Java StAX Parser - Create XML Document

Java StAX parser is a Java API that provides interfaces to create XML documents. There are cursor APIs such as XMLStreamWriter and Iterator APIs such as XMLEventWriter to create XML documents. In this chapter, we are going to use XMLStreamWriter to create XML documents. This interface has methods such as writeStartDocument(), writeStartElement(), writeCharacters() etc., to write XML content.

Create XML using Java StAX parser

We can create an XML document in Java using StAX parser through following steps –

- **Step 1:** Creating XMLStreamWriter object
- **Step 2:** Writing the XML document
- **Step 3:** Creating XMLStreamReader object
- **Step 4:** Writing the content into file
- **Step 5:** Testing the output using console

Step 1: Creating XMLStreamWriter object

The XMLOutputFactory is an abstract class which is used to create XMLEventWriters and XMLStreamWriters. We can either use XMLEventWriter or XMLStreamWriter objects to write XML documents.

The createXMLStreamWriter() method of XMLOutputFactory creates an XMLStreamWriter object and the argument can be a Writer, OutputStream or any JAXP result. Here, we have created an XMLStreamWriter that writes to a Writer stream.

```
XMLOutputFactory xMLOutputFactory = XMLOutputFactory.newInstance();
XMLStreamWriter xMLStreamWriter =
xMLOutputFactory.createXMLStreamWriter(Writer stream);
```

Step 2: Writing the XML document

The XMLStreamWriter interface has many methods that specify how to write an XML document. They are self explanatory and does not return anything, but writes the content to the Writer. Following are the most commonly used methods of XMLStreamWriter interface –

```
xMLStreamWriter.writeStartDocument();
xMLStreamWriter.writeStartElement("cars");
xMLStreamWriter.writeCharacters("Ferrari");
xMLStreamWriter.writeEndElement();
xMLStreamWriter.writeEndDocument();
```

Step 3: Creating XMLStreamReader object

The XMLInputFactory is an abstract class for getting streams. Using createXMLStreamReader() method, we can create an XMLStreamReader object that allows us to read the InputStream specified. It gives forward, read-only access to the InputStream of XML data.

```
XMLInputFactory factory = XMLInputFactory.newInstance();
XMLStreamReader streamReader = factory.createXMLStreamReader(InputStream
stream);
```

Step 4: Writing the content into file

The transform() function of Transformer class tranforms the XML source to result. Source can be of type DOMSource, SAXSource, StAXSource or StreamSource and result can be of type DOMResult, SAXResult, StAXResult or StreamResult.

To create Transformer object, we create an instance of TransformerFactory. Here, source is of type StAXSource and result is of type StreamResult. We can create a new File object and pass this as an argument to StreamResult object.

```
TransformerFactory transformerFactory = TransformerFactory.newInstance();
Transformer transformer = transformerFactory.newTransformer();
transformer.transform(source, result);
```

Step 5: Testing the output using console

We can print the file content on the console by passing 'System.out' as an argument for StreamResult. The transform() method of Transformer class converts the source into StreamResult and writes it on the console.

Another way of printing the result is by simply converting the Writer object we have used for XMLStreamWriter into a String and printing that string on the console.

```
StreamResult consoleResult = new StreamResult(System.out);
transformer.transform(source, consoleResult);
```

Creating Basic XML file

The **writeStartDocument()** method writes the XML declaration with default version number as 1.0 and encoding as utf-8.

The **writeStartElement("element_name")** method opens a new start tag with the specified argument name as the name of the element. The scope gets closed once we specify corresponding EndElement.

The **writeCharacters("text_content")** method appends the specified text content to the Element.

Example

Here is the basic XML file we want to create –

```
<?xml version="1.0" ?>
<cars>Ferrari</cars>
```

The **CreateBasicXML.java** program creates cars.xml file using XMLStreamWriter class. We have used the methods writeStartDocument() to write the XML declaraction, writeStartElement("cars") to write the new Element named, "cars" and writeCharactrs("Ferrari") to write the text content. We have used writeEndElement() and writeEndDocument() methods to close the scopes correctly.

```
import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.StringWriter;
import javax.xml.stream.XMLInputFactory;
import javax.xml.stream.XMLOutputFactory;
import javax.xml.stream.XMLStreamReader;
import javax.xml.stream.XMLStreamWriter;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.stax.StAXSource;
import javax.xml.transform.stream.StreamResult;

public class CreateBasicXML {
```

```
   public static void main(String[] args) throws TransformerException {
      try {

         //creating XMLStreamWriter
         XMLOutputFactory xMLOutputFactory = XMLOutputFactory.newInstance();
         StringWriter stringWriter = new StringWriter();
         XMLStreamWriter xMLStreamWriter =
            xMLOutputFactory.createXMLStreamWriter(stringWriter);

         //Writing the XML document
         xMLStreamWriter.writeStartDocument();
         xMLStreamWriter.writeStartElement("cars");
         xMLStreamWriter.writeCharacters("Ferrari");
         xMLStreamWriter.writeEndElement();
         xMLStreamWriter.writeEndDocument();
         xMLStreamWriter.flush();
         xMLStreamWriter.close();

         //Creating XMLStreamReader object
         String xmlString = stringWriter.getBuffer().toString();
         ByteArrayInputStream input = new
ByteArrayInputStream(xmlString.getBytes("UTF-8"));
         stringWriter.close();
         XMLInputFactory factory = XMLInputFactory.newInstance();
         XMLStreamReader streamReader =
                 factory.createXMLStreamReader(input);

         //writing the content into XML file
         TransformerFactory transformerFactory =
TransformerFactory.newInstance();
         Transformer transformer = transformerFactory.newTransformer();
         StAXSource source = new StAXSource(streamReader);
         StreamResult result = new StreamResult(new File("D:\\cars.xml"));
         transformer.transform(source, result);

         //Testing the output using console
         System.out.println(xmlString);

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

```
    }
}
```

Output

The XML document we have written using XMLStreamWriter is displayed as follows –

```
<?xml version="1.0" ?><cars>Ferrari</cars>
```

Learn **Java** in-depth with real-world projects through our **Java certification course**. Enroll and become a certified expert to boost your career.

Creating XML file with Attributes

The **writeAttribute("attr_name","attr_value")** method appends the attribute to the current element. It should be written between writeStartElement() and witeEndElement() methods. If this method is placed incorrectly out of scope of an element, it throws IllegalStateException error.

Example

We have to create the following cars.xml file –

```
<cars>
<supercars company="Ferrari">
<carname type="formula one">Ferrari 101</carname>
<carname type="sports">Ferrari 202</carname>
</supercars>
</cars>
```

In the following **CreateAttributes.java** program, we have created two carname elements under supercar element along with attributes.

```
import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.StringWriter;
import javax.xml.stream.XMLInputFactory;
import javax.xml.stream.XMLOutputFactory;
import javax.xml.stream.XMLStreamReader;
import javax.xml.stream.XMLStreamWriter;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerException;
```

```
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.stream.StreamResult;
import javax.xml.transform.stax.StAXSource;

public class CreateAttributes {
   public static void main(String[] args) throws TransformerException {
      try {

         //creating XMLStreamWriter
         XMLOutputFactory xMLOutputFactory = XMLOutputFactory.newInstance();
         StringWriter stringWriter = new StringWriter();
         XMLStreamWriter xMLStreamWriter =
            xMLOutputFactory.createXMLStreamWriter(stringWriter);

         //Writing the XML document
         xMLStreamWriter.writeStartDocument();
         xMLStreamWriter.writeStartElement("cars");

         xMLStreamWriter.writeStartElement("supercars");
         xMLStreamWriter.writeAttribute("company", "Ferrari");

         xMLStreamWriter.writeStartElement("carname");
         xMLStreamWriter.writeAttribute("type", "formula one");
         xMLStreamWriter.writeCharacters("Ferrari 101");
         xMLStreamWriter.writeEndElement();

         xMLStreamWriter.writeStartElement("carname");
         xMLStreamWriter.writeAttribute("type", "sports");
         xMLStreamWriter.writeCharacters("Ferrari 202");
         xMLStreamWriter.writeEndElement();

         xMLStreamWriter.writeEndElement();
         xMLStreamWriter.writeEndDocument();

         xMLStreamWriter.flush();
         xMLStreamWriter.close();

         //Creating XMLStreamReader object
         String xmlString = stringWriter.getBuffer().toString();
         ByteArrayInputStream input = new
ByteArrayInputStream(xmlString.getBytes("UTF-8"));
         stringWriter.close();
```

```
         XMLInputFactory factory = XMLInputFactory.newInstance();
         XMLStreamReader streamReader =
                 factory.createXMLStreamReader(input);

         //writing the content into XML file
         TransformerFactory transformerFactory =
TransformerFactory.newInstance();
         Transformer transformer = transformerFactory.newTransformer();
         StAXSource source = new StAXSource(streamReader);
         StreamResult result = new StreamResult(new File("D:\\cars.xml"));
         transformer.transform(source, result);

         //Testing the output using console
         System.out.println(xmlString);

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

Output

The output window displays the XML content generated using XMLStreamWriter.

```
<?xml version = "1.0" encoding = "UTF-8" standalone = "no"?>
<cars>
  <supercars company = "Ferrari">
    <carname type = "formula one">Ferrari 101</carname>
    <carname type = "sports">Ferrari 202</carname>
  </supercars>
</cars>
```