

## 5. Ausgabedokument in einer Textdatei speichern

Der Ergebnisbaum wird im Beispiel `jdom2.java` als Abfolge von XML-Elementen (also „serialisiert“) auf der Konsole ausgegeben. Für komplexere Aufgaben ist es notwendig, die Ausgabe in einem neuen XML-Dokument zu speichern. Dazu importieren wir zunächst das Package `java.io` und erzeugen das File-Objekt `ausgabedatei`.

`jdom3.java`

```
import org.jdom.Document;
import org.jdom.JDOMException;
import org.jdom.input.SAXBuilder;
import org.jdom.output.XMLOutputter;
import org.jdom.transform.JDOMSource;
import org.jdom.transform.JDOMResult;
```

```
import java.util.List;
import javax.xml.transform.*;
import javax.xml.transform.stream.*;
import java.io.*;
```

```
public class jdom3 {
    static List ergebnis;
    static File ausgabedatei = new File("weblinks_out.html");

    public static void main (String args[]) throws Exception {
        SAXBuilder builder = new SAXBuilder();
        Document document = builder.build("weblinks.xml");
        ergebnis = transform(document, "weblinks.xsl");
        XMLOutputter xmlausgabe = new XMLOutputter();
        try {
            FileOutputStream ausgabestrom =
                new FileOutputStream(ausgabedatei);
            DataOutputStream datei = new DataOutputStream(ausgabestrom);
            xmlausgabe.output(ergebnis, datei);
        }
        catch (IOException e) {
            System.out.println(e);
        }
    }

    public static List transform(Document in, String stylesheet)
        throws JDOMException {
        ...
    }
}
```

Die Methode `transform()` wird so wie in Beispiel `jdom2.java` verwendet.

## 6. Aufgaben, Ausblick

1. Bestimmte Elemente einer XML-Datei sind mit Hilfe der JDOM-Methoden auszuwählen. Falls vorhanden sollen die Attribute und Attributwerte und die Elementinhalte ausgegeben werden.

2. Ein neues XML-Dokument ist mit den JDOM-Methoden `addContent(Element)` und `setAttribute(Attributname, Attributwert)` zu erzeugen (Hinweis:

```
Element wurzelement = new Element("wurzel");
wurzelement.setAttribute("id", "1");
```

```
Element kindelement = new Element("kind");
kindelement.addContent("Textinhalt");
```

```
wurzelement.addContent(kindelement);
```

```
Document dokument = new Document(wurzelement);
```

liefert den einfachen XML-Datenbaum

```
<?xml version="1.0" encoding="UTF-8"?>
<wurzel id="1"><kind>Textinhalt</kind></wurzel>
```

(vgl. dazu auch [13]).

## 7. Literatur, Weblinks

- [1] „JAVA und DOM“, PCNEWS-87 April 2004, S. 26
- [2] „JAVA und SAX“, PCNEWS-89 September 2004,
- [3] <http://www.s3.org/TR/REC-xml> (W3C-Empfehlung zu XML, Version 1.0)
- [4] <http://java.sun.com/j2se/1.4.2/docs/index.html> (Dokumentation aller verfügbaren Packages)
- [5] JDOM-Dokumentation (im Verzeichnis `build/apidocs` der entpackten JDOM-Distribution enthalten)
- [6] <http://www.blz.com/xt/index.html> (Homepage für den XSLT-Prozessor XT)
- [7] <http://xml.apache.org/xalan-j/> (XSLT-Prozessor XALAN im Rahmen des Apache-Projekts)
- [8] August Mistlbacher, Alfred Nussbaumer, „XML Ge-Packt“, mitp-Verlag
- [9] August Mistlbacher, Alfred Nussbaumer, „XML Ent-Packt“, mitp-Verlag
- [10] Herbert Schildt, „Java 2 Ent-Packt“, mitp-Verlag

# ADIM

Arbeitsgemeinschaft für  
Didaktik, Informatik und  
Mikroelektronik  
1190 Wien, Gatterburggasse 7  
Tel.: 01-369 88 58-88  
FAX.: 01-369 88 58-85

**Martin Weissenböck**

## EDV-Skripten

Schulbuch-Nr	Titel
	Turbo Pascal (Borland)
	RUN/C Classic
6226	Turbo-C (Borland)
	Turbo/Power-Basic
	DOS
6861	DOS und Windows
6476	Turbo-Pascal (Borland)
	Quick-Basic (Microsoft)
6450	C++ (Borland)
	AutoCAD I (2D-Grafik)
6863	AutoCAD I (2D-Grafik)
6864	AutoCAD II (AutoLisp+Tuning)
7571	AutoCAD III (3D-Grafik)
6862	Grundlagen der Informatik
7572	Visual Basic (Microsoft)
	Windows und Office
7573	Linux

## CDs

Telekommunikation III  
Multimedia Praxis  
Telekommunikation IV  
Multimedia Praxis 3  
Telekommunikation V/VI  
Multimedia Praxis 2000

## Bestellformular

<http://www.adim.at/dateien/BESTELL.pdf>

## Bestellhinweise

<http://www.adim.at/>