Pemrograman Web

6. DHTML (Dynamic HTML)

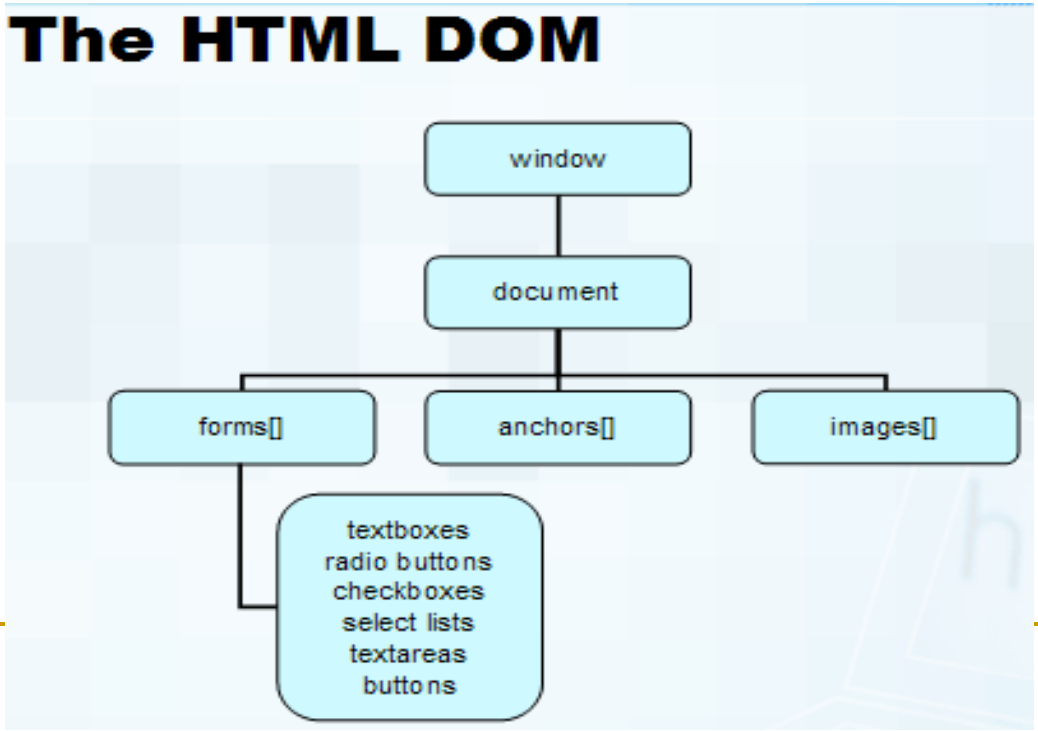
M. Udin Harun Al Rasyid, S.Kom, Ph.D
http://lecturer.eepis-its.edu/~udinharun
udinharun@eepis-its.edu

Table of Contents

Intoduction

- DHTML is the combination of HTML, Cascading Style Sheets, and Javascript used to create dynamic Web content.
- DHTML can move, hide, or animate as a result of user events.

- DHTML is a combination of three existing technologies meshed together by the **Document Object Model (DOM)**
  - HTML: For creating different page elements like text,image links.
  - CSS: to structured documents by separating content of documents and the presentation style of documents.
    CSS simplifies site maintenance and Web authoring.
  - JavaScript: to accesses and dynamically control the individual properties of both HTML and CSS



HTML DOM

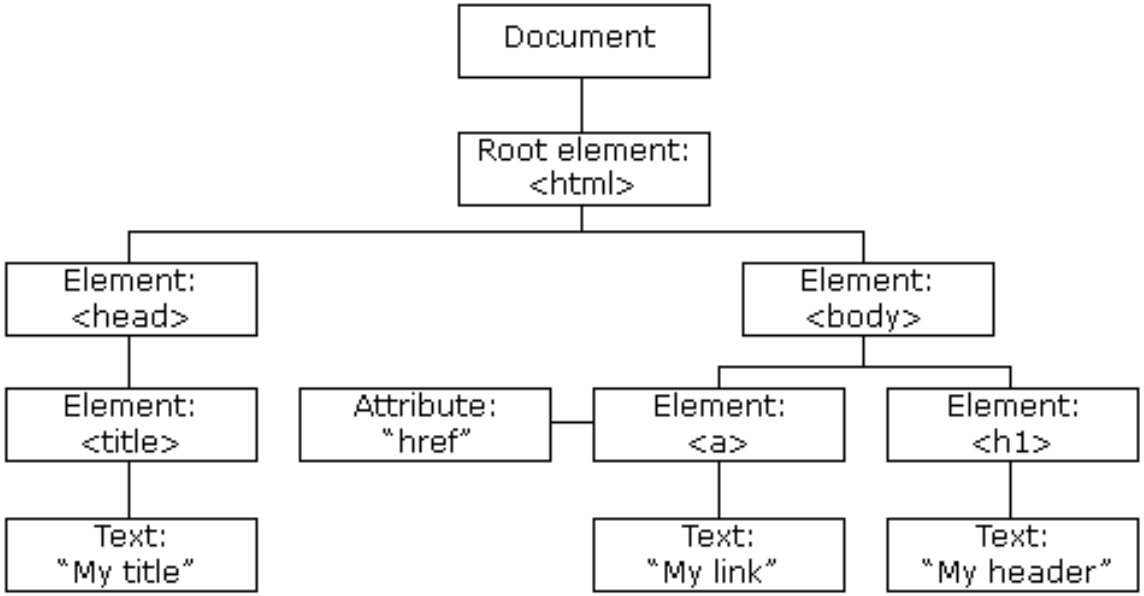
- The HTML DOM ?
  - Defines a standard way for accessing and manipulating HTML documents.
  - Defines the objects and properties of all HTML elements, and the methods to access them.
  - Standard for how to get, change, add, or delete HTML elements.

HTML DOM - Nodes

- According to the W3C HTML DOM standard, everything in an HTML document is a node.
- The HTML DOM views HTML documents as tree structures. The structure is called a Node Tree .
- All nodes in the tree can be accessed by JavaScript with the HTML DOM.
- All HTML elements (nodes) can be modified, and nodes can be created or deleted.

Node Parents, Children, and Siblings

- The terms parent, child, and sibling are used to describe the relationships.
  - In a node tree, the top node is called the root
  - Every node has exactly one parent, except the root (which has no parent)
  - A node can have any number of children
  - Siblings are nodes with the same parent



- Example:

```
<html>
  <head>
    <title>DOM Tutorial</title>
  </head>
  <body>
    <h1>DOM Lesson one</h1>
    <p>Hello world!</p>
  </body>
</html>
```

- The <html> node has no parent node; it is the root node
- The parent node of the <head> and <body> nodes is the <html> node
- The parent node of the "Hello world!" text node is the <p> node
- The <html> node has two child nodes: <head> and <body>
- The <head> node has one child node: the <title> node
- The <title> node also has one child node: the text node "DOM Tutorial"
- The <h1> and <p> nodes are siblings and child nodes of <body>
- The <head> element is the first child of the <html> element
- The <body> element is the last child of the <html> element
- The <h1> element is the first child of the <body> element
- The <p> element is the last child of the <body> element

HTML DOM - Methods

- The HTML DOM can be accessed with JavaScript.
- All HTML elements are defined as objects: Object methods and object properties.
  - **Method** is an action you can do (like add or modify an element).
  - **Property** is a value that you can get or set (like the name or content of a node).

- Example: **getElementById** method.

```
<html>
<body>
<p id="intro">Hello World!</p>
<script>
x=document.getElementById("intro");
document.write("<p>The text from the intro paragraph: " + x.innerHTML + "</p>");
</script>
</body>
</html>
```

Hello World!

The text from the intro paragraph: Hello World!

HTML DOM Objects - Methods and Properties

- Example of HTML DOM methods:
  - getElementById(*id*) - get the node (element) with a specified id
  - appendChild(*node*) - insert a new child node (element)
  - removeChild(*node*) - remove a child node (element)

- Example of HTML DOM properties:
  - innerHTML - the text value of a node (element)
  - parentNode - the parent node of a node (element)
  - childNodes - the child nodes of a node (element)
  - attributes - the attributes nodes of a node (element)

A Real Life Object Illustration

A person is an object.

- A person's **methods** could be eat(), sleep(), work(), play(), etc.
- All persons have these methods, but they are performed at different times.

- A person's **properties** include name, height, weight, age, eye color, etc.
- All persons have these properties, but their values differ from person to person.

Most Common DOM Object Methods

Method	Description
getElementById()	Returns the element that has an ID attribute with the a value
getElementsByTagName()	Returns a node list (collection/array of nodes) containing all elements with a specified tag name
getElementsByClass()	Returns a node list containing all elements with a specified class
appendChild()	Adds a new child node to a specified node
removeChild()	Removes a child node
replaceChild()	Replaces a child node
insertBefore()	Inserts a new child node before a specified child node
createAttribute()	Creates an Attribute node
createElement()	Creates an Element node
createTextNode()	Creates a Text node
getAttribute()	Returns the specified attribute value
setAttribute()	Sets or changes the specified attribute, to the specified value

HTML DOM - Properties

- The easiest way to get or replace the content of an element is by using the **innerHTML** property.

```
<html>
<body>
<p id="intro">Hello World!</p>
<script>
var txt=document.getElementById("intro").innerHTML;
document.write(txt);
</script>
</body>
</html>
```

HTML DOM - Access

- Different ways to access HTML element:
  - By using the getElementById() method
  - By using the getElementsByTagName() method
  - By using the getElementByClassName() method

```
<html>
<body>
<p>Hello World!</p>
<p>The DOM is very useful!</p>
<script>
x=document.getElementsByTagName("p");
document.write("Text of first paragraph: " + x[0].innerHTML);
document.write("<br> Text of second paragraph: " + x[1].innerHTML);
</script>
</body>
</html>
```

Hello World!

The DOM is very useful!

Text of first paragraph: Hello World!
Text of second paragraph: The DOM is very useful!

HTML DOM - Modifying

- Modifying the HTML DOM:
  - Changing HTML content
  - Changing CSS styles
  - Changing HTML attributes
  - Creating new HTML elements
  - Removing existent HTML elements
  - Changing event(handlers)

Changing HTML Content

```
<html>
<body>
<p id="p1">Hello World!</p>
<script>
document.getElementById("p1").innerHTML="New text!";
</script>
</body>
</html>
```

New text!

Changing HTML Style

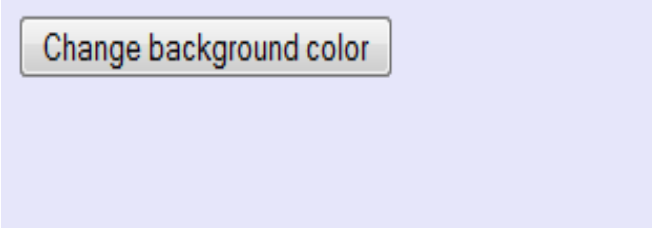
```
<html>
<body>
<p id="p1">Hello world!</p>
<p id="p2">Hello world!</p>
<script>
document.getElementById("p2").style.color="blue";
document.getElementById("p2").style.fontFamily="Arial";
document.getElementById("p2").style.fontSize="larger";
</script>
</body>
</html>
```

Hello world!

Hello world!

Changing HTML Content using Event

```
<html>
<body>
<input type="button"
onclick="document.body.style.backgroundColor='lavender';"
value="Change background color">
</body>
</html>
```



HTML DOM - Elements

Creating New HTML Elements - appendChild()

- To add a new element to the HTML DOM, you must create the element (element node) first, and then append it to an existing element.

```
<html>
<body>
<div id="d1">
<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("d1");
element.appendChild(para);
</script>
</body>
</html>
```

This is a paragraph.

This is another paragraph.

This is new.

HTML DOM - Events

- Events are actions that can be detected by JavaScript, example: A mouse click, a web page or an image loading, etc.

Mouse Events:

Event	Attribute	Description
click	onclick	The event occurs when the user clicks on an element
dblclick	ondblclick	The event occurs when the user double-clicks on an element
mousedown	onmousedown	The event occurs when a user presses a mouse button over an element
mousemove	onmousemove	The event occurs when a user moves the mouse pointer over an element
mouseover	onmouseover	The event occurs when a user mouse over an element
mouseout	onmouseout	The event occurs when a user moves the mouse pointer out of an element
mouseup	onmouseup	The event occurs when a user releases a mouse button over an element

Finish