Java SAX Parser - Modify XML Document

Java SAX parser is a Java API that can be used to parse and modify XML documents. To modify any XML document using SAX parser, We have to store the data in any data structure because we cannot go back as it has only forward read-only access. In this chapter, we are going to see how to read and modify existing XML document in detail.

Modify XML Using Java SAX Parser

- **Step 1:** Implementing a Handler class
- **Step 2:** Creating a SAXParser Object
- **Step 3:** Reading the XML
- **Step 4:** Creating object for Handler class
- **Step 5:** Parsing the XML Document
- **Step 6:** Writing the updated content into file

Refer Parse XML Document chapter of this section for the first five steps.

Step 6: Writing the Updated Content into File

The updated content can be written into the file using write() method of FileWriter object as follows –

```
FileWriter filewriter = new FileWriter("newfile.xml");
filewriter.write("content_here");
```

Modifying XML File

SAX parser will not provide random access to elements as DOM provides. We can implement the call back methods in Handler class that implements DefaultHandler to create and add new elements to the already exisiting XML document.

Example

In this example, we are going to see how to modify studentData.xml file by adding result element to the student element. Here is the **sudentData.xml** file that we need to modify by appending <Result>Pass<Result/> at the end of </marks> tag.

```
<?xml version = "1.0"?>
<class>
   <student rollno = "393">
      <firstname>dinkar</firstname>
      <lastname>kad</lastname>
      <nickname>dinkar</nickname>
      <marks>85</marks>
   </student>

   <student rollno = "493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>vinni</nickname>
      <marks>95</marks>
   </student>

   <student rollno = "593">
      <firstname>jasvir</firstname>
      <lastname>singn</lastname>
      <nickname>jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

In the following **ModifyXML.java** program, we have created SAXParser object and parsed the input file. We have implemented the Handler class with the necessary call back methods and writeFile() method to write the updated content into the file. We have used an array of String to store and write the content to the file.

```
import java.io.*;
import org.xml.sax.*;
import javax.xml.parsers.*;
import org.xml.sax.helpers.DefaultHandler;

//Implementing a Handler class
class UserDemoHandler extends DefaultHandler {

   static String displayText[] = new String[1000];
   static int numberLines = 0;
   static String indentation = "";
```

```
   public void startDocument() {
      displayText[numberLines] = indentation;
      displayText[numberLines] += "<?xml version = \"1.0\" encoding = \""+
            "UTF-8" + "\"?>";
      numberLines++;
   }

   public void processingInstruction(String target, String data) {
      displayText[numberLines] = indentation;
      displayText[numberLines] += "<?";
      displayText[numberLines] += target;

      if (data != null && data.length() > 0) {
         displayText[numberLines] += ' ';
         displayText[numberLines] += data;
      }
       displayText[numberLines] += "?>";
       numberLines++;
   }

   public void startElement(String uri, String localName, String
qualifiedName,
           Attributes attributes) {
      displayText[numberLines] = indentation;
      indentation += "    ";
      displayText[numberLines] += '<';
      displayText[numberLines] += qualifiedName;

      if (attributes != null) {

         int numberAttributes = attributes.getLength();
         for (int loopIndex = 0; loopIndex < numberAttributes; loopIndex++) {
            displayText[numberLines] += ' ';
            displayText[numberLines] += attributes.getQName(loopIndex);
            displayText[numberLines] += "=\"";
            displayText[numberLines] += attributes.getValue(loopIndex);
            displayText[numberLines] += '"';
          }
       }
          displayText[numberLines] += '>';
          numberLines++;
   }
```

```
   public void characters(char characters[], int start, int length) {
      String characterData = (new String(characters, start, length)).trim();
      if(characterData.indexOf("\n") < 0 && characterData.length() > 0) {
         displayText[numberLines] = indentation;
         displayText[numberLines] += characterData;
         numberLines++;
      }
   }

   public void endElement(String uri, String localName, String qualifiedName)
{

      indentation = indentation.substring(0, indentation.length() - 4) ;
      displayText[numberLines] = indentation;
      displayText[numberLines] += "</";
      displayText[numberLines] += qualifiedName;
      displayText[numberLines] += '>';
      numberLines++;
      if (qualifiedName.equals("marks")) {
         startElement("", "Result", "Result", null);
         characters("Pass".toCharArray(), 0, "Pass".length());
         endElement("", "Result", "Result");
      }
   }

   public void writeFile(FileWriter filewriter) throws IOException {

      for(int loopIndex = 0; loopIndex < numberLines; loopIndex++) {
         filewriter.write(displayText[loopIndex].toCharArray());
         filewriter.write('\n');
         System.out.println(displayText[loopIndex].toString());
      }
      filewriter.close();
   }
}

public class ModifyXML extends DefaultHandler {
   public static void main(String args[]) {
      try {

         //Creating SAXParser object
```

```
            SAXParserFactory factory = SAXParserFactory.newInstance();
            SAXParser saxParser = factory.newSAXParser();

            //Reading the XML
            File inputFile = new File("src/input.txt");

            //Creating object for Handler class
            UserDemoHandler userHandler = new UserDemoHandler();

            //Parsing the XML
            saxParser.parse(inputFile, userHandler);

            //Writing the updated content into file
            FileWriter filewriter = new FileWriter("src/newfile.xml");
            userHandler.writeFile(filewriter);

         }
         catch (Exception e) {
            e.printStackTrace(System.err);
         }
      }
   }
```

The output window displays the updated content of the XML document after adding result element to each student element.

```
<?xml version = "1.0" encoding = "UTF-8"?>
<class>
  <student rollno = "393">
    <firstname>
      dinkar
    </firstname>
    <lastname>
      kad
    </lastname>
    <nickname>
      dinkar
    </nickname>
    <marks>
      85
    </marks>
    <Result>
```

```
      Pass
    </Result>
  </student>
  <student rollno = "493">
    <firstname>
      Vaneet
    </firstname>
    <lastname>
      Gupta
    </lastname>
    <nickname>
      vinni
    </nickname>
    <marks>
      95
    </marks>
    <Result>
      Pass
    </Result>
  </student>
  <student rollno = "593">
    <firstname>
      jasvir
    </firstname>
    <lastname>
      singn
    </lastname>
    <nickname>
      jazz
    </nickname>
    <marks>
      90
    </marks>
    <Result>
      Pass
    </Result>
  </student>
</class>
```