University Of Babylon
College of Computer Technology
Department Of Information Networks

Web Page Design

XML DOM – Part II

HAIDER M. HABEEB
MSC. INFORMATION TECHNOLOGY

DOM Attribute List

- The attributes property of an element node returns a list of attribute nodes.
- It's called a **named node map**
- An attribute list keeps itself up-to-date. If an attribute is deleted or added, the list is automatically updated.

DOM Attribute List

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.getElementsByTagName("book")[0].attributes;

document.write(x.getNamedItem("category").nodeValue);
document.write("<br />" + x.length);
```

cooking
1

XML DOM – Navigating Nodes

- Node relationships are defined as properties to the nodes:
  - parentNode
  - childNodes
  - firstChild
  - lastChild
  - nextSibling
  - previousSibling

XML DOM - Navigating Nodes

- **DOM - Parent Node**

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.getElementsByTagName("book")[0];
document.write(x.parentNode.nodeName);
```

XML DOM - Navigating Nodes

- **Avoid Empty Text Nodes**
  - Firefox, and some other browsers, will treat empty white-spaces or new lines as text nodes, Internet Explorer will not.

```
n=xmlDoc.getElementsByTagName("book")[0];
y=n.firstChild;
while (y.nodeType!=1)
  {
  y=y.nextSibling;
  }
}
```

XML DOM - Navigating Nodes

- **Get the First Child Element**

```
<script type="text/javascript">
//check if the first node is an element node
function get_firstChild(n)
{
y=n.firstChild;
while (y.nodeType!=1)
  {
  y=y.nextSibling;
  }
return y;
}
</script>
```

XML DOM - Navigating Nodes

- **Get The Next Sibling**

```
function get_nextSibling(n)
{
y=n.nextSibling;
while (y.nodeType!=1)
  {
  y=y.nextSibling;
  }
return y;
}
```

```
x=get_nextSibling(xmlDoc.getElementsByTagName("title")[0]);
document.write(x.nodeName);
```

XML DOM - Navigating Nodes

- **Get The Last Child**

```
function get_lastChild(n)
{
y=n.lastChild;
while (y.nodeType!=1)
  {
  y=y.previousSibling;
  }
return y;
}
```

```
x=get_lastChild(xmlDoc.getElementsByTagName("book")[0]);
document.write(x.nodeName);
```

XML DOM - Navigating Nodes

- **Get the Previous Sibling**

```
function get_previousSibling(n)
{
y=n.previousSibling;
while (y.nodeType!=1)
  {
  y=y.previousSibling;
  }
return y;
}
```

```
x=get_previousSibling(xmlDoc.getElementsByTagName("price")[0]);
document.write(x.nodeName);
```

XML DOM Get Node Values

- The **nodeValue** property is used to get the text value of a node.

```
x=xmlDoc.getElementsByTagName("title")[0];
y=x.childNodes[0];
txt=y.nodeValue;
```

- The **getAttribute()** method returns the value of an attribute.

```
xmlDoc=loadXMLDoc("books.xml");
txt=xmlDoc.getElementsByTagName("title")[0]
.getAttribute("lang");
```

XML DOM Get Node Values

- **Find the result of the following DOM?**

```
xmlDoc=loadXMLDoc("books.xml");
x=xmlDoc.getElementsByTagName('book');

for (i=0;i<x.length;i++)
{
document.write(x[i].getAttribute('category'));
document.write("<br />");
}
```

XML DOM Get Node Values

- Get an Attribute Value - **getAttributeNode()**
- The getAttributeNode() method returns an attribute **node**.

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.getElementsByTagName("title")[0].
getAttributeNode("lang");
txt=x.nodeValue;
document.write(txt);
```

XML DOM Change Node Values

- The **nodeValue** property is used to change a node value.
- The **setAttribute()** method is used to change an attribute value.

XML DOM Change Node Values

- The **nodeValue** property is used to change a node value.

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.getElementsByTagName("title")[0].
childNodes[0];
x.nodeValue="Easy Cooking";
```

XML DOM Change Node Values

- **Change an Attribute Using setAttribute()**

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.getElementsByTagName('book');
x[0].setAttribute("category","food");
```

XML DOM Change Node Values

- If the attribute does not exist, a new attribute is created (with the name and value specified).

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.getElementsByTagName('book');
x[0].setAttribute("edition","first");
```

XML DOM Change Node Values

- **Change an Attribute Using nodeValue**

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.getElementsByTagName("book")[0]
y=x.getAttributeNode("category");
y.nodeValue="food";
```

XML DOM Remove Nodes

- The **removeChild()** method removes a specified node.
- The **removeAttribute()** method removes a specified attribute.

XML DOM Remove Nodes

- **Remove an Element Node**

```
y=xmlDoc.getElementsByTagName("book")[0];
xmlDoc.documentElement.removeChild(y);
```

- **Remove Myself - Remove the Current Node**

```
x=xmlDoc.getElementsByTagName("book")[0];
x.parentNode.removeChild(x);
```

XML DOM Remove Nodes

- **Remove a Text Node**

```
x=xmlDoc.getElementsByTagName("title")[0];
y=x.childNodes[0];
x.removeChild(y);
```

- **Clear a Text Node**

```
x=xmlDoc.getElementsByTagName("title")[0]
.childNodes[0];
x.nodeValue="";
```

XML DOM Remove Nodes

- **Remove an Attribute Node by Name**

```
x=xmlDoc.getElementsByTagName("book");
x[0].removeAttribute("category");
```

XML DOM Create Nodes

- **Create a New Element Node**
- The createElement() method creates a new element node:

```
xmlDoc=loadXMLDoc("books.xml");

newel=xmlDoc.createElement("edition");

x=xmlDoc.getElementsByTagName("book")[0];
x.appendChild(newel);
```

XML DOM Create Nodes

- **Create a New Attribute Node**
- The createAttribute() is used to create a new attribute node:

```
xmlDoc=loadXMLDoc("books.xml");

newatt=xmlDoc.createAttribute("edition");
newatt.nodeValue="first";

x=xmlDoc.getElementsByTagName("title");
x[0].setAttributeNode(newatt);
```

XML DOM Create Nodes

- **Create an Attribute Using setAttribute()**
- Since the setAttribute() method creates a new attribute if the attribute does not exist, it can be used to create a new attribute.

```
x=xmlDoc.getElementsByTagName('book');
x[0].setAttribute("edition","first");
```

XML DOM Create Nodes

- **Create a Text Node**
- The createTextNode() method creates a new text node:

```
newel=xmlDoc.createElement("edition");
newtext=xmlDoc.createTextNode("first");
newel.appendChild(newtext);

x=xmlDoc.getElementsByTagName("book")[0];
x.appendChild(newel);
```

XML DOM Add Nodes

- **Add a Node - appendChild()**
- The appendChild() method adds a child node to an existing node.
- The new node is added (appended) after any existing child nodes.

```
newel=xmlDoc.createElement("edition");

x=xmlDoc.getElementsByTagName("book")[0];
x.appendChild(newel);
```

XML DOM Add Nodes

- **Insert a Node - insertBefore()**
  - The insertBefore() method is used to insert a node before a specified child node.
  - This method is useful when the position of the added node is important:

```
newNode=xmlDoc.createElement("book");

x=xmlDoc.documentElement;
y=xmlDoc.getElementsByTagName("book")[3];

x.insertBefore(newNode,y);
```

XML DOM Add Nodes

- **Add Text to a Text Node - insertData()**
  - The insertData() method inserts data into an existing text node.
  - The insertData() method has two parameters:
  - offset - Where to begin inserting characters (starts at zero)
  - string - The string to insert

```
x=xmlDoc.getElementsByTagName("title")[0]
.childNodes[0];

x.insertData(0,"Easy ");
```

XML DOM Clone Nodes

- **Copy a Node**
- The cloneNode() method creates a copy of a specified node.
- The cloneNode() method has a parameter (true or false).
- This parameter indicates if the cloned node should include all attributes and child nodes of the original node.

XML DOM Clone Nodes

```
oldNode=xmlDoc.getElementsByTagName('book')[0];

newNode=oldNode.cloneNode(true);
xmlDoc.documentElement.appendChild(newNode);

//Output all titles
y=xmlDoc.getElementsByTagName("title");
for (i=0;i<y.length;i++)
{
document.write(y[i].childNodes[0].nodeValue);
document.write("<br />");
}
```