

Curso de Especialização Tecnológica

Tecnologia e Programação de Sistemas de Informação – 3ª Edição

Componente de formação: Integração de SI – Ferramentas

Introdução – Java XML

Stax (Streaming API for XML) é uma API para leitura e escrita de documentos XML.

O JAXB define uma API para o programador para leitura e escrita de objetos de documentos XML.

O Stax tem duas categorias:

- Cursor API
- Event Iterator API – Tem duas interfaces principais: *XML EventReader*, para analisar XML e o *XMLEventWriter* para gerar XML.

As aplicações podem usar qualquer uma destas para análise de documentos XML.

1. XMLEventReader - Read XML Example

O Exemplo seguinte é sobre aplicações que fazem loop a todo o documento solicitando o próximo evento. O Event Iterator é implementado sobre o Cursor API.

No seguinte exemplo o documento XML é lido e posteriormente são criados os objetos.

```
<?xml version="1.0" encoding="UTF-8"?>
<config>
  <item date="January 2009">
    <mode>1</mode>
    <unit>900</unit>
    <current>1</current>
    <interactive>1</interactive>
  </item>
  <item date="February 2009">
    <mode>2</mode>
    <unit>400</unit>
    <current>2</current>
    <interactive>5</interactive>
```

```
</item>
<item date="December 2009">
  <mode>9</mode>
  <unit>5</unit>
  <current>100</current>
  <interactive>3</interactive>
</item>
</config>
```

Curso de Especialização Tecnológica

Tecnologia e Programação de Sistemas de Informação – 3ª Edição

Criando a classe para armazenar os registos do ficheiro XML.

```
package de.vogella.xml.stax.model;

public class Item {
    private String date;
    private String mode;
    private String unit;
    private String current;
    private String interactive;

    public String getDate() {
        return date;
    }

    public void setDate(String date) {
        this.date = date;
    }

    public String getMode() {
        return mode;
    }

    public void setMode(String mode) {
        this.mode = mode;
    }

    public String getUnit() {
        return unit;
    }

    public void setUnit(String unit) {
        this.unit = unit;
    }

    public String getCurrent() {
```

```

        return current;
    }

    public void setCurrent(String current) {
        this.current = current;
    }

    public String getInteractive() {
        return interactive;
    }

    public void setInteractive(String interactive) {
        this.interactive = interactive;
    }

    @Override
    public String toString() {
        return "Item [current=" + current + ", date=" + date + ", interactive="
            + interactive + ", mode=" + mode + ", unit=" + unit + "];"
    }
}

```

O seguinte código lê o ficheiro XML e cria uma lista de objetos.

```

package de.vogella.xml.stax.read;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.InputStream;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

import javax.xml.stream.XMLEventReader;
import javax.xml.stream.XMLInputFactory;
import javax.xml.stream.XMLStreamException;
import javax.xml.stream.events.Attribute;

```

Curso de Especialização Tecnológica

Tecnologia e Programação de Sistemas de Informação – 3ª Edição

```
import javax.xml.stream.events.EndElement;
import javax.xml.stream.events.StartElement;
import javax.xml.stream.events.XMLEvent;

import de.vogella.xml.stax.model.Item;

public class StaxParser {
    static final String DATE = "date";
    static final String ITEM = "item";
    static final String MODE = "mode";
    static final String UNIT = "unit";
    static final String CURRENT = "current";
    static final String INTERACTIVE = "interactive";

    @SuppressWarnings({ "unchecked", "null" })
    public List<Item> readConfig(String configFile) {
        List<Item> items = new ArrayList<Item>();
        try {
            // First, create a new XMLInputFactory
            XMLInputFactory inputFactory = XMLInputFactory.newInstance();
            // Setup a new eventReader
            InputStream in = new FileInputStream(configFile);
            XMLEventReader eventReader = inputFactory.createXMLEventReader(in);
            // read the XML document
            Item item = null;

            while (eventReader.hasNext()) {
                XMLEvent event = eventReader.nextEvent();

                if (event.isStartElement()) {
                    StartElement startElement = event.asStartElement();
                    // If we have an item element, we create a new item
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return items;
    }
}
```

```

if (startElement.getName().getLocalPart() == (ITEM)) {
    item = new Item();
    // we read the attributes from this tag and add the date
    // attribute to our object
    Iterator<Attribute> attributes = startElement
        .getAttributes();
    while (attributes.hasNext()) {
        Attribute attribute = attributes.next();
        if (attribute.getName().toString().equals(DATE)) {
            item.setDate(attribute.getValue());
        }
    }
}

if (event.isStartElement()) {
    if (event.asStartElement().getName().getLocalPart()
        .equals(MODE)) {
        event = eventReader.nextEvent();
        item.setMode(event.asCharacters().getData());
        continue;
    }
}

if (event.asStartElement().getName().getLocalPart()
    .equals(UNIT)) {
    event = eventReader.nextEvent();
    item.setUnit(event.asCharacters().getData());
    continue;
}

if (event.asStartElement().getName().getLocalPart()
    .equals(CURRENT)) {
    event = eventReader.nextEvent();
    item.setCurrent(event.asCharacters().getData());
    continue;
}

```

Curso de Especialização Tecnológica

Tecnologia e Programação de Sistemas de Informação – 3ª Edição

```
    if (event.asStartElement().getName().getLocalPart()
        .equals(INTERACTIVE)) {
        event = eventReader.nextEvent();
        item.setInteractive(event.asCharacters().getData());
        continue;
    }
}

// If we reach the end of an item element, we add it to the list
if (event.isEndElement()) {
    EndElement endElement = event.asEndElement();
    if (endElement.getName().getLocalPart() == (ITEM)) {
        items.add(item);
    }
}

}

} catch (FileNotFoundException e) {
    e.printStackTrace();
} catch (XMLStreamException e) {
    e.printStackTrace();
}
return items;
}
}
```

Com o seguinte código testa o programa. Não esquecer que o ficheiro config.xml deve existir na pasta **project**.

```
package de.vogella.xml.stax.read;
```

```

import java.util.List;

import de.vogella.xml.stax.model.Item;

public class TestRead {
    public static void main(String args[]) {
        StaXParser read = new StaXParser();
        List<Item> readConfig = read.readConfig("config.xml");
        for (Item item : readConfig) {
            System.out.println(item);
        }
    }
}

```

2. Write XML File Example

Assuma que pretende escrever o seguinte ficheiro XML.

```

<?xml version="1.0" encoding="UTF-8"?>
<config>
  <mode>1</mode>
  <unit>900</unit>
  <current>1</current>
  <interactive>1</interactive>
</config>

```

StaX não providencia funcionalidade para formatar automaticamente o ficheiro XML, para tal, é necessário adicionar *end-of-lines* e *tab informations* ao ficheiro XML.

```

package de.vogella.xml.stax.writer;

import java.io.FileOutputStream;

import javax.xml.stream.XMLEventFactory;
import javax.xml.stream.XMLEventWriter;
import javax.xml.stream.XMLOutputFactory;

```


Curso de Especialização Tecnológica

Tecnologia e Programação de Sistemas de Informação – 3ª Edição

```
import javax.xml.stream.XMLStreamException;
import javax.xml.stream.events.Characters;
import javax.xml.stream.events.EndElement;
import javax.xml.stream.events.StartDocument;
import javax.xml.stream.events.StartElement;
import javax.xml.stream.events.XMLEvent;

public class StaxWriter {
    private String configFile;

    public void setFile(String configFile) {
        this.configFile = configFile;
    }

    public void saveConfig() throws Exception {
        // create an XMLOutputFactory
        XMLOutputFactory outputFactory = XMLOutputFactory.newInstance();
        // create XMLEventWriter
        XMLEventWriter eventWriter = outputFactory
            .createXMLEventWriter(new FileOutputStream(configFile));
        // create an EventFactory
        XMLEventFactory eventFactory = XMLEventFactory.newInstance();
        XMLEvent end = eventFactory.createDTD("\n");
        // create and write Start Tag
        StartDocument startDocument = eventFactory.createStartDocument();
        eventWriter.add(startDocument);

        // create config open tag
        StartElement configStartElement = eventFactory.createStartElement("",
            "", "config");
        eventWriter.add(configStartElement);
        eventWriter.add(end);
    }
}
```

```

    // write the different nodes
    createNode(eventWriter, "mode", "1");
    createNode(eventWriter, "unit", "901");
    createNode(eventWriter, "current", "0");
    createNode(eventWriter, "interactive", "0");

    eventWriter.add(eventFactory.createEndElement("", "", "config"));
    eventWriter.add(end);
    eventWriter.add(eventFactory.createEndDocument());
    eventWriter.close();
}

private void createNode(XMLEventWriter eventWriter, String name,
    String value) throws XMLStreamException {

    XMLEventFactory eventFactory = XMLEventFactory.newInstance();
    XMLEvent end = eventFactory.createDTD("\n");
    XMLEvent tab = eventFactory.createDTD("\t");
    // create Start node
    StartElement sElement = eventFactory.createStartElement("", "", name);
    eventWriter.add(tab);
    eventWriter.add(sElement);
    // create Content
    Characters characters = eventFactory.createCharacters(value);
    eventWriter.add(characters);
    // create End node
    EndElement eElement = eventFactory.createEndElement("", "", name);
    eventWriter.add(eElement);
    eventWriter.add(end);

}
}

```

De seguida um pequeno teste:

Curso de Especialização Tecnológica

Tecnologia e Programação de Sistemas de Informação – 3ª Edição

```
package de.vogella.xml.stax.writer;

public class Testwrite {

    public static void main(String[] args) {
        Staxwriter configFile = new Staxwriter();
        configFile.setFile("config2.xml");
        try {
            configFile.saveConfig();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

XPATH

É uma linguagem para procurar/selecionar nós de um documento XML. O Java 5 introduziu o `java.xml.xpath` que fornece a biblioteca XPath.

O seguinte exemplo explica como utilizar o XPath para manipular um documento XML via Java.

Criando o seguinte ficheiro XML, temos:

```
<?xml version="1.0" encoding="UTF-8"?>
<people>
  <person>
    <firstname>Lars</firstname>
    <lastname>Vogel</lastname>
    <city>Heidelberg</city>
  </person>
  <person>
    <firstname>Jim</firstname>
```

```
<lastname>Knopf</lastname>
<city>Heidelberg</city>
</person>
<person>
  <firstname>Lars</firstname>
  <lastname>Strangelastname</lastname>
  <city>London</city>
</person>
<person>
  <firstname>Landerman</firstname>
  <lastname>Petrelli</lastname>
  <city>Somewhere</city>
</person>
<person>
  <firstname>Lars</firstname>
  <lastname>Tim</lastname>
  <city>SomewhereElse</city>
</person>
</people>
```

Crie uma nova package “myxml” e uma nova class “QueryXML”.

```
package myxml;

import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.xpath.XPath;
import javax.xml.xpath.XPathConstants;
import javax.xml.xpath.XPathExpression;
import javax.xml.xpath.XPathExpressionException;
import javax.xml.xpath.XPathFactory;
```

Curso de Especialização Tecnológica

Tecnologia e Programação de Sistemas de Informação – 3ª Edição

```
import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class QueryXML {
    public void query() throws ParserConfigurationException, SAXException,
        IOException, XPathExpressionException {
        // standard for reading an XML file
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        factory.setNamespaceAware(true);
        DocumentBuilder builder;
        Document doc = null;
        XPathExpression expr = null;
        builder = factory.newDocumentBuilder();
        doc = builder.parse("person.xml");

        // create an XPathFactory
        XPathFactory xFactory = XPathFactory.newInstance();

        // create an XPath object
        XPath xpath = xFactory.newXPath();

        // compile the XPath expression
        expr = xpath.compile("//person[firstname='Lars']/lastname/text()");
        // run the query and get a nodeset
        Object result = expr.evaluate(doc, XPathConstants.NODESET);

        // cast the result to a DOM NodeList
        NodeList nodes = (NodeList) result;
        for (int i=0; i<nodes.getLength();i++){
            System.out.println(nodes.item(i).getNodeValue());
        }
    }
}
```

```

// new XPath expression to get the number of people with name Lars
expr = xpath.compile("count(//person[firstname='Lars'])");
// run the query and get the number of nodes
Double number = (Double) expr.evaluate(doc, XPathConstants.NUMBER);
System.out.println("Number of objects " +number);

// do we have more than 2 people with name Lars?
expr = xpath.compile("count(//person[firstname='Lars']) >2");
// run the query and get the number of nodes
Boolean check = (Boolean) expr.evaluate(doc, XPathConstants.BOOLEAN);
System.out.println(check);
}

public static void main(String[] args) throws XPathExpressionException,
ParserConfigurationException, SAXException, IOException {
    QueryXML process = new QueryXML();
    process.query();
}
}

```

Bom trabalho!