Linguagem de Definição de Dados

Análise e Desenvolvimento de Sistemas

Paulo Maurício Gonçalves Júnior

Instituto Federal de Educação, Ciência e Tecnologia de Pernambuco

23 de agosto de 2023

Parte VIII

SAX – Simple API for XML

Introdução

- SAX é um mecanismo para para acessar documentos XML de forma serial, utilizando um modelo de eventos.
- Toda vez que um nó XML é encontrado, o parser XML chama um método de um objeto.
- Vantagem
  - Precisa de pouca memória para executar, pois o documento é analisado sequencialmente.
- Desvantagens
  - Não pode revisitar conteúdo.
  - Validação: Como não lê todo o documento de uma vez, se o documento possuir um erro de sintaxe, SAX só o reportará quando o encontrar.

Eventos

- SAX funciona executando um evento toda vez que que um conteúdo for encontrado.

Nome do evento	**Descrição**
startDocument	Início do processamento
endDocument	Fim do processamento
startElement	Uma tag de abertura foi encontrada
endElement	Uma tag de fechamento foi encontrada
characters	Um nó de texto foi encontrado

Tabela: Eventos principais

Métodos I

- Esses eventos estão associados a métodos.
- Inicialmente, criamos uma classe que herda de `DefaultHandler`.
- Essa classe implementa todos os eventos a serem executados quando os eventos forem lançados.
- Dessa forma, implementaremos os métodos que serão chamados pelo parser toda vez que um conteúdo for encontrado.
- Apenas os métodos associados aos eventos desejados precisam ser implementados.

```
public class UserHandler extends DefaultHandler {
  public void startDocument() throws SAXException;
  public void endDocument() throws SAXException;
  public void startElement(String namespaceURI, String localName,
    String qName, Attributes atts) throws SAXException;
  public void endElement(String namespaceURI, String localName, String
    qName) throws SAXException;
  public void characters(char ch[], int start, int length) throws
    SAXException;
}
```

Métodos II

- O método `startElement` recebe quatro parâmetros:
  1. O espaço de nomes do elemento. Só aparece se tiver um espaço de nomes associado.
  2. O nome local: nome do elemento. Só aparece se não tiver um espaço de nomes associado.
  3. O nome qualificado: nome do elemento com o prefixo. Só aparece se tiver um espaço de nomes associado.
  4. Coleção de atributos.
- O método `endElement` recebe os três primeiros parâmetros.
- O método `characters` recebe um array de caracteres, o índice do primeiro caracter e a quantidade de caracteres. Podemos obter uma String em Java passando esses valores para um dos construtores:

```
String s = new String(ch, start, length);
```

Parser I

- Depois da criação da classe que tratará os eventos recebidos, criaremos o parser XML, informando o arquivo XML que deverá ser processado e passaremos a ele um objeto da classe criada previamente.
- Considerando o seguinte arquivo e a classe que implementa um `DefaultHandler`:

```
File inputFile = new File("input.xml");
UserHandler userhandler = new UserHandler();
```

- Podemos criar o parser de duas formas:

```
try {
  SAXParserFactory factory = SAXParserFactory.newInstance();
  SAXParser saxParser = factory.newSAXParser();
  saxParser.parse(inputFile, userhandler);
} catch (Exception e) {
  e.printStackTrace();
}
```

Parser II

- ou

```
try {
  XMLReader reader = XMLReaderFactory.createXMLReader();
  reader.setContentHandler(userHandler);
  reader.parse(new InputSource(new FileReader(inputFile)));
} catch (Exception e) {
  e.printStackTrace();
}
```