INSY

Document Object Model

Dominik Dolezal

Höhere Lehranstalt für Informationstechnologie

13. Februar 2017

Inhalt

tgm
Die Schule der Technik

CSV

tgm
Die Schule der Technik

Klassisches **tabellenbasiertes** Austauschformat

f. name	**name**	**address**	**city**	**state**
James	Smith	6649 N Blue Gum St	New Orleans	LA
Jenna	Darakjy	4 B Blue Ridge Blvd	Brighton	MI
Art	Venere	8 W Cerritos Ave 54	Bridgeport	NJ
Lenna	Paprocki	639 Main St	Anchorage	AK

- **C**omma-**s**eparated **v**alues
- Simple Textdateien mit speziellen Regeln
- Format für
  - Austausch zwischen Anwendungen
  - Übertragung im Web
  - Import-/Export-Funktionalitäten

tgm
Die Schule der Technik

Beispiel

f. name	name	address	city	state
James	Smith	6649 N Blue Gum St	New Orleans	LA
Jenna	Darakjy	4 B Blue Ridge Blvd	Brighton	MI
Art	Venere	8 W Cerritos Ave 54	Bridgeport	NJ
Lenna	Paprocki	639 Main St	Anchorage	AK

```
f. name;name;address;city;state
James;Smith;6649 N Blue Gum St;New Orleans;LA
Jenna;Darakjy;4 B Blue Ridge Blvd;Brighton;MI
Art;Venere;8 W Cerritos Ave 54;Bridgeport;NJ
Lenna;Paprocki;639 Main St;Anchorage;AK
```

CSV

tgm
Die Schule der Technik

`f. name;name;address;city;state`

- CSV ist *nicht* standardisiert!
- Lediglich allgemeine RFC-Beschreibung (RFC 4180)
- Nicht einmal der Zeichensatz ist geregelt
- Es gibt jedoch gängige Dialekte
- Mehrere mögliche Trennzeichen zwischen *Zellen*:
  Semikolon `;` Komma `,` Tabulator `\t` Doppelpunkt `:` Leerzeichen
- Mehrere mögliche Trennzeichen zwischen *Zeilen*:
  Newline `\n` oder CRLF (carriage return line feed) `\r\n`

CSV

Die Schule der Technik

James;Peter	Smith	6649 N Blue Gum St	New Orleans	LA

```
"James;Peter";Smith;6649 N Blue Gum St;...
```

- Zellen werden durch Anführungszeichen escaped
- Je nach Programm werden manchmal auch standardmäßig alle Zellen automatisch escaped

CSV

tgm
Die Schule der Technik

James;”Peter”	Smith	6649 N Blue Gum St	New Orleans	LA

```
"James;""Peter""";Smith;6649 N Blue Gum St;...
```

- Anführungszeichen werden fürs Escapen verdoppelt

CSV

tgm
Die Schule der Technik

f. name	name	address	city	state
James	Smith	6649 N Blue Gum St	New Orleans	LA
Jenna	Darakjy	4 B Blue Ridge Blvd	Brighton	MI

- CSV-Dateien sind simple Textdateien mit speziellen Regeln
- Es gibt verschiedene Dialekte
- Eignen sich gut für tabellarische Informationen
- Eignen sich schlecht für
  - hierarchische Beziehungen
  - referentielle Beziehungen
  - Unterstützung von Layout
  - benutzerdefinierte Regeln (z.B. nur Zahlen, E-Mail-Adressen)

JSON

tgm
Die Schule der Technik

- **J**ava**S**cript **O**bject **N**otation
- Ebenfalls nur eine RFC-Beschreibung
- JSON-Dokumente sind (mit kleinen Ausnahmen) valider JavaScript-Code, welcher 1:1 in JavaScript-Objekte umgewandelt werden kann
- Wird aber auch in (fast allen) anderen Programmiersprachen verwendet
- Hauptanwendungsgebiete sind JavaScript-Applikationen, Ajax, Webapplikationen und Webservices

JSON

tgm
Die Schule der Technik

```
[{
  "first_name":"James",
  "last_name":"Smith",
  "address":"6649 N Blue Gum St",
  "city":"New Orleans",
  "state":"LA",
  "zip":70116,
  "female":false,
  "phones":["504-621-8927","504-845-1427"]
},{
  "first_name":"Jenna",
  "last_name":"Darakjy",
  "address":"4 B Blue Ridge Blvd",
  "city":"Brighton",
  "state":"MI",
  "zip":48116,
  "female":true,
  "phones":["810-292-9388","810-374-9840"]}]
```

JSON

tgm
Die Schule der Technik

- JSON kennt Datentypen
  - Strings, Boolean, Zahlen
  - null
  - Arrays (in eckigen Klammern `[]`)
  - Objekte (in geschwungenen Klammern {})
- Objekte beinhalten eine (ungeordnete) Liste von Eigenschaften
- Eine Eigenschaft besteht aus einem Schlüssel (Zeichenkette) und einem Wert
- Hierarchien können dargestellt werden
- Einfache Repräsentation (Serialisierung) von Objekten der objektorientierten Programmierung
- Es sind jedoch keine komplexen Zusammenhänge darstellbar oder fortgeschrittene Validierungen möglich

XML

tgm
Die Schule der Technik

- **E**xtensible **M**arkup **L**anguage
- Eine Auszeichnungssprache
- Ebenfalls Textdokumente, die „menschenlesbar“ sein sollen
- Strukturierte Darstellung von Informationen (ggf. auch inkl. benutzerdefinierten Regeln)
- Ebenfalls zur Serialisierung von Objekten geeignet
- Anwendung bei Webapplikationen, Webservices, Konfigurationen, ...
- Spezifikation von der W3C

XML

tgm
Die Schule der Technik

```
<?xml version="1.0"?>
<menu>
  <food calories="650">
    <name>Belgian Waffles</name>
    <price>$5.95</price>
    <description>
      Two of our famous Belgian Waffles with plenty of real
          maple syrup
    </description>
  </food>
  <food calories="900">
    <name>Strawberry Belgian Waffles</name>
    <price>$7.95</price>
    <description>
      Light Belgian waffles covered with strawberries
    </description>
  </food>
</menu>
```

XML

tgm
Die Schule der Technik

```
<menu>
  <food calories="650">
    <name>Belgian Waffles</name>
    <price>$5.95</price>
    <description>
      Two of our famous Belgian Waffles with plenty of real
          maple syrup
    </description>
  </food>
</menu>
```

- Es gibt immer genau ein Wurzelelement
- Elemente werden geöffnet und müssen wieder geschlossen werden
- Elemente können weitere Elemente sowie Attribute beinhalten

XML vs. HTML

HTML beschreibt Layout des Inhalts, XML beschreibt Struktur/Semantik des Inhalts

```
<h1>HandyKatalog</h1>
<h2>Nokia 8210</h2>
<table border="1">
 <tr>
  <td>Batterie</td><td>900mAh</td>
 </tr>
 <tr>
  <td>Gewicht</td><td>141g</td>
 </tr> …
</table>
```


```
<HandyKatalog>
 <Hersteller name="Nokia">
  <Modell name="8210">
   <Batterie>900mAh</Batterie>
   <Gewicht>141g</Gewicht>
   …
  </Modell>
 </Hersteller>
</HandyKatalog>
```

DOM

tgm
Die Schule der Technik

- **D**ocument-**O**bject-**M**odel
- Ebenfalls W3C-Standard
- Für den programmgesteuerten Zugriff auf XML-Dateien
- Darstellung des Dokuments als Baumstruktur
- Gesamtes Dokument wird in den Speicher geladen
- Wahlfreier Zugriff möglich

DOM

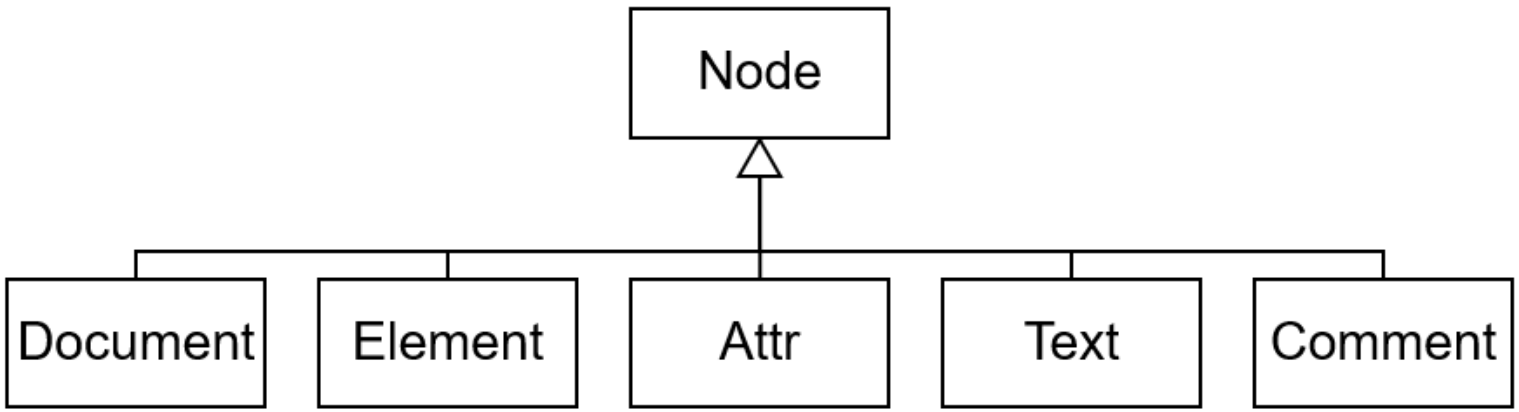
tgm
Die Schule der Technik

- Die Spezifikation von DOM unterscheidet prinzipiell zwischen
  - DOM Core: Wichtige gemeinsame Interfaces (Node, Element, ...) und Konzepte (Namespaces, ...)
  - DOM HTML: HTML-spezifische Definitionen (HTML-Elemente, Attribute, Methoden und Events)
  - DOM XML: Modell für Zugriff und Manipulation von XMl-Dateien
- Außerdem gibt es weitere Spezifikationen für weitere Konzepte (Validierung, XPath, ...)

DOM Node

tgm
Die Schule der Technik

Wichtigstes Interface: `Node`

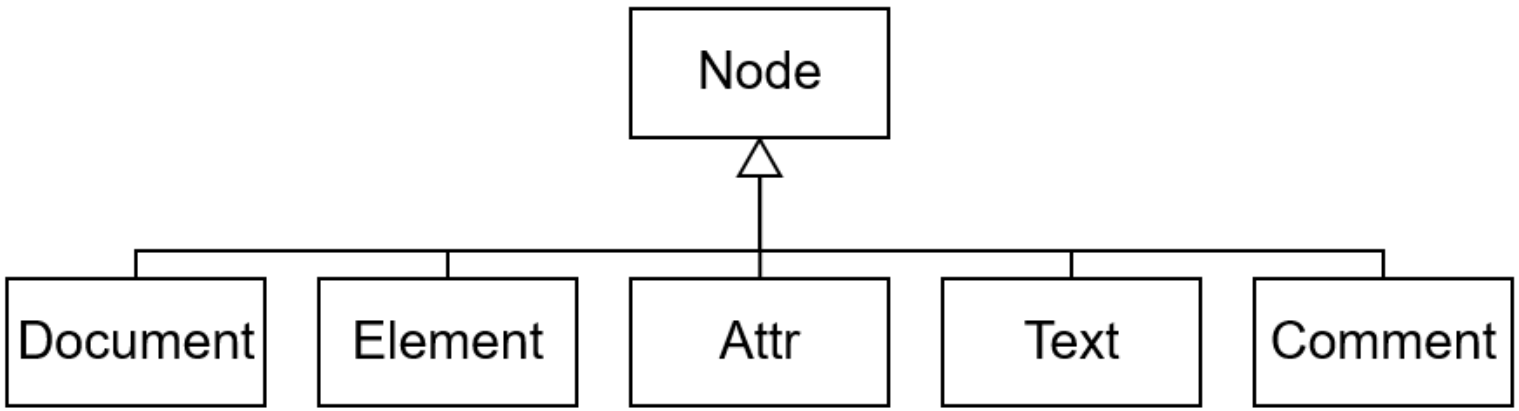


Alle Subtypen sind ebenfalls Interfaces (konkrete Implementierung ist egal)!

DOM Node

tgm
Die Schule der Technik

Je nach tatsächlichen Typ gibt es unterschiedliche mögliche Kindelemente und Rückgabewerte von `nodeValue` und `nodeName`!
Komplette Auflistung (**Lehrstoff!**):
http://www.w3schools.com/xml/dom_nodetype.asp



tgm
Die Schule der Technik

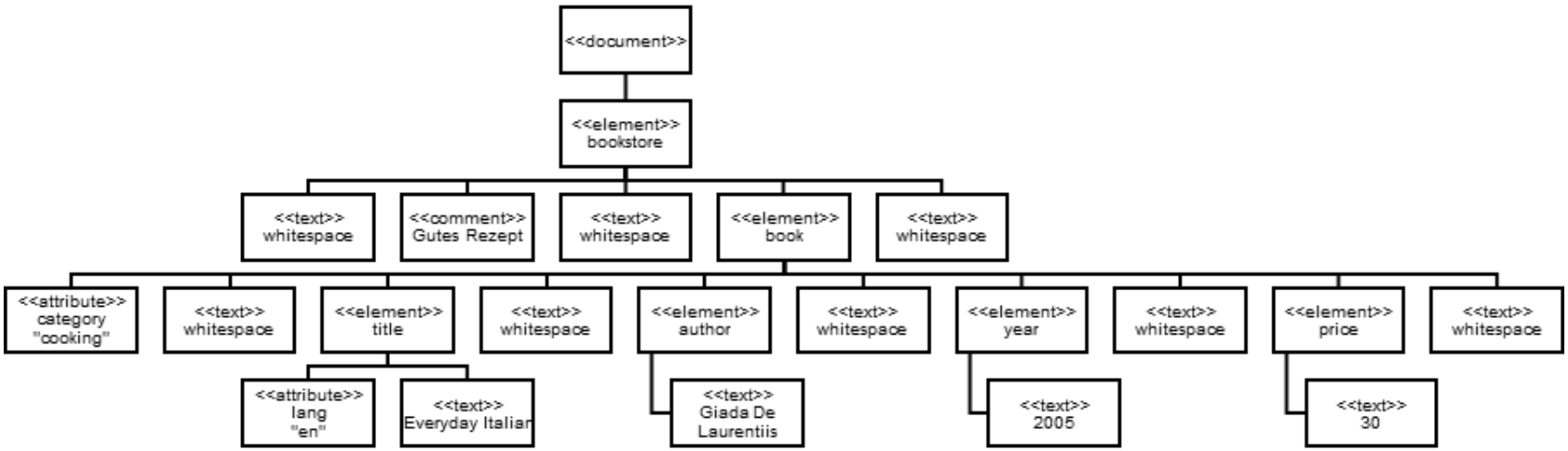
Wie viele Nodes besitzt der DOM-Baum?

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>

<!-- Gute Rezepte -->
<book category="cooking">
<title lang="en">Everyday Italian</title>
<author>Giada De Laurentiis</author>
<year>2005</year>
<price>30.00</price>
</book>

</bookstore>
```

DOM

tgm
Die Schule der Technik

22 Nodes.



Auf Whitespaces aufpassen!

Python: XML-Dokument einlesen

tgm
Die Schule der Technik

```
from xml.etree import ElementTree as et

doc = et.parse("bookstore.xml")
print (et.tostring(doc.getroot()).decode('utf-8'))
```

Es gibt in Python mehrere XML-APIs – eine davon ist `ElementTree`

Python: find und text

Die Schule der Technik

```
from xml.etree import ElementTree as et

doc = et.parse("bookstore.xml")
print(doc.find("book/title").text)
```

`find` liefert das erste passende Element anhand des Namens bzw. Pfads zurück
`text` liefert den Textinhalt der Node

Python: findall

tgm
Die Schule der Technik

```
from xml.etree import ElementTree as et

doc = et.parse("bookstore.xml")

for element in doc.findall("book"):
    print(element.find("author").text + " " +
          element.find("year").text +
          ", EUR" + element.find("price").text)
```

`findall` liefert eine Liste an Elementen über den Namen oder Pfad

Python: attrib

tgm
Die Schule der Technik

```
from xml.etree import ElementTree as et

doc = et.parse("bookstore.xml")

for element in doc.findall("book"):
    print(element.attrib)
    print(element.attrib["category"])
```

attrib speichert als assoziatives Array alle Attribute des Elements

Python: Kindelemente und tag

tgm
Die Schule der Technik

```
from xml.etree import ElementTree as et

doc = et.parse("bookstore.xml")

for child in doc.getroot():
    print(child.tag, child.attrib)
```

Um alle Kindelemente zu erhalten, können `Element`-Objekte direkt iteriert werden

Python: write

tgm
Die Schule der Technik

```
from xml.etree import ElementTree as et

doc = et.parse("bookstore.xml")

for element in doc.findall("book"):
    if "stock" not in element.attrib:
        element.attrib["stock"] = "10"

doc.write("bookstore2.xml")
```

Über **not in** wird geprüft, ob sich das Attribut im assoz. Array befindet
`write` schreibt den gesamten DOM-Baum in das angegebene File

Java: XML-Dokument einlesen

tgm
Die Schule der Technik

```
Document doc = null;
DocumentBuilderFactory factory =
    DocumentBuilderFactory.newInstance();
try {
  DocumentBuilder builder = factory.newDocumentBuilder();
  doc = builder.parse(new File("bookstore.xml"));
  // ...
} catch (SAXParseException e) {
  e.printStackTrace();
} catch (SAXException e) {
  e.printStackTrace();
} catch (ParserConfigurationException e) {
  e.printStackTrace();
} catch (IOException e) {
  e.printStackTrace();
}
```

Java: Imports

tgm
Die Schule der Technik

```
import java.io.File;
import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;

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

Java: NodeList

tgm
Die Schule der Technik

Elemente auf höchster Ebene mithilfe von `NodeList` durchgehen:

```
NodeList kinder = doc.getChildNodes();
for(int i = 0; i < kinder.getLength(); i++) {
  // einzelnen Node aus der NodeList holen
  Node docChild = kinder.item(i);
  System.out.println(docChild.getNodeName() + " - "
    + docChild.getNodeValue());
}
```

Wurzelelement holen:

```
Node root = doc.getDocumentElement();
```

Java: NodeType

tgm
Die Schule der Technik

NodeType überprüfen:

```
Node kind =
    doc.getDocumentElement().getChildNodes().item(3);
if(kind.getNodeType()==Node.ELEMENT_NODE) {
  NamedNodeMap bookAttribute =
      kind.getAttributes();
}
```

Java: Attribute

tgm
Die Schule der Technik

Attribute durchgehen:

```
Node kind =
    doc.getDocumentElement().getChildNodes().item(3);
if(kind.getNodeType()==Node.ELEMENT_NODE) {
  NamedNodeMap bookAttribute = kind.getAttributes();
  for(int ba = 0; ba < bookAttribute.getLength(); ba++) {
    Node bookAttribut = bookAttribute.item(ba);
    System.out.println("Attribut von book: \n"
      + bookAttribut.getNodeName() + " - "
      + bookAttribut.getNodeValue());
  }
}
```

Java: Attribute

tgm
Die Schule der Technik

Attribute über Namen ansprechen:

```
Node book =
    doc.getDocumentElement().getChildNodes().item(3);
if(book.getNodeType() == Node.ELEMENT_NODE) {
  NamedNodeMap bookAttribute = book.getAttributes();
  Node bookAttribut =
      bookAttribute.getNamedItem("category");
  System.out.println("Attribut von book: "
    + bookAttribut.getNodeName() + " - "
    + bookAttribut.getNodeValue());
}
```

Java: getElementsByTagName

tgm
Die Schule der Technik

Alle Elemente eines Dokuments mit bestimmten Namen erhalten:

```
NodeList x = doc.getElementsByTagName("title");
for (int i=0; i < x.getLength(); i++){
  Node title = x.item(i);
  System.out.println("Element: " + title.getNodeName() + " "
    + title.getNodeValue());
}
```

Java: Speichern

tgm Die Schule der Technik

```
// Vorbereiten zum Speichern
DOMSource source = new DOMSource(doc);
// Ausgabe-Kanal festlegen
StreamResult result = new StreamResult(new
    File("output.xml"));
// Mit Hilfe der factory kann ein Transformer geholt werden
TransformerFactory tFactory =
    TransformerFactory.newInstance();
try {
  // Laden des Transformers
  Transformer transformer = tFactory.newTransformer();
  // Speichern der vorbereiteten Daten im Ausgabe-Kanal
transformer.transform(source, result);
} catch (TransformerConfigurationException e) {
  e.printStackTrace();
} catch (TransformerException e) {
  e.printStackTrace();
}
```

Java: Attribute setzen

tgm
Die Schule der Technik

```
NodeList titles = doc.getElementsByTagName("title");
for (int i = 0; i < x.getLength(); i++) {
  Element title = (Element) x.item(i);
  ((Element) title).setAttribute("short", "true");
}
```

Zusammenfassung

tgm
Die Schule der Technik

Document Object Model

- W3C-Standard zur Darstellung von Dokumenten als Baumstruktur
- Zugriff in Python über `ElementTree`-API
- Wichtige Methoden / Variablen:
  - `ElementTree.parse()`
  - `Element.find()`
  - `Element.findall()`
  - `Element.text`
  - `Element.attrib`