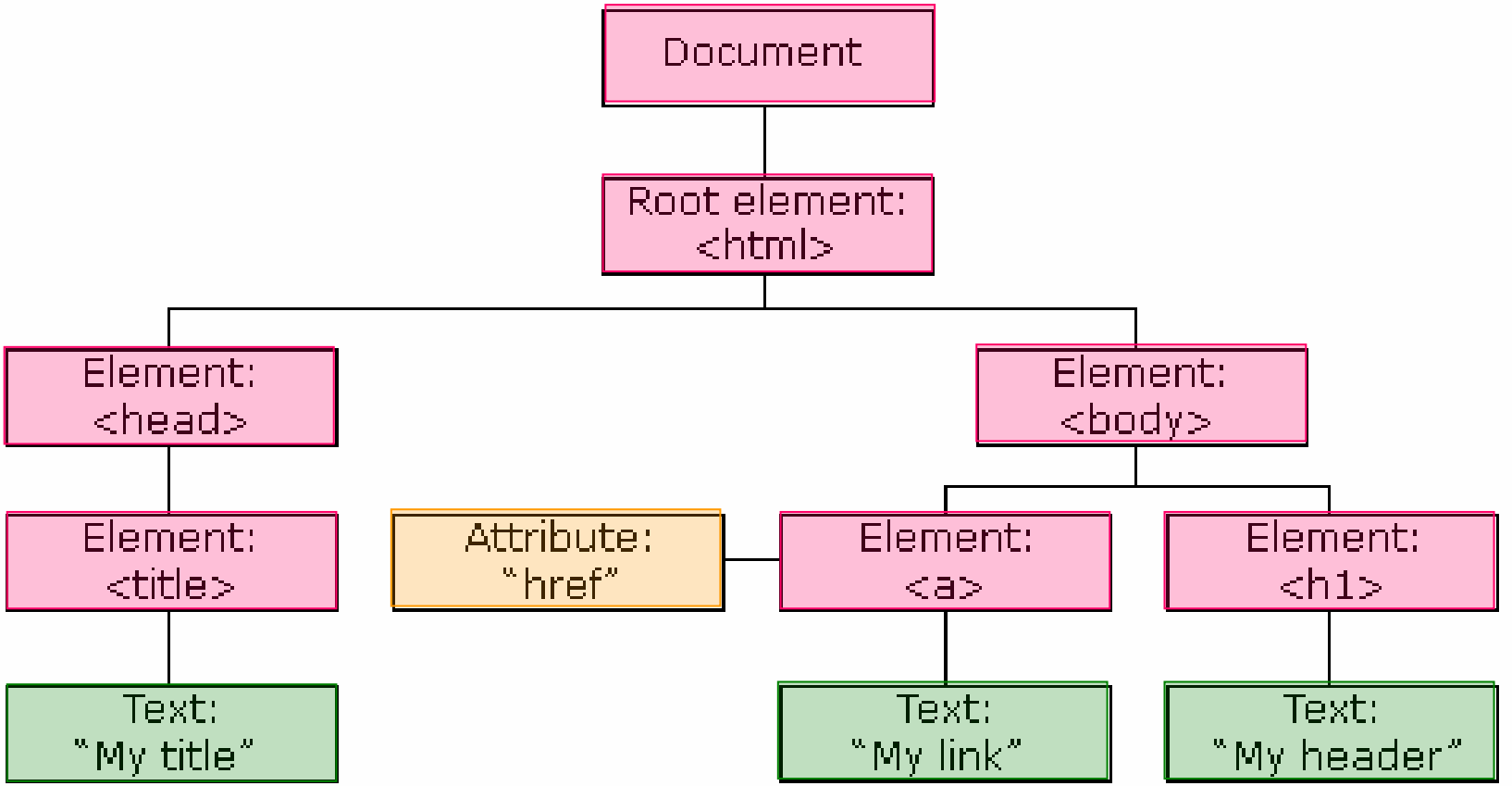
- Giới thiệu về HTML DOM
- Thuộc tính (Property) và Phương thức (Method)
- Xử lý sự kiện (Event)

Giới thiệu về HTML DOM

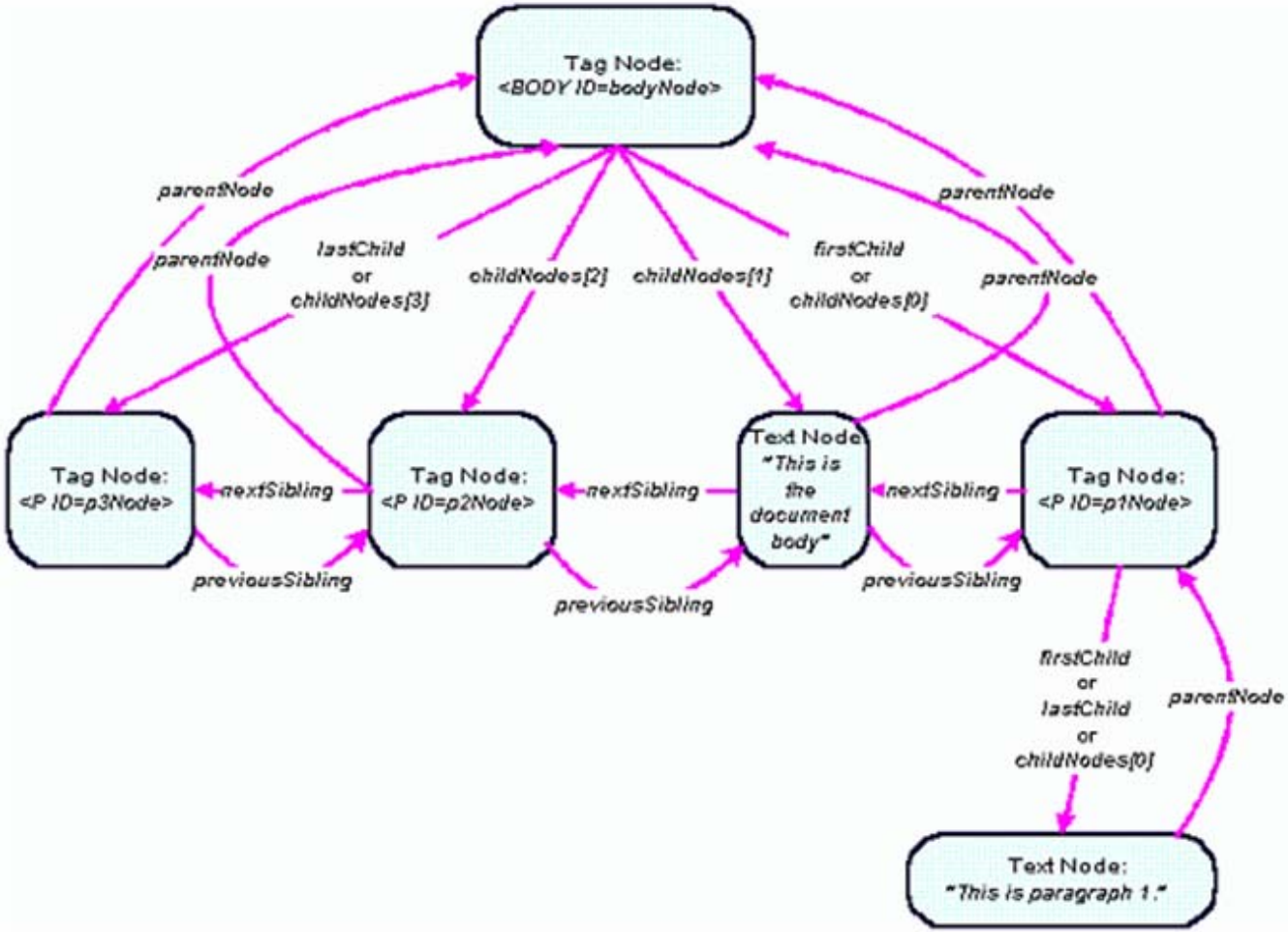
- HTML DOM = HTML Document Object Model
- Xem trang web như một cây gồm nhiều nút (node)
- Mỗi nút là một thành phần (tag HTML, thuộc tính, nội dung của tag)

- DOM định nghĩa một cách để truy xuất và điều khiển các thành phần trong 1 trang web

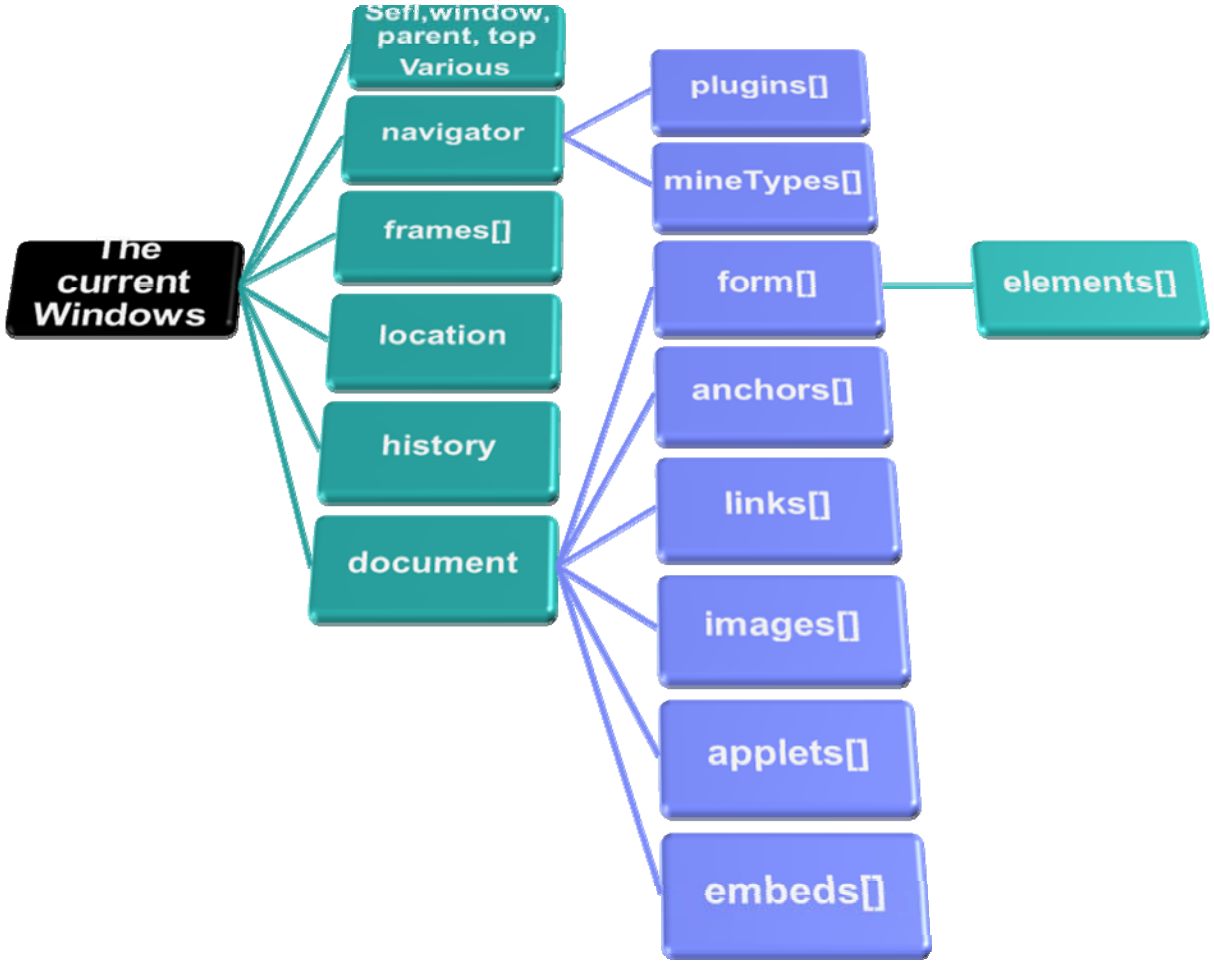
Giới thiệu về HTML DOM - Ví dụ



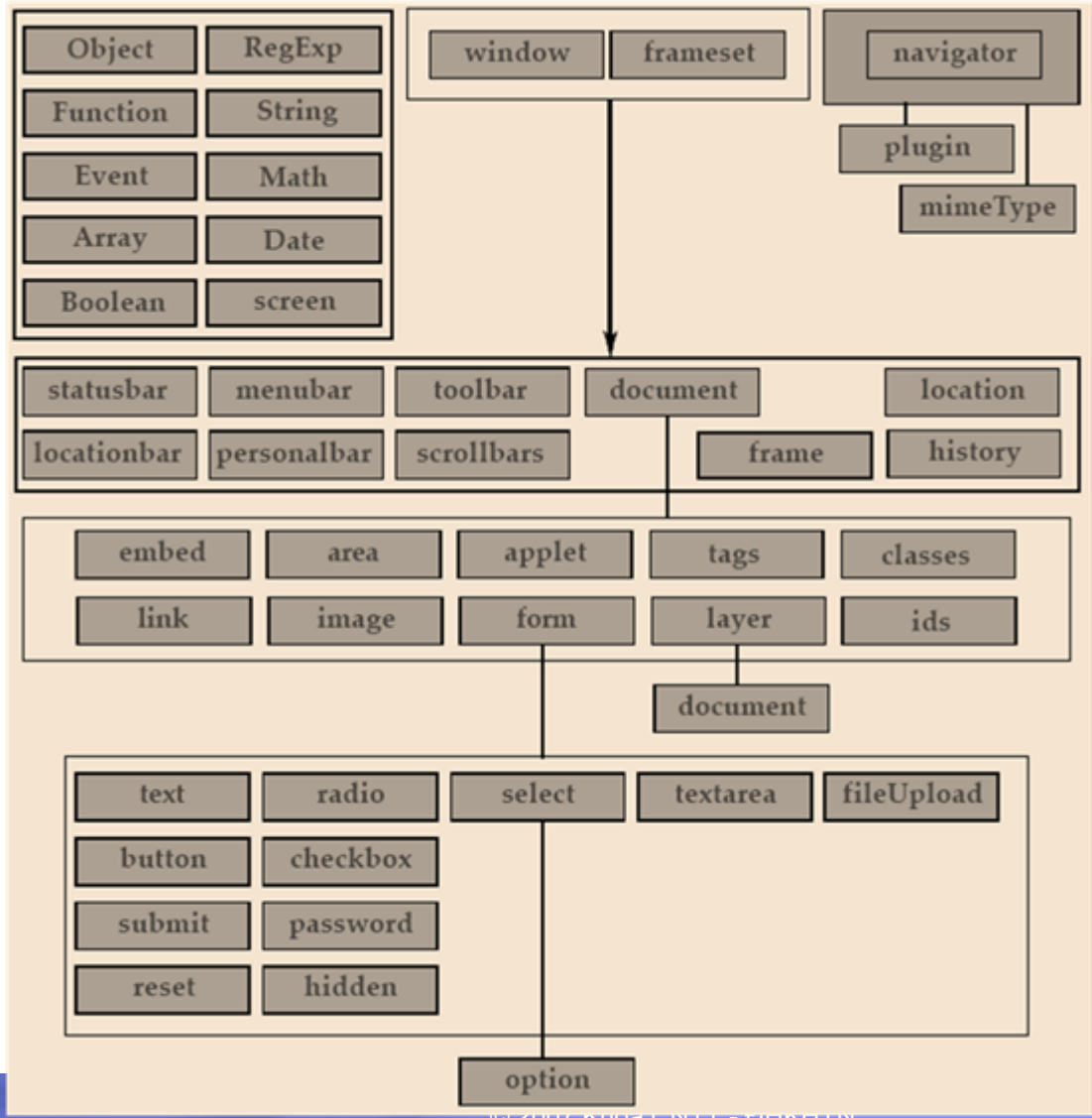
```
<HTML>
<HEAD>
<TITLE> Simple DOM
</HEAD>
<BODY ID="bodyNode
This is the docume
<P ID = "p2Node">
<P ID = "p3Node">
</BODY>
</HTML>
```



Giới thiệu về HTML DOM – Cấu trúc DOM đơn giản



Giới thiệu về HTML DOM – Cấu trúc DOM đầy đủ



Nội dung

- Giới thiệu về HTML DOM
- Thuộc tính (Property) và Phương thức (Method)
- Xử lý sự kiện (Event)

Property & Method – Cú pháp chung

- Mỗi đối tượng DOM đều có danh sách thuộc tính (Properties) và danh sách các phương thức (Method) tương ứng.

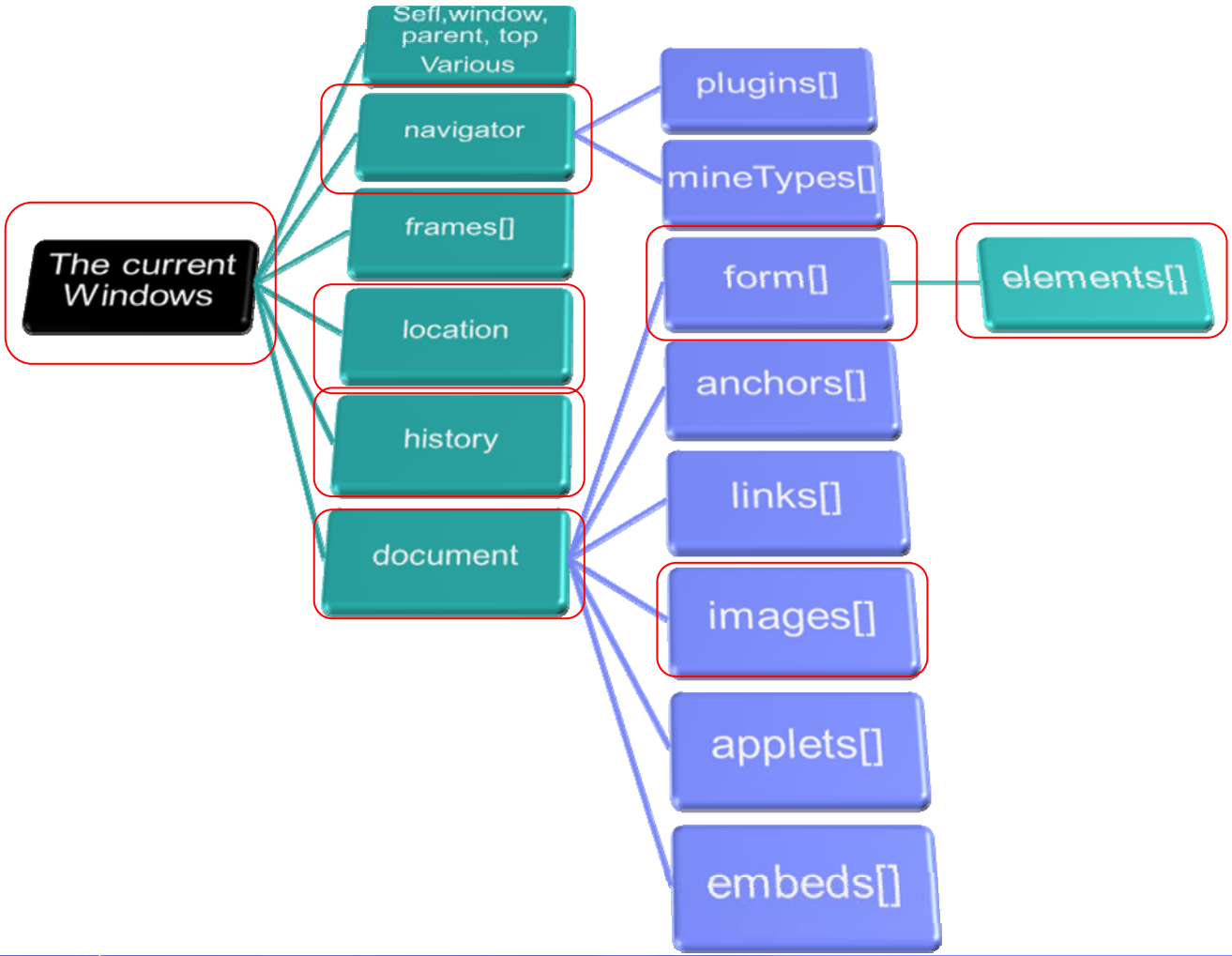
- **objectName.propertyName = value**
- Ví dụ:

```
document.bgColor = “blue”;
```

- **objectName.methodName()**
- Ví dụ:

```
window.focus();
```

Property & Method – Các loại Objects



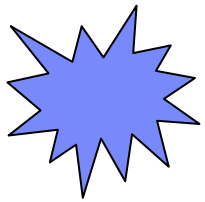
Property & Method – window Object

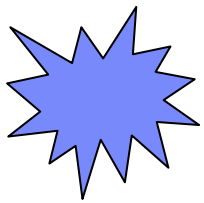
- Là thể hiện của đối tượng cửa sổ trình duyệt
- Tồn tại khi mở 1 tài liệu HTML
- Sử dụng để truy cập thông tin window
- Điều khiển các sự kiện xảy ra trong window
- Nếu tài liệu định nghĩa nhiều frame, browser tạo 1 window object cha và các window object con cho từng frame

Property & Method – window Object (tt)

Thuộc tính

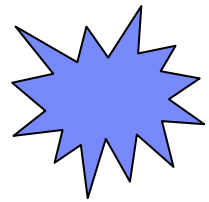
- document
- history
- location
- parent
- frames[]
- name
- status
- event
- screen
- …

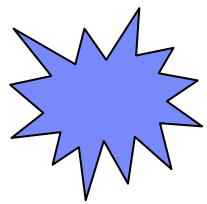




Phương thức

- alert
- confirm
- prompt
- blur
- focus
- open
- close
- setTimeout
- setInterval
- …



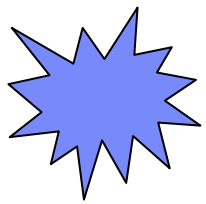


Property & Method – location Object

- Chứa thông tin về URL hiện tại

Thuộc tính

- hash
- host
- hostname
- href
- pathname
- port
- protocol
- search



Phương thức

- assign(url)
- reload()
- replace(url)

Property & Method – history Object

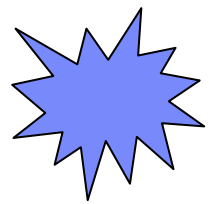
- Cung cấp danh sách các URL đã được duyệt bởi người dùng

Thuộc tính

- length

Phương thức

- back()
- go(url)
- forward()

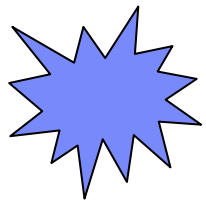


Property & Method – navigator Object

- Cung cấp thông tin về trình duyệt Browser

Thuộc tính

- appName
- appVersion
- appCodeName
- cookieEnabled
- online
- platform
- …



Phương thức

- javaEnabled()
- …

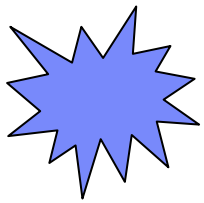
Property & Method – document Object

- Biểu diễn cho toàn bộ các thành phần trong 1 tài liệu HTML
- Dùng để lấy thông tin về tài liệu, các thành phần HTML và xử lý sự kiện

Property & Method – document Object (tt)

Thuộc tính

- bgColor
- fgColor
- aLinkColor
- linkColor
- vlinkColor
- lastModified
- location
- referrer
- title



Phương thức

- clear()
- close()
- open(…,…)
- write(text)
- writeln(text)
- getElementById(“id”)
- getElementByName(“Name”)
- getElementByTagName(“tagName”)
- createTextNode(“ text ")
- createElement(“HTMLtag")

Tập hợp

anchors [ ]
forms [ ]
frames [ ]
images [ ]
links [ ]

Property & Method – Image Object

- Truy xuất đến tag <img> trên trang web
- Có thể truy xuất thông qua :
  - documnet.images[index]
  - document.images[“ImageName”]
  - document.ImageName
- **Một số thuộc tính của Image Obj :**
  - alt, border, classname, title,
  - width, height, hspace, vspace,
  - id, name, src, lowsrc, longDesc,
  - isMap, complete

Property & Method – form Object

- Dùng để truy xuất đến tag <form> trên trang web
- Có thể truy xuất thông qua
  - documnet.forms[index]
  - document.forms[“FormName”]
  - document.FormName
- **Một số thuộc tính :**
  - action, method, id, name, target
  - classname, title, language, dir
  - elements[ ]
- **Một số phương thức :**
  - reset( ), submit( )

Property & Method – element Object

- Tương ứng với form field.

  document.*formName.controlName*

- Ví dụ:

```
<form name="searchForm" action="cgi-bin/search.pl">
  <input type="text" name="entry">
  <input type="submit" name="sender" value="Search">
</form>
```

có thể:

```
document.searchForm.entry
document.searchForm.elements[0]
document.forms["searchForm"].elements["entry"]
document.forms["searchForm"].entry
```

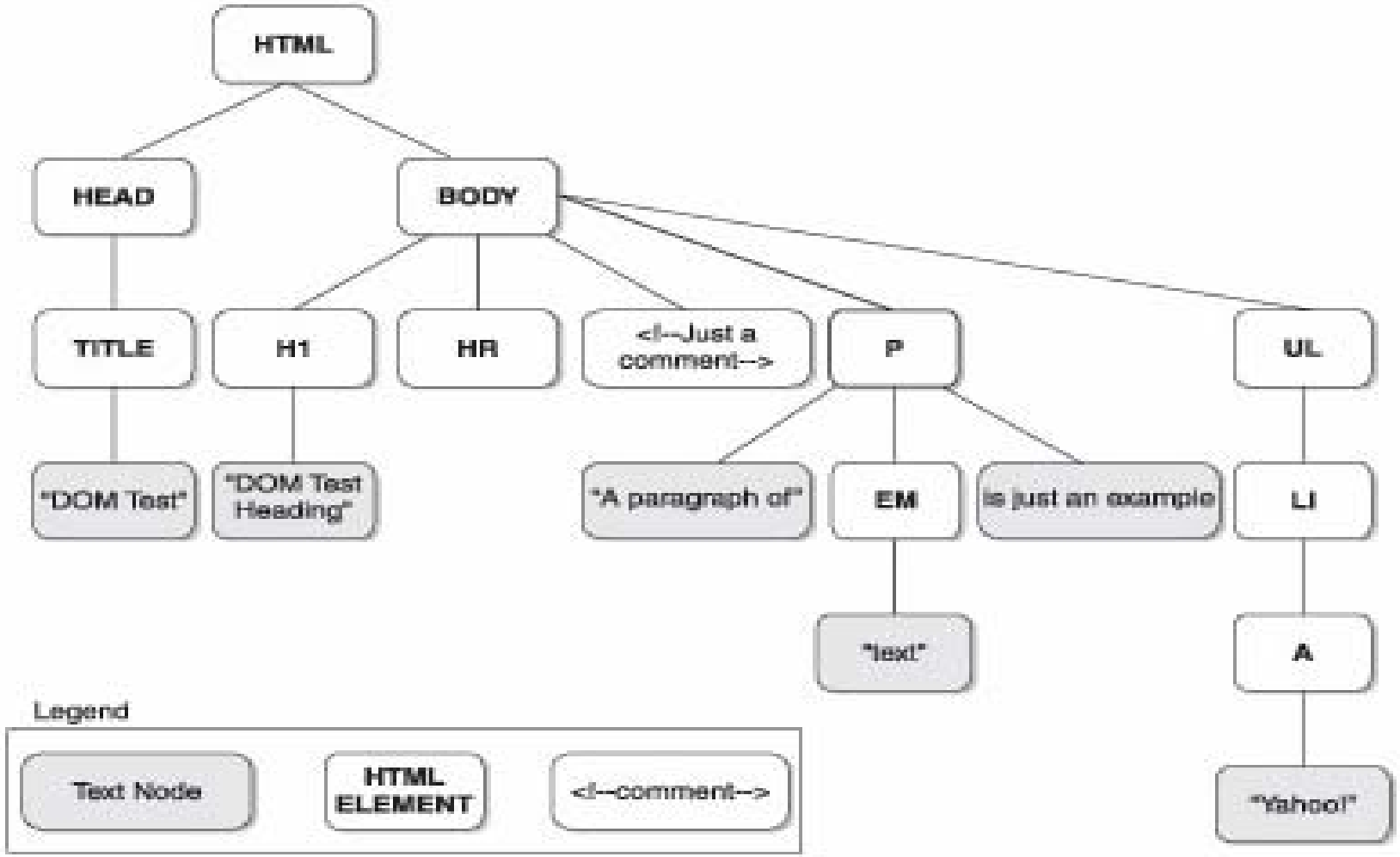
Đối tượng Document - DOM

- **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	...
2	Attribute	Thuộc tính của HTML hay XML element	align="center"
3	Text	Nội dung chứa trong HTML or XML element	This is a text fragment!
8	Comment	HTML comment	<!-- This is a comment -->
9	Document	Đối tượng tài liệu gốc, thường là element nằm ở cấp cao nhất trong cây cấu trúc của tài liệu	<html>
10	DocumentType	Định nghĩa loại tài liệu	<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

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

- **getElementsByName (name1)**

  Trả về danh sách node có giá trị của thuộc tính **name = name1**

```
Ví dụ:
var nodeList=
document.getElementsByName("name1");
for(var i=0;i<nodeList.length;++i)
{
      var nodeName = nodeList(i).nodeName;
      var nodeType  = nodeList(i).nodeType;
      var nodeValue = nodeList.item(i).nodeValue;
}
```

Đối tượng Document - DOM

- **getElementsByTagName (tagName)**

  Trả về danh sách node có **NodeName = tagName**

```
Ví dụ:
var nodeList= document.getElementsByTagName("img");
for(var i=0;i<nodeList.length;++i)
{
   var nodeName = nodeList(i).nodeName;
   var nodeType  = nodeList(i).nodeType;
   var nodeValue = nodeList.item(i).nodeValue;
}
```

Đối tượng Document - DOM

- createElement (nodeName)

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

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

    // <img src="images/test.gif" />
```

Đối tượng Document - DOM

- createTextNode (content)

Ví dụ:

```
var textNode = document.createTextNode("Newtext");
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.

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<img src="images/test.gif" />
//</p>
```

Đối tượng Document - DOM

- **insertBefore (newChild, childReference)**

  Chèn node **newChild** vào trước node **childReference** trong danh sách các node con của một node.

```
Ví dụ:
   // <p id="id1"><img src="images/165215.jpg">
   // </p>
   var pNode = document.getElementById("id1");
   var firstChild = pNode.firstChild;
   var newNode = document.createTextNode("Some
new text");
   pNode.insertBefore(newNode,firstChild);
   // <p id="id1">Some new text
         <img src="images/165215.jpg">
   // </p>
```

Đối tượng Document - DOM

- **removeChild (child)**

  Xóa node **child** ra khỏi danh sách các node con của node gọi thực hiện phương thức, giá trị trả về là node bị xóa.

```
Ví dụ:
   var pNode = document.getElementById("p1");
   if(pNode.hasChildNodes())
         pNode.removeChild(pNode.lastChild);
```

Đối tượng Document - DOM

- **replaceChild (newChild, oldChild)**

  Thay thế node **oldChild** bằng node **newChild** trong danh sách các node con của node hiện hành

```
Ví dụ:
var replace = document.getElementById(“isReplaced”);
if (replace)
{
   var newNode = document.createElement("strong");
   var newText =
         document.createTextNode("strongelement");
   newNode.appendChild(newText);
   replace.parentNode.replaceChild(newNode,replace);
}
```

Đối tượng Document - DOM

- **innerHTML**

  Chỉ định nội dung HTML bên trong một node.

```
Ví dụ:
//<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"
```

Đối tượng Document - DOM

- **setAttribute (attributeName , value)**

  Chỉ định attribute của một node với giá trị là value.

Ví dụ:

```
<font id="font1" >
    Some text
</font>
<script language="javascript" >
    var fontNode = document.getElementById("font1");
    fontNode.setAttribute("color","red");
    fontNode.setAttribute("size","5");
</script>
<font id="font1" color="red" size="5">
    Some text
</font>
```

Đối tượng Document - DOM

- **getAttribute (attributeName)**

  Lấy giá trị của một attribute trong node

```
Ví dụ:
   var font1 = document.getElementById("font1");
   alert(font1.getAttribute("color"));
```

- **removeAttribute (attributeName)**

  Hủy một attribute trong node

```
Ví dụ:
   font1.removeAttribute("color");
   font1.removeAttribute("size");
```

Đối tượng Document - DOM

- Thay đổi định dạng **CSS** của một node thông qua thuộc tính style

Ví dụ:

```
<p id="myParagraph" style="color: red;">This is a text</p>
<script language="javascript" >
        var node =
        document.getElementById("myParagraph");
        node.style.color = "green";
        node.style.fontSize = "14";
        node.style.backgroundColor = "yellow";
</script>
```

Đối tượng Document - DOM

- Thay đổi định dạng css thông qua thuộc tính className

Ví dụ:

```
<head>
<style type="text/css">
    .look1  {   color: black;  background-color: yellow;
font-style: normal; }
    .look2  {   background-color: orange; font-style:
italic; }
</style>
</head>
<body>
        <p id="p1" class="look1">
              this is my text     </p>
        <script language="javascript" >
              var pNode = document.getElementById("p1");
              pNode.className = "look2";
        </script>
```

Đối tượng Document - DOM

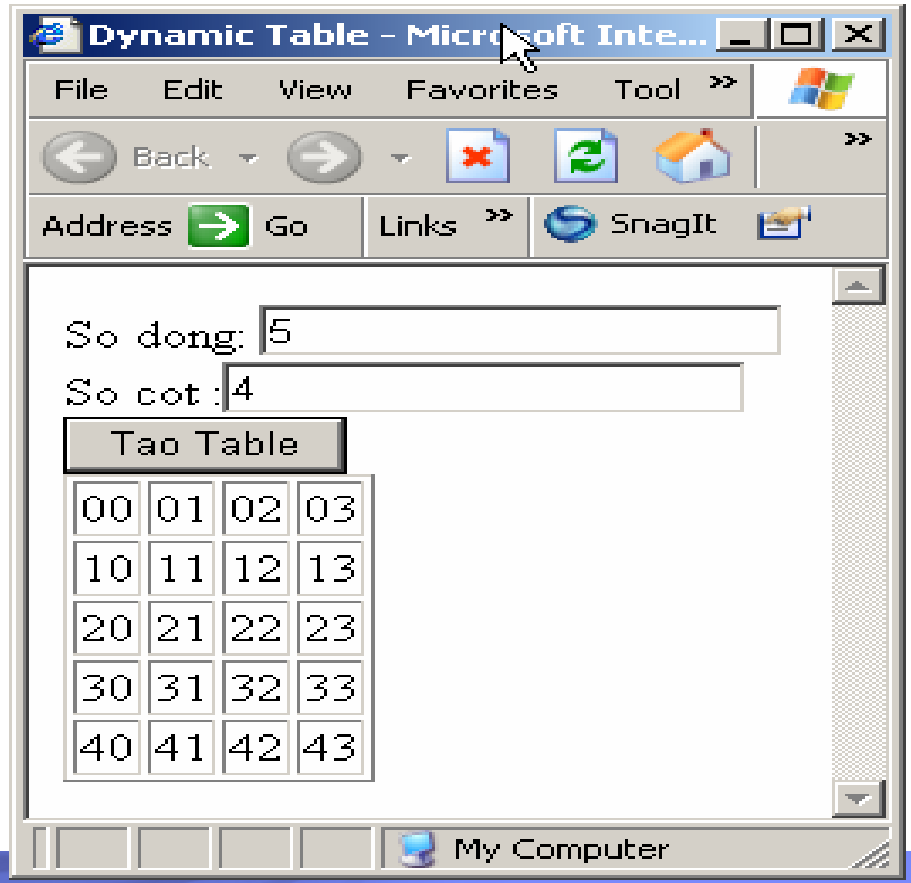
- Thay đổi tham chiếu đến file **CSS**

Ví dụ:

```
<head>
        <script language="javascript" >
         function changeSkin()
         {
              document.getElementById("myStyle").href =
                                        "css/style2.css";
         }
        </script>
      <link id="myStyle" rel="stylesheet"
      type="text/css" href="css/style1.css" />
</head>
<body>
       <p class="style1">
             Hello world
        </p>
      <input type="button" onclick="changeSkin()"
value="change skin" />
```

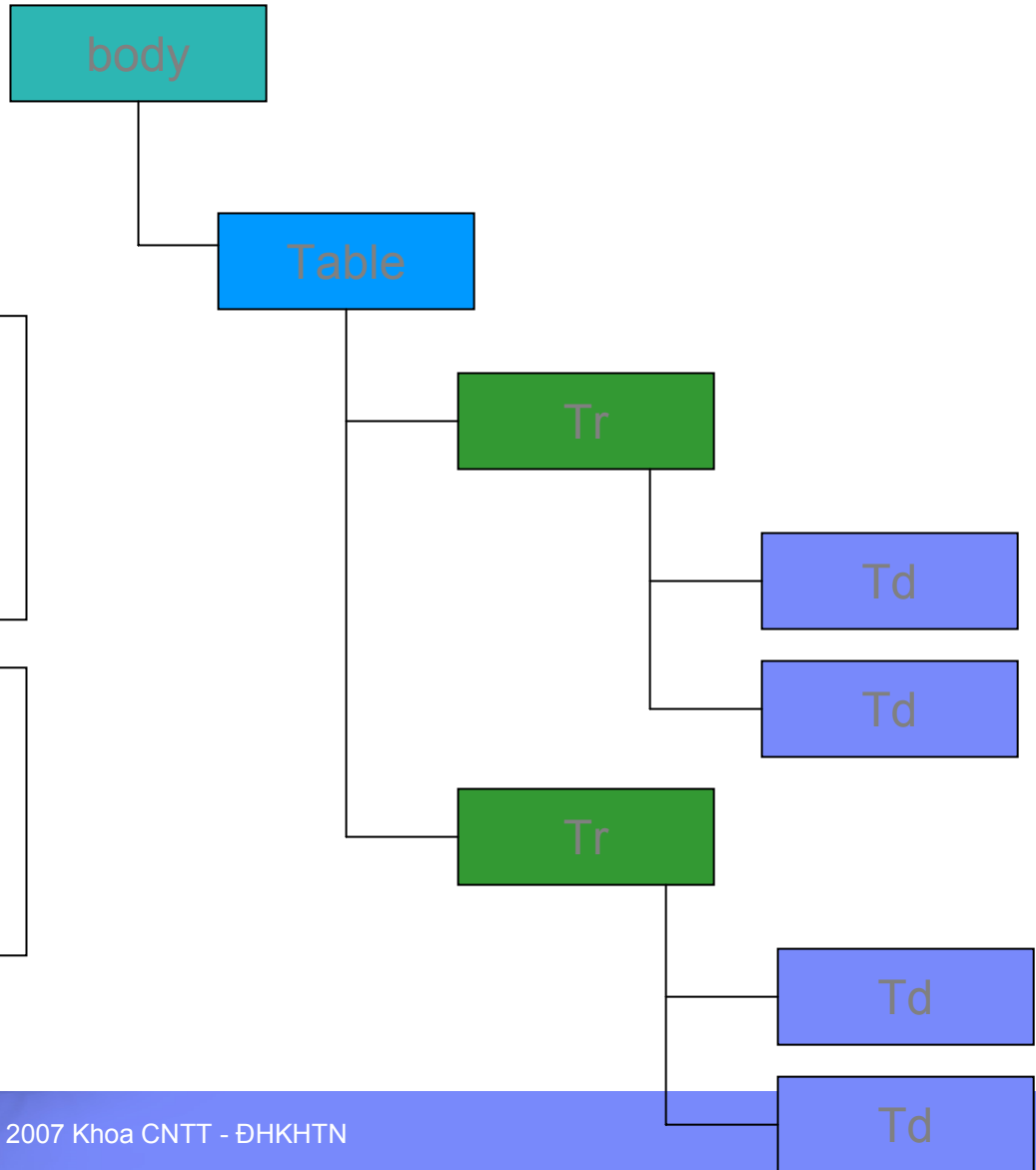
Ví dụ: Dynamic Table

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



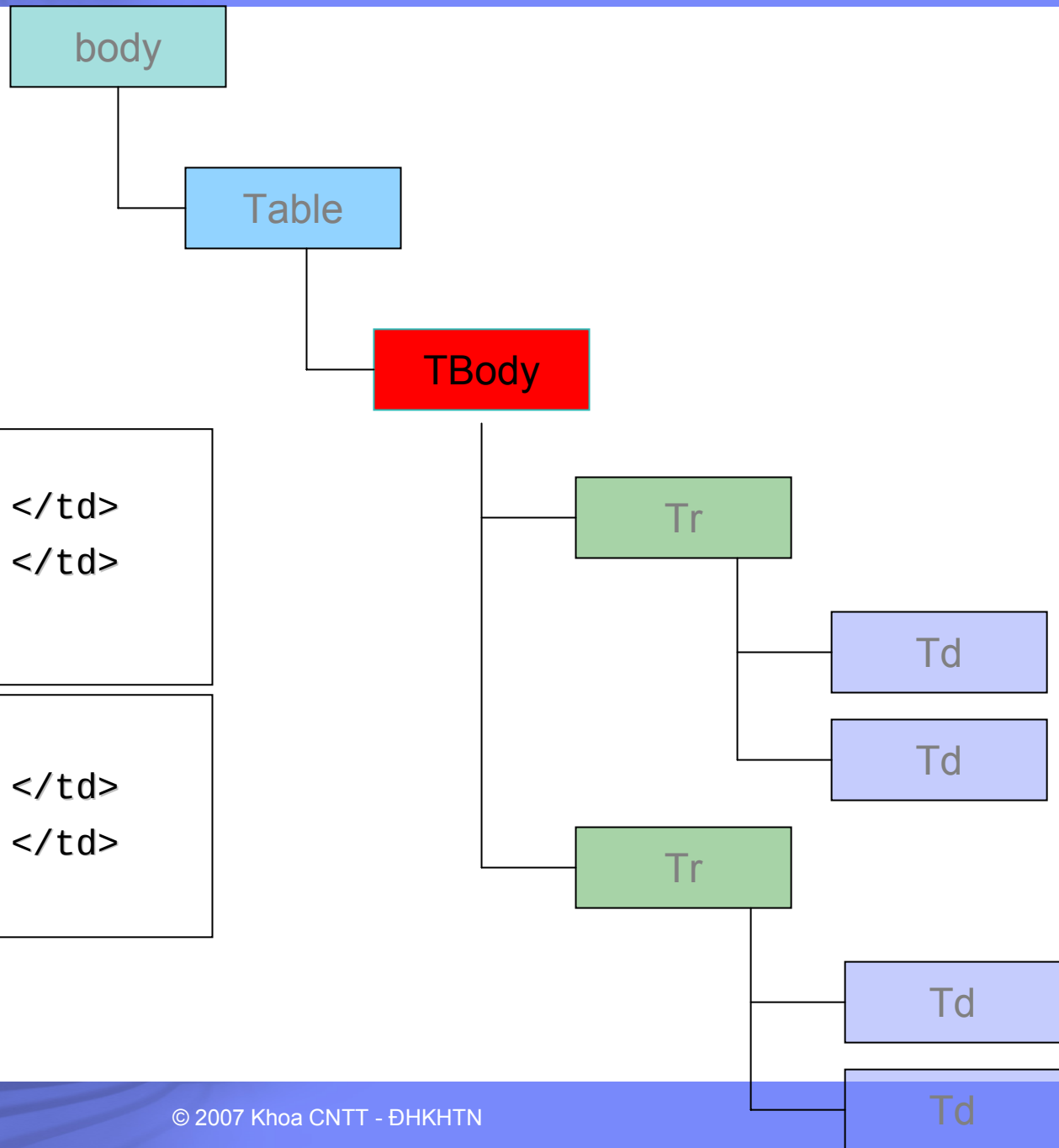
Ví dụ: Dynamic Table

```
<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);
        }
```

Nội dung

- Giới thiệu về HTML DOM
- Thuộc tính (Property) và Phương thức (Method)
- Xử lý sự kiện (Event)

Xử lý sự kiện

- Event Object
- Event Handler
- Xử lý sự kiện
- Ví dụ

Xử lý sự kiện

- Event Object
- Event Handler
- Xử lý sự kiện
- Ví dụ

Xử lý sự kiện – Event Object

- Events là các sự kiện xảy ra trên trang Web
- Do người dùng hoặc do hệ thống tạo ra
- Mỗi sự kiện sẽ liên quan đến một event object
- Cung cấp thông tin về event
  - Loại event
  - Vị trí con trỏ tại thời điểm xảy ra sự kiện

Xử lý sự kiện

- Event Object
- Event Handler
- Xử lý sự kiện
- Ví dụ

Xử lý sự kiện – Event Handler

- Giúp cho người lập trình bắt và xử lý các sự kiện của các đối tượng trong trang web.
- Cú pháp

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

- Ví dụ :

```
<INPUT TYPE=“button” NAME=“docode” onclick="DoOnClick();”>

<INPUT TYPE="button” NAME=“Button1”
  VALUE="Open Sesame!” onClick=”window.open('mydoc.html','newWin')”>
```

Xử lý sự kiện – Event Handler

Danh sách một số Event Handler thường sử dụng

onabort	onload
onblur	onmousedown
onchange	onmousemove
onclick	onmouseout
onerror	onmouseover
onfocus	onmouseup
onkeydown	onreset
onkeyup	onresize
onselect	onsubmit
	onunload

Xử lý sự kiện

- Event Object
- Event Handler
- Xử lý sự kiện
- Ví dụ

Xử lý sự kiện cho các thẻ HTML

- Cú pháp

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

- Ví dụ:

```
<body>

  <INPUT TYPE="button“ NAME="Button1“
       VALUE="OpenSesame!”
       onClick="window.open('mydoc.html','newWin');">

</body>
```

- Lưu ý: Dấu “…” và ‘…’

Xử lý sự kiện

- Xử lý bằng function

```
<SCRIPT LANGUAGE="JavaScript">
      function greeting() {
            alert("Welcome to my world");
      }
</SCRIPT>

<BODY onLoad="greeting()">
</BODY>
```

Xử lý sự kiện

- Xử lý bằng thuộc tính :
  - Gán tên hàm xử lý cho 1 object event

object.eventhandler = functionname;

```
<SCRIPT LANGUAGE="JavaScript">
        function greeting() {
                alert("Welcome to my world");
        }

        window.onload = greeting;
</SCRIPT>
```

Xử lý sự kiện – Danh sách các sự kiện của Form field

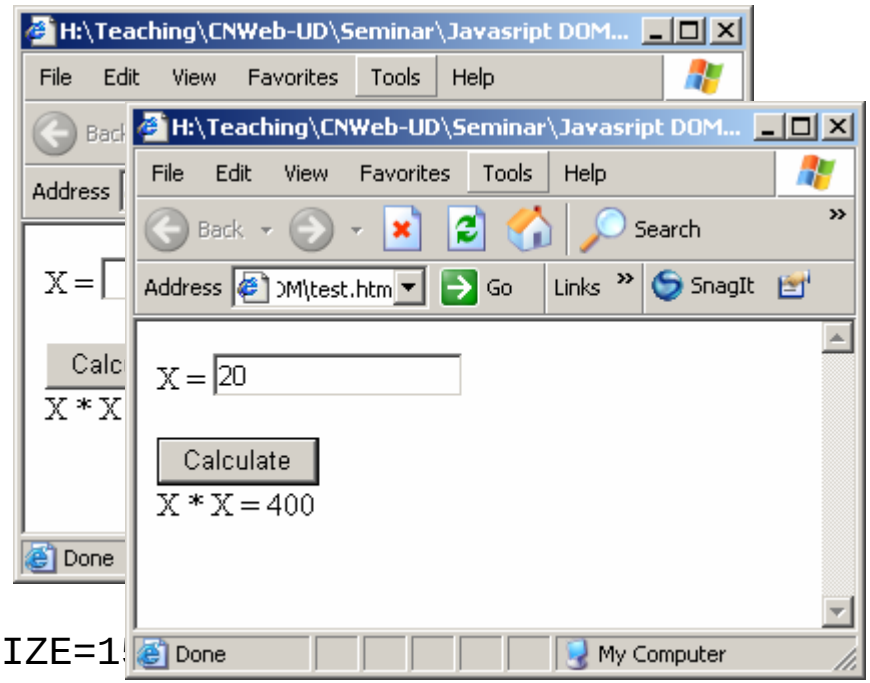
	Blur	Click	Change	Focus	Load	Mouseover	Select	Submit	Unload
Button		x							
Checkbox		x							
Document					x				x
Form								x	
Link		x				x			
Radio		x							
Reset		x							
Selection	x		x	x					
Submit		x							
Text	x		x	x			x		
Textarea	x		x	x			x		

Xử lý sự kiện

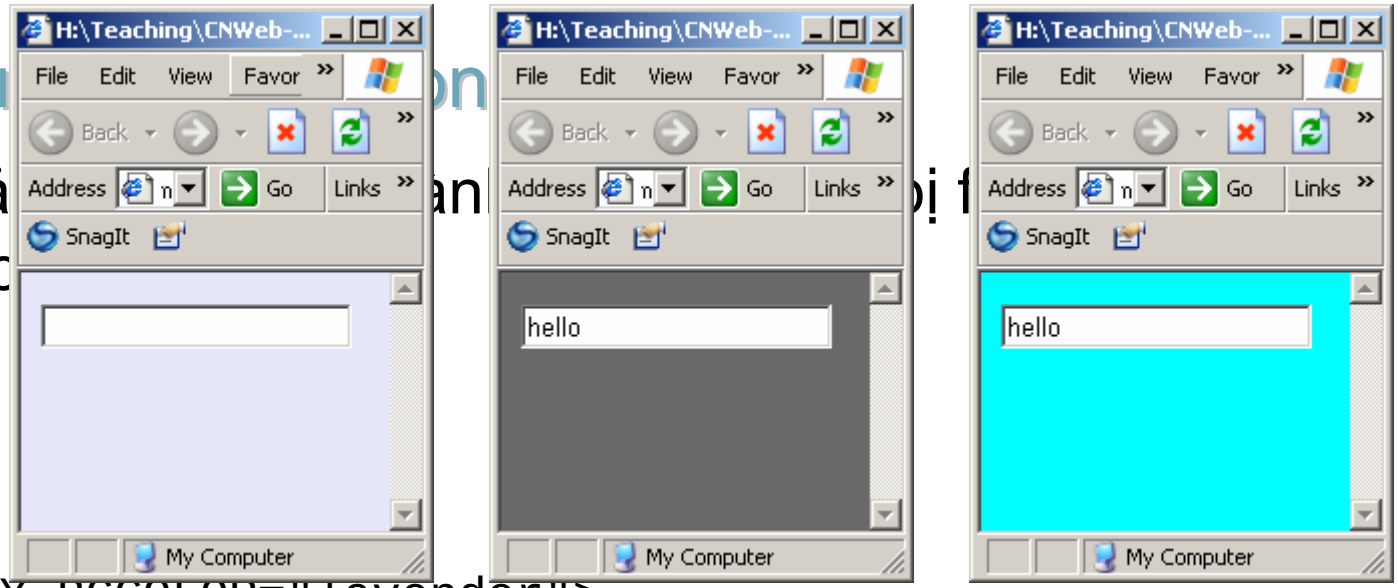
- Event Object
- Event Handler
- Xử lý sự kiện
- Ví dụ

Ví dụ: onclick Event

```
<SCRIPT LANGUAGE="JavaScript">
        function compute(frm){
            var x = frm.expr.value;
            result.innerHTML = x*x;
        }
</SCRIPT>
<FORM>
  X = <INPUT TYPE="text" NAME="expr" SIZE=1
  <BR><BR>
  <INPUT TYPE="button" VALUE="Calculate"
        ONCLICK="compute(this.form)">
  <BR>
  X * X = <SPAN ID="result"></SPAN>
</FORM>
```



Ví dụ



- Xả an ị f) và mất foc
- Ví

```
<BODY BGCOLOR="lavender">
  <FORM>
      <INPUT type="text" name="myTextbox"
          onblur="(document.bgColor='aqua')"
          onfocus="(document.bgColor='dimgray')">
  </FORM>
</BODY>
```

Ví dụ: onMouseOver - onMouseOut

```
<script language = "javascript">
  function showLink(num) {
        if (num==1)
              document.forms[0].elements[0].va
                    "You have selected Aptech
        } else {
           document.forms[0].elements[0].value="
        }
  }
</script>

<form> <input type=text size=60> </form>
<a href="#" onMouseOver="showLink(1)" onMouseOut="showLink(0)">
Aptech</a>
```