Title: Implementation of XML Parser and XML DOM (Continues…)

1. Getting Attribute value.

Attribute nodes have text values. This can be done using the **getAttribute** method. The following code segment is an example code segment to implement the above said:

```
txt = "";
var x = xmlDoc.getElementsByTagName("book");
   for (i=0; i<x.length; i++){
               txt+= x[i].getAttribute("category") + "<br>";
   }
document.getElementById("XMLDOM").innerHTML = txt;
```

or

```
var x = xmlDoc.getElementsByTagName("title")[0];
var y = x.getAttributeNode("lang");
txt = y.nodeValue;
document.getElementById("XMLDOM").innerHTML = txt;
```

2. Changing an attribute value

```
var x = xmlDoc.getElementsByTagName("book");
x[0].setAttribute("category", "type");-->
x[0].setAttribute("category","Magical");
txt= x[0].getAttribute("category");
document.getElementById("XMLDOM").innerHTML = txt;
```

3. Changing attribute value using setAttribute method

```
var x = xmlDoc.getElementsByTagName("book");
x[0].setAttribute("category", "type");-->
x[0].setAttribute("category","Magical");
txt= x[0].getAttribute("category");
document.getElementById("XMLDOM").innerHTML = txt;
```

4. Remove an element node

The **removeChild()** method removes a specified node. When a node is removed, all its child nodes are also removed.

```
var root = xmlDoc.documentElement;
var currNode = root.childNodes[1];
var removedNode = currNode.removeChild(currNode.childNodes[1]);
document.getElementById("XMLDOM").innerHTML = "Removed Node:
"+removedNode.nodeName;
```

5. Remove a text node

```
var x = xmlDoc.getElementsByTagName("title")[0];
txt += "Child nodes: "+x.childNodes.length+"<br>";
var y = x.childNodes[0];
x.removeChild(y);
txt1 += "Child nodes: "+x.childNodes.length;
```

6. Remove an attribute node

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

```
var x = xmlDoc.getElementsByTagName("book");
document.getElementById("XMLDOM").innerHTML =
x[1].getAttribute("category")+"<br>";
x[1].removeAttribute("category");
document.getElementById("XMLDOM").innerHTML += x[1].getAttribute("category");
```

7. Replace an element.

```
var x = xmlDoc.documentElement;

//creating a book element, title element and a text node
var newNode = xmlDoc.createElement("book");
var newTitle = xmlDoc.createElement("title");
var newText = xmlDoc.createTextNode("A notebook");

//add a text node to the title node
newTitle.appendChild(newText);

//add title node to the book node
newNode.appendChild(newTitle);

var y = xmlDoc.getElementsByTagName("book")[0];
x.replaceChild(newNode, y);
var z = xmlDoc.getElementsByTagName("title");

for (i=0; i<z.length; i++){
   txt += z[i].childNodes[0].nodeValue+"<br>";
}
document.getElementById("XMLDOM").innerHTML = txt;
```

8. 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

```
var x = xmlDoc.getElementsByTagName("title")[0].childNodes[0];
document.getElementById("XMLDOM").innerHTML = x.nodeValue + "<br>";
x.replaceData(0,12,"Fantastic Beasts");

document.getElementById("XMLDOM").innerHTML += x.nodeValue;
```

9. Create a new attribute node

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

```
var newAtt = xmlDoc.createAttribute("Edition");
newAtt.nodeValue = "First";

var x = xmlDoc.getElementsByTagName("title");
x[0].setAttributeNode(newAtt);
document.getElementById("XMLDOM").innerHTML = x[0].getAttribute("Edition");
```

10. Create a text node

```
newEl = xmlDoc.createElement("Edition");
newText = xmlDoc.createTextNode("First");

newEl.appendChild(newText);
var x = xmlDoc.getElementsByTagName("book")[0];
x.appendChild(newEl);

var xlen = x.childNodes.length;
var y = x.firstChild;
txt = "";

for (i = 0; i < xlen; i++){
   if (y.nodeType == 1) {
          txt += y.nodeName + "<br>";
          }
          y = y.nextSibling;
   }
   document.getElementById("XMLDOM").innerHTML = txt;
```

11. Adding a node

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.

```
newElement = xmlDoc.createElement("edition");
var 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("XMLDOM").innerHTML = txt;
```

12. 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:

```
var newNode = xmlDoc.createElement("book");
var x = xmlDoc.documentElement;
var y = xmlDoc.getElementsByTagName("book");
document.getElementById("XMLDOM").innerHTML = "Book elements before: " + y.length
+ "<br>";
x.insertBefore(newNode, y[3]);
document.getElementById("XMLDOM").innerHTML += "Book elements after: " +
y.length;
```

13. 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

```
var x = xmlDoc.getElementsByTagName("title")[0].childNodes[0];
txt = x.nodeValue + "<br>";
x.insertData(0,"Easy ");
txt += x.nodeValue;
document.getElementById("XMLDOM").innerHTML = txt;
```

14. 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.

```
var x = getElementsByTagName("book")[0];
var clon = x.cloneNode(true);
xmlDoc.documentElement.appendChild(clon);

var y = xmlDoc.getElementsByTagName("title");

for (i = 0; i < y.length; i++) {
        txt += y[i].childNodes[0].nodeValue + "<br>";
    }
document.getElementById("XMLDOM").innerHTML = txt
```

Task:

XML:

```
<?xml version = "1.0" encoding = "UTF-8" ?>

<bookstore>
	<book category = "Entertainment">
		<title lang = "En">Harry Potter</title>
		<Author>JK. Rowling</Author>
		<Year>2009</Year>
		<Price>$45.99</Price>
	</book>

	<book category = "Fiction">
		<title lang = "En">Avengers</title>
		<Author>Warner Bros</Author>
		<Year>2010</Year>
		<Price>$32.89</Price>
	</book>

	<book category = "Mysery">
		<title lang = "Sp">The Great Gatsby</title>
		<Author>F Scott Fitzgerald</Author>
		<Year>1920</Year>
		<Price>$1.99</Price>
	</book>

	<book category = "Historical">
		<title lang = "Fr">Things Fall Apart</title>
		<Author>Chinua Achebe</Author>
		<Year>1958</Year>
		<Price>$12.99</Price>
	</book>

	<book category = "Tragedy">
		<title lang = "Es">Hamlet</title>
		<Author>William Shakespeare</Author>
		<Year>1601</Year>
		<Price>$24.99</Price>
	</book>

</bookstore>
```

Using the given XML file, execute all the above examples.