Lab Sheet 03 SWT32051 – Practical for Service Oriented Web Application

XML DOM Continue

Get the Value of an Attribute.

Attribute nodes have text values. This can be done using the **getAttribute()** method or using the **nodeValue** property of the attribute node.

i.Get an Attribute Value - getAttribute() methods

Try this example

```
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
   if (this.readyState == 4 && this.status == 200) {
      myFunction(this);
   }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
   var x, i, xmlDoc, txt;
   xmlDoc = xml.responseXML;
   txt = "";
   x = xmlDoc.getElementsByTagName('book');
   for (i = 0; i < x.length; i++) {
      txt += x[i].getAttribute('category') + "<br>";
   }
   document.getElementById("demo").innerHTML = txt;
}</script>
</body>
```

```
</html>
```

ii.Get an Attribute Value - getAttributeNote() methods.

This method returns an attribute node.

Try this example.

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
   if (this.readyState == 4 && this.status == 200) {
      myFunction(this);
   }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
   var xmlDoc = xml.responseXML;
   var x = xmlDoc.getElementsByTagName("title")[0];
   var y = x.getAttributeNode("lang");
   var txt = y.nodeValue;
   document.getElementById("demo").innerHTML = txt;
}
</script>
</body>
</html>
```

Change Node Values

1)The nodeValue property is used to change a node value.

2)The setAttribute() method is used to change an attribute value.

Change the Value of an Element

In the DOM, everything is a node. Element nodes do not have a text value. The text value of an element node is stored in a child node. This node is called a text node. To change the text value of an element, you must change the value of the elements's text node.

Change the value of a Text Node

The **nodeValue** property can be used to change **the value of a text node**

Try this example

```
<!DOCTYPE html>

<html>

<body>

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

<script>

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() {

    if (this.readyState == 4 && this.status == 200) {

        myFunction(this);

    }

};

xhttp.open("GET", "books.xml", true);

xhttp.send();


function myFunction(xml) {

    var xmlDoc = xml.responseXML;

    var x;
```

```
    var txt = "";
    x = xmlDoc.getElementsByTagName("title")[0].childNodes[0];
    txt += x.nodeValue + "<br>";
    x.nodeValue = "Easy Cooking";
    x = xmlDoc.getElementsByTagName("title")[0].childNodes[0];
    txt += x.nodeValue + "<br>";
    document.getElementById("demo").innerHTML = txt;
}
</script>
</body>
</html>
```

Change the Value of an Attribute

This can be done using the setAttribute() method or setting the nodeValue property of the attribute node.

Change an Attribute Using setAttribute()

The **setAttribute()** method **changes the value of an attribute**.

If the attribute does not exist, a new attribute is created.

This example changes the category attribute of the <book> element

Try this example.

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
```

```
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var xmlDoc = xml.responseXML;
    var x = xmlDoc.getElementsByTagName('book');
    x[0].setAttribute("category","food");
    document.getElementById("demo").innerHTML =
    x[0].getAttribute("category");
}
</script>
</body>
</html>
```

Change an Attribute using nodeValue

The **nodeValue** property is **the value of a attribute node**.

Changing the value property changes the value of the attribute.

Try this example

```
<!DOCTYPE html>
```

```
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var xmlDoc = xml.responseXML;
    var x = xmlDoc.getElementsByTagName("book")[0]
    var y = x.getAttributeNode("category");
    var txt = y.nodeValue + "<br>";
    y.nodeValue ="food";
    txt += y.nodeValue;
    document.getElementById("demo").innerHTML = txt;
}
</script>
</body>
```

```
</html>
```

XML DOM Remove Nodes

Remove an Element Node

The **removeChild()** method removes a specified node.

When a node is removed, all its child nodes are also removed

This example remove the first <book> element from the loaded xml

Try this example

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var xmlDoc = xml.responseXML;
    var root = xmlDoc.documentElement;
```

```
    var currNode = root.childNodes[1];
    removedNode = currNode.removeChild(currNode.childNodes[1]);
    document.getElementById("demo").innerHTML =
    "Removed node: " + removedNode.nodeName;
}
</script>
</body>
</html>
```

Remove a Text Node

The **removeChild()** method can also be used to remove a text node
Try this example

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
```

```
function myFunction(xml) {
    var x, y, xmlDoc, txt;
    xmlDoc = xml.responseXML;
    txt = "";
    x = xmlDoc.getElementsByTagName("title")[0];
    txt += "Child nodes: " + x.childNodes.length +"<br>";
    y = x.childNodes[0];
    x.removeChild(y);
    txt += "Child nodes: " + x.childNodes.length;
    document.getElementById("demo").innerHTML = txt;
}
</script>
</body>
</html>
```

In this purpose we are also using nodeValue property

The **nodeValue** property can be used to change the value of a text node

Try this example

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
```

```
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var x, xmlDoc, txt;
    xmlDoc = xml.responseXML;
    x = xmlDoc.getElementsByTagName("title")[0].childNodes[0];
    txt = "Value: " + x.nodeValue + "<br>";
    x.nodeValue = "";
    txt += "Value: " + x.nodeValue;
    document.getElementById("demo").innerHTML = txt;
}
</script>
</body>
</html>
```

Remove an Attribute Node by Name

The **removeAttribute()** method removes an attribute node by its name.

This example removes the "category" attribute in the first <book> element

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var xmlDoc = xml.responseXML;
    var x = xmlDoc.getElementsByTagName("book");
    document.getElementById("demo").innerHTML =
    x[0].getAttribute('category') + "<br>";
    x[0].removeAttribute('category');
    document.getElementById("demo").innerHTML +=
    x[0].getAttribute('category');
}
</script>
</body>
```

```
</html>
```

XML DOM Replace Nodes

Replace an Element Node

The replaceChild() method is used to replace a node.

The following example replaces the first<book> element

Try this example

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var x, y, z, i, newNode, newTitle, newText, xmlDoc, txt;
    xmlDoc = xml.responseXML;
    txt = "";
```

```
    x = xmlDoc.documentElement;

    // Create a book element, title element and a text node
    newNode = xmlDoc.createElement("book");
    newTitle = xmlDoc.createElement("title");
    newText = xmlDoc.createTextNode("A Notebook");
    // Add a text node to the title node
    newTitle.appendChild(newText);
    // Add the title node to the book node
    newNode.appendChild(newTitle);
    y = xmlDoc.getElementsByTagName("book")[0];
    // Replace the first book node with the new book node
    x.replaceChild(newNode, y);
    z = xmlDoc.getElementsByTagName("title");
    // Output all titles
    for (i = 0; i < z.length; i++) {
        txt += z[i].childNodes[0].nodeValue + "<br>";
    }
    document.getElementById("demo").innerHTML = txt;
}
</script>
</body>
</html>
```

Replace Data in a Text Node.

The replaceData() method is used to replace data in a text node.

The replaceData() method has three parameters:

- offset - Where to begin replacing characters. Offset value starts at zero
- length - How many characters to replace
- string - The string to insert

try this example

```
<!DOCTYPE html>

<html>

<body>

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

<script>

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() {

    if (this.readyState == 4 && this.status == 200) {

        myFunction(this);

    }

};

xhttp.open("GET", "books.xml", true);

xhttp.send();

function myFunction(xml) {

    var xmlDoc = xml.responseXML;

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

    document.getElementById("demo").innerHTML =

    x.nodeValue;

    x.replaceData(0, 8, "Easy");
```

```
    document.getElementById("demo").innerHTML +=
    "<br>" + x.nodeValue;
}
</script>
</body>
</html>
```

And also use nodeValue Property in this purpose.

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var x, xmlDoc, txt;
    xmlDoc = xml.responseXML;
```

```
    x = xmlDoc.getElementsByTagName("title")[0].childNodes[0];
    txt = x.nodeValue + "<br>";
    x.nodeValue="Easy Italian";
    txt += x.nodeValue;
    document.getElementById("demo").innerHTML = txt
}
</script>
</body>
</html>
```

XML DOM Create Nodes

Create a New attribute Node

The **createAttribute()** is used to create a new attribute node.

Example

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
```

```
xhttp.open("GET", "books.xml", true);

xhttp.send();

function myFunction(xml) {

   var x, newatt, xmlDoc;

   xmlDoc = xml.responseXML;

   newatt = xmlDoc.createAttribute("edition");

   newatt.nodeValue = "first";

   x = xmlDoc.getElementsByTagName("title");

   x[0].setAttributeNode(newatt);

   document.getElementById("demo").innerHTML =

   "Edition: " + x[0].getAttribute("edition");

}

</script>

</body>

</html>
```

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.

```
<!DOCTYPE html>

<html>

<body>

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

<script>

var xhttp = new XMLHttpRequest();
```

```
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var xmlDoc = xml.responseXML;
    var x = xmlDoc.getElementsByTagName("title");
    x[0].setAttribute("edition", "first");
    document.getElementById("demo").innerHTML =
    "Edition: " + x[0].getAttribute("edition");
}
</script>
</body>
</html>
```

Create a Text Node

The **createTextNode()** method creates a new text node.

Example

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
```

```
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var xmlDoc = xml.responseXML;
    var x, y, i, newEle, newText, txt;
    // add an edition element
    newEle = xmlDoc.createElement("edition");
    newText = xmlDoc.createTextNode("first");
    newEle.appendChild(newText);
    x = xmlDoc.getElementsByTagName("book")[0];
    x.appendChild(newEle);
    // display all elements
    xlen = x.childNodes.length;
    y = x.firstChild;
    txt = "";
    for (i = 0; i < xlen; i++) {
        if (y.nodeType == 1) {
```

```
        txt += y.nodeName + "<br>";
      }
    y = y.nextSibling;
    }
    document.getElementById("demo").innerHTML = txt;
}
</script>
</body>
</html>
```

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.

*use insertBefore() method if the position of the node is important.

This example creates an element (<edition>), and adds it after the last child of the first <book> element

Try this example.

```
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
```

```
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var x, y, i, newElement, txt, xmlDoc;
    xmlDoc = xml.responseXML;
    newElement = xmlDoc.createElement("edition");
    x = xmlDoc.getElementsByTagName("book")[0]
    x.appendChild(newElement);
   // Display all elements
    xlen = x.childNodes.length;
    y = x.firstChild;
    txt = "";
    for (i = 0; i < xlen; i++) {
        if (y.nodeType == 1) {
            txt += y.nodeName + "<br>";
        }
        y = y.nextSibling;
    }
    document.getElementById("demo").innerHTML = txt;
}
</script>
```

```
</body>

</html>
```

The new element is added with a value.

```
newText=xmlDoc.createTextNode("first");
```

Insert a Node – insertBefore()

The **insertBefore()** method inserts a node before a specified child node.

This method is useful when the position of the added node is important:

Try this example.

```
<!DOCTYPE html>

<html>

<body>

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

<script>

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() {

    if (this.readyState == 4 && this.status == 200) {

        myFunction(this);

    }

};

xhttp.open("GET", "books.xml", true);

xhttp.send();

function myFunction(xml) {

    var xmlDoc = xml.responseXML;
```

```
    var newNode = xmlDoc.createElement("book");
    var x = xmlDoc.documentElement;
    var y = xmlDoc.getElementsByTagName("book");
    document.getElementById("demo").innerHTML =
    "Book elements before: " + y.length + "<br>";

    x.insertBefore(newNode, y[3]);
    document.getElementById("demo").innerHTML +=
    "Book elements after: " + y.length;
}
</script>
</body>
</html>
```

If the second parameter of insertBefore() is null, the new node will be added after the last existing child node.

Add a New Attribute.

The **setAttribute()** method sets the value of an attribute.

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
```

```
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var xmlDoc = xml.responseXML;
    var x = xmlDoc.getElementsByTagName("title");
    x[0].setAttribute("edition", "first");
    document.getElementById("demo").innerHTML =
    "Edition: " + x[0].getAttribute("edition");
}
</script>
</body>
</html>
```

* If the attribute already exists, the setAttribute() method will overwrite the existing value.

Add Text to a Text Node – insertData()

The insertData() method has two parameters:

- offset - Where to begin inserting characters (starts at zero)
- string - The string to insert

The following example will add "Easy" to the text node of the first <title> element of the loaded XML

Try this example

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
    }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();
function myFunction(xml) {
    var x, txt, xmlDoc;
    xmlDoc = xml.responseXML;
    x = xmlDoc.getElementsByTagName("title")[0].childNodes[0];
    txt = x.nodeValue + "<br>";
    x.insertData(0,"Easy ");
    txt += x.nodeValue;
    document.getElementById("demo").innerHTML = txt;
}
</script>
</body>
```

</html>

XML DOM Clone Nodes

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.

The following code example copies the first <book> node and appends it to the root node of the document

```
oldNode = xmlDoc.getElementsByTagName('book')[0];
newNode = oldNode.cloneNode(true);
xmlDoc.documentElement.appendChild(newNode);
```