Programming of Web Pages

Lecture 5 – HTML DOM

What is the HTML DOM?

The HTML DOM is a standard object model and programming interface for HTML.

It defines:

The **HTML elements** as objects

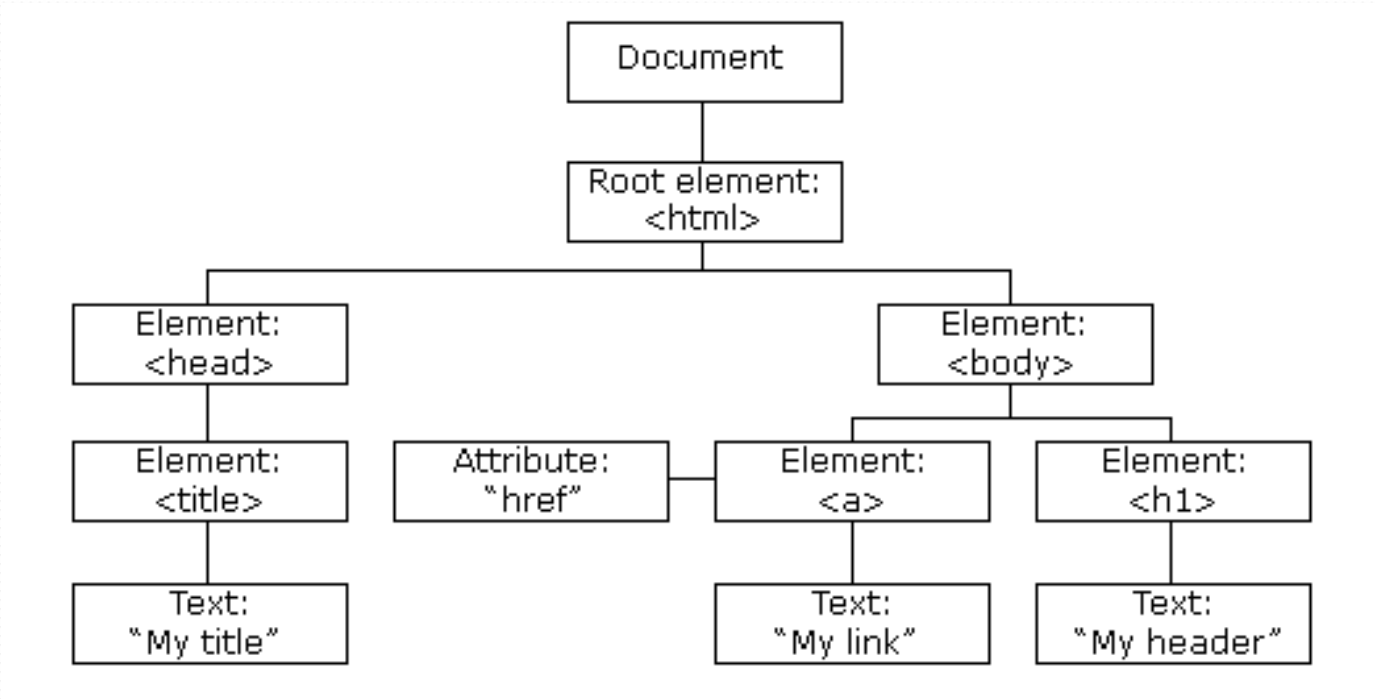
The **properties** of all HTML elements

The **methods** to access all HTML elements

The **events** for all HTML elements

The HTML DOM Tree of Objects

The **HTML DOM** model is constructed as a tree of **Objects**:



JavaScript and DOM

Using JavaScript and DOM you can:

- Change all the HTML elements in the page
- Change all the HTML attributes in the page
- Change all the CSS styles in the page
- Remove existing HTML elements and attributes
- Add new HTML elements and attributes
- React to all existing HTML events in the page

Methods and properties

Methods are actions you can perform (on HTML Elements)

Properties are values (of HTML Elements) that you can set or change

```
<body>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = "Hello World!";
</script>

</body>
```

Document object

The HTML DOM **document object** is the owner of all other objects in your web page.

If you want to access any element in an HTML page, you always start with accessing the document object.

Finding HTML Elements

```
<p id="intro">Hello World!</p>

<p>This example demonstrates the <b>getElementById</b> method!</p>

<script>
var myElement = document.getElementById("intro");
myElement.innerHTML = "Cześć!";
</script>
```

Cześć!

This example demonstrates the **getElementById** method!

Finding HTML Elements

```
<p>The DOM is very useful.</p>
<p>This example demonstrates the <b>getElementsByTagName</b> method</p>

<p id="demo"></p>

<script>
var x = document.getElementsByTagName("p");
document.getElementById("demo").innerHTML =
'Pierwszy paragraf: ' + x[0].innerHTML + '<br> ' +
'Drugi paragraf: ' + x[1].innerHTML;
</script>
```

The DOM is very useful.

This example demonstrates the **getElementsByTagName** method

Pierwszy paragraf: Hello World!
Drugi paragraf: The DOM is very useful.

Finding HTML Elements

```
<p>The DOM is very useful.</p>
<p>This example demonstrates the <b>getElementsByTagName</b> method</p>

<p id="demo"></p>

<script>
var x = document.getElementsByTagName("p");
document.getElementById("demo").innerHTML =
'Pierwszy paragraf: ' + x[0].innerHTML + '<br> ' +
'Drugi paragraf: ' + x[1].innerHTML;
</script>
```

The DOM is very useful.

Method **getElementsByClassName** works analogously

Drugi paragraf: The DOM is very useful.

Changing HTML Elements

```
<img id="myImage" src="smiley.gif">

<script>
document.getElementById("myImage").src = "landscape.jpg";
</script>
```

Changing HTML Elements

```
<p id="p2">Hello World!</p>

<script>
document.getElementById("p2").style.color = "blue";
</script>

<p>The paragraph above was changed by a script.</p>
```

Events

```
<h1 onclick="this.innerHTML='Ooops!'">Click on this text!</h1>
```

```
<h1 onclick="changeText(this)">Click on this text!</h1>

<script>
function changeText(id) {
    id.innerHTML = "Ooops!";
}
</script>
```

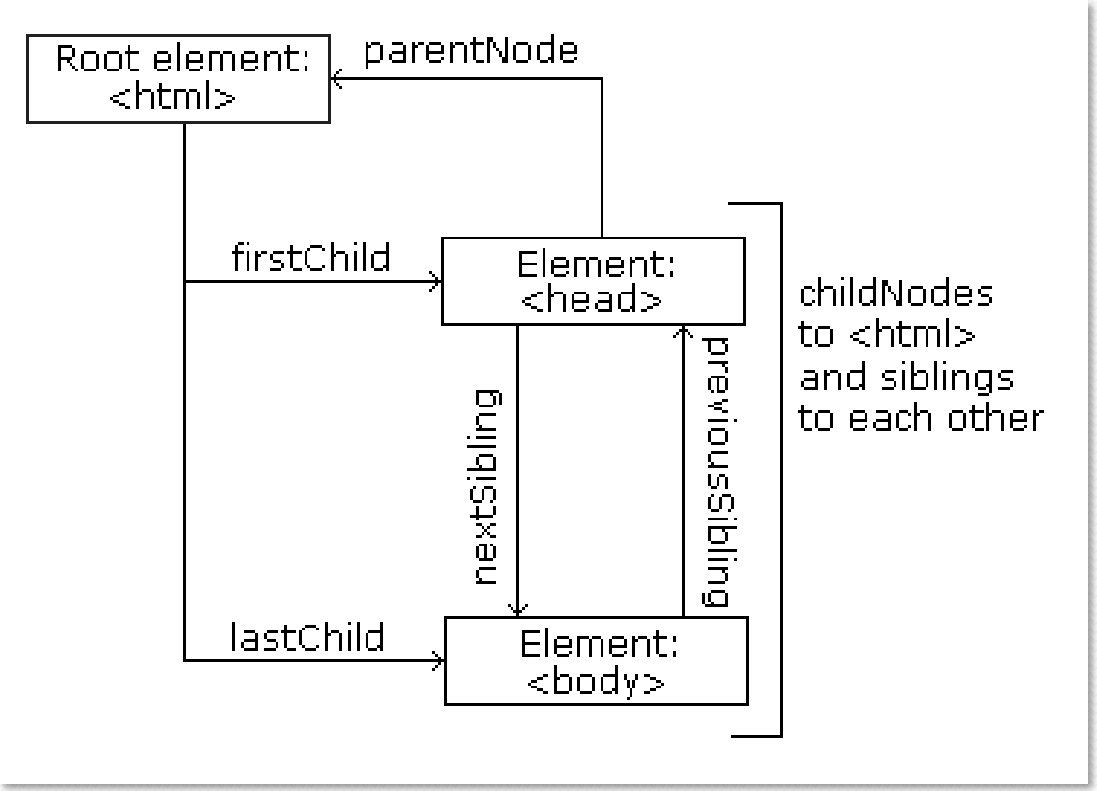
EventListener

```
<button id="myBtn">Try it</button>

<script>
document.getElementById("myBtn").addEventListener("click", myFunction);

function myFunction() {
    alert ("Hello World!");
}
</script>
```

EventListener – parameters

```
<button id="myBtn">Try it</button>

<p id="demo"></p>

<script>
var p1 = 5;
var p2 = 7;

document.getElementById("myBtn").addEventListener("click", function() {
    myFunction(p1, p2);
});

function myFunction(a, b) {
    var result = a * b;
    document.getElementById("demo").innerHTML = result;
}
</script>
```

Relations



- parentNode
- childNodes[*nodenumber*]
- firstChild
- lastChild
- nextSibling
- previousSibling

Adding elements

```
<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
var para = document.createElement("p");
var node = document.createTextNode("This is new.");
para.appendChild(node);

var element = document.getElementById("div1");
element.appendChild(para);
</script>
```

This is a paragraph.

This is another paragraph.

This is new.

Adding elements

```
<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
var para = document.createElement("p");
var node = document.createTextNode("This is new.");
para.appendChild(node);

var element = document.getElementById("div1");
var child = document.getElementById("p1");
element.insertBefore(para,child);
</script>
```

This is new.

This is a paragraph.

This is another paragraph.

Deleting elements

```
<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
var parent = document.getElementById("div1");
var child = document.getElementById("p1");
parent.removeChild(child);
</script>
```

Replacing elements

```
<div id="div1">
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>

<script>
var para = document.createElement("p");
var node = document.createTextNode("This is new.");
para.appendChild(node);

var parent = document.getElementById("div1");
var child = document.getElementById("p1");
parent.replaceChild(para,child);
</script>
```

This is new.

This is another paragraph.

Lists

```
var myNodelist = document.getElementsByTagName("p");
var i;
for (i = 0; i < myNodelist.length; i++) {
    myNodelist[i].style.backgroundColor = "red";
}
```