JavaScript and jQuery Course

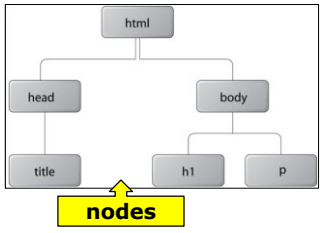
Session 09 DOM

DOM

DOM – hierarchical tree representation of document

- collection of *nodes* that represent the web page
- JavaScript can modify the DOM

- descendants
- parents
- children
- siblings



2

Node Properties

`nodeValue` ←	Returns the text for a text, attribute, comment node
`parentNode`	Returns the parent of that node
`childNodes`	Returns an array of node objects
`firstChild` ←	Returns a node object for the first child node
`lastChild`	Returns a node object for the last child node
`nextElementSibling`	Returns node object for next sibling

If no node exists, returns `null`

3

Methods of document interface

`getElementById(id)` ←

returns the element object that matches the id

`getElementsByTagName(tag_name)`

Elements = plural

returns an array of elements that matches the `tag_name` value

```
<input name="fname" id="fname" type="text">
<input name="lname" id="lname" type="text">
```

all `<input>` elements

4

Methods of document interface

`getElementsByTagName(tag_name)`

returns an array of elements that matches the `tag_name` value

```
<input name="fname" id="fname" type="text">
<input name="lname" id="lname" type="text">
```

```
// clear out text box values
   for (var i = 0;  i < document.getElementsByTagName('input').length;  i++) {

        document.getElementsByTagName('input')[i].value = " ";
   }
document.getElementsByTagName('input')[0].value = " ";
```

array

5

Methods of document interface

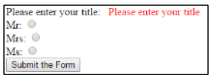
`getElementsByName(name_value)`

returns an array of elements that matches the `name_value` value

Elements = plural

```
<input name="title" type="radio" id="Mr" value="Mr">
<input name="title" type="radio" id="Mrs" value="Mrs">
<input name="title" type="radio" id="Ms" value="Ms">
```

all elements with `name` attribute
–used by the web server



6

Methods of document interface

`getElementsByName(name_value)`

returns an array of elements that matches the `name_value` value

```
// determine if any title was checked

for (var i = 0;    i < document.getElementsByName('title').length;     i++) {

     if (document.getElementsByName('title')[i].checked) {
          titleOK = true;
     }

}

document.getElementsByName('title')[0].checked
```

array

7

Methods of document interface

`getElementsByClassName(class_name)`

returns an array of elements that matches the `class_name` value

Elements = plural

```
<span class="err">Please enter…</span>
```

all elements that have a class of `err`

8

Methods of document interface

`getElementsByClassName(class_name)`

returns an array of elements that matches the `class_name` value

```
<span class="err"> </span>
```

```
// clear out error messages
for (var i = 0;  i < document.getElementsByClassName('err').length;  i++){
    document.getElementsByClassName('err')[i].firstChild.nodeValue = "";
}
```

array

```
document.getElementsByClassName('err')[0]
```

9

DOM - How to modify the text of an HTML element – firstChild.nodeValue

Step 1: access the element - create an object reference to it

```
document.getElementById('myh3')
```

```
<h3 id="myh3">Text will change</h3>
```

10

DOM - How to modify the text of an HTML element - firstChild.nodeValue

Step 2: access the text node - use the `firstChild` property

```
document.getElementById('myh3').firstChild
```

```
<h3 id="myh3">Text will change</h3>
```

11

DOM - How to modify the text of an HTML element - firstChild.nodeValue

Step 3: use `nodeValue` property - access the text for the text node

```
document.getElementById('myh3').firstChild.nodeValue
```

```
<h3 id="myh3">Text will change</h3>
```

12

DOM - How to modify the text of an HTML element - firstChild.nodeValue

Step 4: set a value

```
document.getElementById('myh3').firstChild.nodeValue =
    "Hello";
```

```
<h3 id="myh3">Text will change</h3>
```

13

DOM

Why not use innerHTML?

```
document.getElementById('myh3').innerHTML = "Hello";
```

innerHTML is part of JS

It is not part of the DOM

14

Property of document interface

`document.getElementById("").nextElementSibling`

returns an object for the next element sibling, or returns null if it does not exist

```
<input type="text" id="fname" name="fname">
<span class="err"> </span>
```

```
// display error message without using an id
if(document.getElementById('fname') == "") {

    document.getElementById('fname').nextElementSibling.firstChild.nodeValue
    = "Please enter your first name";
}
```

15

Method of document interface

```
document.el.addEventListener("click", functionName)

function calc() {..}

<input type= "button " id= " btn">
```

```
Var btn = document.getElementById("btn");

document.btn.addEventListener("click", calc);

vs

document.btn.onclick = calc;
```

16

Methods of document interface - not in text
querySelector(), querySelectorAll()

```
// Access the first match of the selector
document.querySelector('.myClass');


// Return a NodeList of all instances of .myClass
document.querySelectorAll('.myClass');


// Access the myID id - unique
document.querySelector('#myID');


// Return a NodeList of all 'div' instances
document.querySelectorAll('div');

//Access the first match of the selector
document.querySelector('ul.someList li:last-child');
```

17

DOM access methods - return object reference

How we can access an element(s) on the page

```
document.getElementById(id)

document.getElementsByClassName(className)

document.getElementsByTagName(tagName)

document.querySelector(cssSelector)

document.querySelectorAll(cssSelector)
```

18

Traversing the DOM

How we can access an element(s) on the page

```
firstElementChild / firstChild

lastElementChild / lastChild

nextElementChild / nextElementSibling

previousElementChild / previousElementSibling

childNodes (has a length property)

childNodes[i]

Children / children[i] – only child elements

parentElement / parentNode
```

19

Modifying attributes

How we can add and remove HTML attributes

```
has Attribute() – t/f

setAttribute('attr', 'value')

removeAttribute('attr')
```

20

Modifying styles – Traditional JavaScript

Uses JS syntax, not CSS syntax – style property

```
var headEl – document.getElementsByTagName("h1“)[0];

headEl.style.color = "hotpink";

headEl.style.backgroundColor = "salmon";
```

JS Syntax

21

Manipulating the DOM

How we can add and remove HTML elements and text

document refers to <body> / you can identify a parent node also

```
document.createElement()

document.createTextNode()

document.appendChild()

doccument.insertBefore(newNode/what, referenceNode/where)

document.replaceChild(newChild, oldChild)

document.removeChild(child)
```

22

Methods of document interface - not in text

createElement(), createTextNode(), appendChild()

```
// create a new div element
var newB = document.createElement("b");

// and give it some content
var newContent =
      document.createTextNode("Hello from a new b!");

//add the text node to the newly created <b>
newB.appendChild(newContent);

// add the newly created element and its content
// into the DOM
var thePara = document.getElementById("para");

thePara.appendChild(newB);
```

23