Java DOM Parser - Modify XML Document

Java DOM parser API provides methods to modify the already existing XML documents. We can add or remove elements and attributes, modify the text content of elements and attribute values using the methods of Java DOM parser.

The setTextContent() method replaces the original text of elements, removeAttribute() removes the existing attributes and the setAttribute() method sets the new attributes to elements.

Modify XML Using Java DOM Parser

We can modify an XML document in java using DOM parser through following steps.

- **Step 1:** Creating a DocumentBuilder Object
- **Step 2:** Reading the XML
- **Step 3:** Parsing the XML Document
- **Step 4:** Updating the content of XML document
- **Step 5:** Writing the content into XML file
- **Step 6:** Output to console for testing

Refer this page of this section for first three steps.

Step4: Updating the content of XML document

We can update text content, attribute values, add new elements and new attributes to our existing XML documents.

```
Element element = xmldoc.getDocumentElement();
element.setTextContent("14");
element.removeAttribute("attr_name");
element.setAttribute("attr_name","attr_value");
```

The **setTextContent("text_content")** method is used to set the text content of an Element. This method is used with the Element object and takes String as an argument.

The **removeAttribute("attr_name")** method removes the attribute from the Element object. It throws **NO_MODIFICATION_ALLOWED_ERR** error if the Element from which we want to remove the attribute is readonly.

The **setAttribute("attr_name","attr_value")** method takes attribute name and attribute value as arguments and sets it to the Element object.

Updating Text Content

Let us consider college.xml file where we have three department details. Now, we will try to update staff count for Electrical and Electronics department from 23 to 14. The method **setTextContent("text")** is used to update the text content of an element.

college.xml

Following is the college.xml file before updating. Now let us try to update this in our java program.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><college>
   <department id="101">
      <name>Computer Science</name>
      <staffCount>20</staffCount>
   </department>
   <department id="102">
      <name>Electrical and Electronics</name>
      <staffCount>23</staffCount>
   </department>
   <department id="103">
      <name>Mechanical</name>
      <staffCount>15</staffCount>
   </department>
</college>
```

ModifyXMLDemo.java

In the following java program, we retrieved all the nodes with tag name, "department" into a NodeList. Then, we iterated all the nodes to find "Electrical and Electronics" department and changed the staffCount attribute to 14.

```
import java.io.File;
import java.io.FileOutputStream;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
```

```
import javax.xml.transform.stream.StreamResult;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;

public class ModifyXMLDemo {

   public static void main(String argv[]) {

      try {

         //Creating a DocumentBuilder Object
         DocumentBuilderFactory docFactory =
DocumentBuilderFactory.newInstance();
         DocumentBuilder docBuilder = docFactory.newDocumentBuilder();

         //Reading the XML file
         File inputFile = new File("src/modifydom.xml");

         //Parsing the XML Document
         Document doc = docBuilder.parse(inputFile);

         //Updating the staffCount for "Electrical and Electronics" department
         NodeList deptList=doc.getElementsByTagName("department");
         for(int i=0;i<deptList.getLength();i++)
         {
            Element element= (Element) (deptList.item(i));
            String
s=element.getElementsByTagName("name").item(0).getTextContent();
            if(s.equals("Electrical and Electronics"))
            {
                Element staffCount = (Element)
element.getElementsByTagName("staffCount").item(0);
                staffCount.setTextContent("14");
            }
         }

         //Writing the updated content into the file
         TransformerFactory transformerFactory =
TransformerFactory.newInstance();
         Transformer transformer = transformerFactory.newTransformer();
         DOMSource source = new DOMSource(doc);
```

```
            FileOutputStream output = new FileOutputStream("college.xml");
            StreamResult result = new StreamResult(output);
            transformer.transform(source, result);

            //writing the content on console
            StreamResult consoleResult = new StreamResult(System.out);
            transformer.transform(source, consoleResult);
         } catch (Exception e) { e.printStackTrace();}
      }
   }
```

Output

Following is the updated XML document after updating staffCount.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><college>
   <department id="101">
      <name>Computer Science</name>
      <staffCount>20</staffCount>
   </department>
   <department id="102">
      <name>Electrical and Electronics</name>
      <staffCount>14</staffCount>
   </department>
   <department id="103">
      <name>Mechanical</name>
      <staffCount>15</staffCount>
   </department>
</college>
```


Adding New Elements

Now, let us go a bit further and try to add one more department named "Civil" to our above "college.xml" file. To add new elements, we can use **createElement("Element_name")** method to create and **appendChild(Element)** to append the element to the existing document's root element.

ModifyXMLAddElementsDemo.java

In the following program, we first got the root element by using getDocumentElement() method. Then, we created "Civil" department element and added to the root element.

```
import java.io.File;
import java.io.FileOutputStream;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import org.w3c.dom.Document;
import org.w3c.dom.Element;

public class ModifyXMLAddElementsDemo {

   public static void main(String argv[]) {

      try {

         //Creating a DocumentBuilder Object
         DocumentBuilderFactory docFactory =
DocumentBuilderFactory.newInstance();
         DocumentBuilder docBuilder = docFactory.newDocumentBuilder();

         //Reading the XML file
         File inputFile = new File("college.xml");

         //Parsing the XML Document
         Document doc = docBuilder.parse(inputFile);

         //Adding new department element
         Element rootElement = doc.getDocumentElement();
         Element department = doc.createElement("department");
         department.setAttribute("id", "104");
         Element name = doc.createElement("name");
         Element staffCount = doc.createElement("staffCount");
         department.appendChild(name);
         department.appendChild(staffCount);
         name.appendChild(doc.createTextNode("Civil"));
         staffCount.appendChild(doc.createTextNode("10"));
         rootElement.appendChild(department);
```

```
         //Creating transformer object
         TransformerFactory transformerFactory =
TransformerFactory.newInstance();
         Transformer transformer = transformerFactory.newTransformer();

         //Writing updated content into the file
         DOMSource source = new DOMSource(doc);
         FileOutputStream output = new FileOutputStream("college.xml");
         StreamResult result = new StreamResult(output);
         transformer.transform(source, result);

         //writing the content on console
         StreamResult consoleResult = new StreamResult(System.out);
         transformer.transform(source, consoleResult);
      } catch (Exception e) { e.printStackTrace();}
   }
}
```

Output

The updated file after adding "Civil" department is as follows :

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><college>
   <department id="101">
      <name>Computer Science</name>
      <staffCount>20</staffCount>
   </department>
   <department id="102">
      <name>Electrical and Electronics</name>
      <staffCount>14</staffCount>
   </department>
   <department id="103">
      <name>Mechanical</name>
      <staffCount>15</staffCount>
   </department>
<department id="104"><name>Civil</name><staffCount>10</staffCount></department></
```