Seminar: XML und Datenbanken

XML-Verarbeitungsmodelle
und
Language Bindings

Christian Müller

24.01.2003

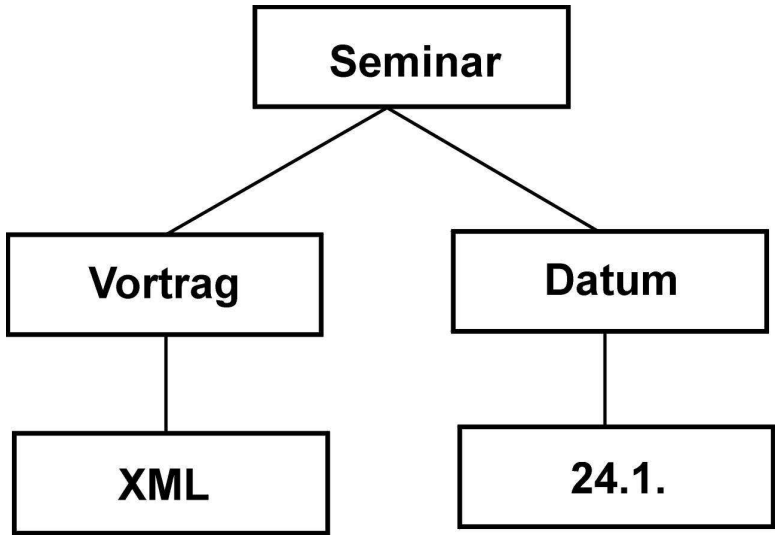
Übersicht

Einleitung

Möglichkeiten zur Verarbeitung von XML

- ereignisbasiert
- Baumstruktur

```
<Seminar>
  <Vortrag>XML</Vortrag>
  <Datum>24.1.</Datum>
</Seminar>
```



SAX

- Java-API
- kein Parser
- ereignisbasiert
- beim Parsen werden Ereignisse ausgelöst (Callbacks)
- Anwendungscode in diese Methoden einfügen

Ereignisse

`...`	startDocument
`<vortragender>`	startElement
`  <name>`	startElement
`  Müller`	characters
`  </name>`	endElement
`  <vorname>`	startElement
`  Christian`	characters
`  </vorname>`	endElement
`</vortragender>`	endElement
`...`	endDocument

XMLReader

```
XMLReader reader = XMLReaderFactory.
 createXMLReader(“org.apache.xerces.parsers.SAXParser“);
reader.setContentHandler(new MyContentHandler());
reader.setErrorHandler(new MyErrorHandler());
reader.setFeature(“http://xml.org/features/validation“,
                  true);
reader.parse(new InputSource(“beispiel.xml“));
```

ContentHandler

startDocument() :

- zu Beginn des Dokuments aufgerufen

endDocument() :

- zu Ende des Dokuments aufgerufen

setDocumentLocator(Locator locator) :

- zu Beginn des Dokuments aufgerufen
- Locator stellt Informationen bereit

ContentHandler (2)

startElement(String namespaceURI, String localName,
String qName, Attributes atts) :

- namespaceURI: Namensraumaddresse des Elements
- localName: Name des Elements ohne Namensraumadresse
- qName: qualifizierter Name
- atts: Attribute des Elements

ContentHandler (3)

- endElement(String namespaceURI, String localName,
  String qName)

- characters(char[] ch, int start, int length) :
  - beim Auftreten von Text aufgerufen

- weitere Methoden

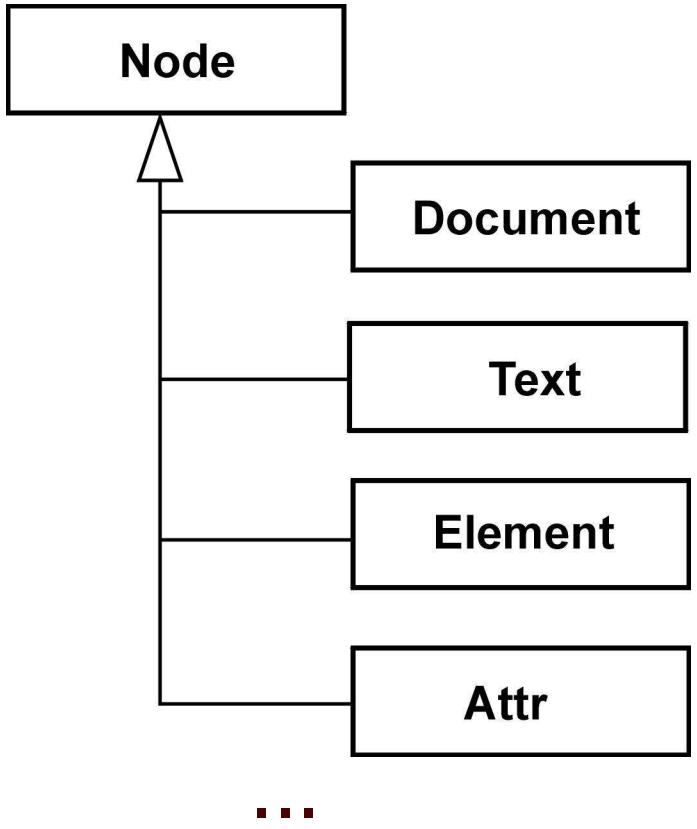
Bewertung von SAX

- sequentielle Verarbeitung
- kein Rück- bzw. Vorgriff auf andere Knoten
- geringer Ressourcenverbrauch (Speicher)
- keine Ausgabe vorgesehen

DOM

- sprachunabhängig
- kein Parser
- Baummodell stellt Struktur dar
- verschiedene Sprachbindungen
- Java-Sprachbindungen

Knotentypen



DomParser

```
org.apache.xerces.parsers.DomParser parser =
   new DomParser();
parser.parse(“beispiel.xml“);
Document doc = parser.getDocument();
```

Node

Methoden, die alle Knotentypen haben:

- getNodeType() : short
- getNodeName() : String
- getNodeValue() : String
- getParentNode() : Node
- getFirstChild() : Node
- getLastChild() : Node
- getChildNodes() : NodeList
- getAttributes() : NamedNodeMap

Bewertung von DOM

- sprachunabhängig
- nicht herstellerunabhängig
- hoher Speicherbedarf
- Vor- und Rückgriff auf andere Knoten
- keine Ausgabe vorgesehen

JDOM

- Java-API
- Baummodell
- Beta-Version
- einlesen von XML mittels SAXBuilder oder DOMBuilder

Ausgabe

org.jdom.output

- DOMOutputter
- SAXOutputter
- XMLOutputter

XMLOutputter

Ausgabe in Datei:

```
XMLOutputter out = new XMLOutputter();
FileOutputStream stream =
     new FileOutputStream(“beispiel.xml“);
out.output(JDOMDocument,stream);
```

Unterschiede zu DOM

- keine Erweiterung von DOM
- JDOM besteht aus konkreten Klassen
- Ausgabemechanismus
- JDOM noch im Standardisierungsprozess

JAXP

- Java-API von SUN
- Nutzung von SAX und DOM herstellerunabhängig
- Wechsel des Parsers ohne Neukompilierung
  - Wechsel der JAXP-Distribution
  - JAXP-Systemeigenschaften

Verwendung von SAX

```
SAXParserFactory factory =
   SAXParserFactory.newInstance();
factory.setValidating(true);
SAXParser parser = factory.newSAXParser();
parser.parse(“beispiel.xml“,new MyHandler());
```

- DOM analog mittels DocumentBuilder

XL

- Forschungsprojekt der TU München
- XML-Programmiersprache
- Entwicklung und Komposition von Web Services
- XML einziger Datentyp
- Konzentration des Programmierers auf Applikationslogik
- plattformunabhängig

Beispiel

```
service http://www.auktion.com
 let bieter;
 operation registriereBieter
  precondition empty($bieter[@uri=$input/uri]);
  postcondition exists($bieter[@uri=$input/uri]);
 body
  insert $input into $bieter;
  let $output = <msg>Danke.</msg>
 endbody
 endoperation
endservice
```

Zusammenfassung

- ereignisbasierte Verarbeitung
- baumstrukturierte Modelle
  - sprachunabhängig
  - an Java gebunden
- herstellerunabhängige Nutzung von SAX und DOM
- XML-Programmiersprache

Ende