Einführung in XML
Extensible Markup Language

- XML-Dokumente
- Dokumenttyp-Definition (DTD)
- Stylesheets (CSS, XSL)

Online-Doku:

- `http://www.w3.org/XML/`
- `http://xml.coverpages.org/xmlIntro.html`

Hello, XML!

- XML-Datei: hello.xml

```
<?xml version="1.0" standalone="yes"?>
<GREETING>
  Hello XML!
</GREETING>
```

- Stylesheet: hello.css

```
GREETING {display: block; font-size: 24pt;
          font-weight: bold;}
```

- Browser: Mozilla, Netscape 6 oder Internet Explorer 5

Hello, HTML!

- HTML-Datei: hello.html

```
<html>
<body>
  Hello HTML!
</body>
</html>
```

- Stylesheet: helloHTML.css

```
body {display: block; font-size: 24pt;
      font-weight: bold;}
```


Positive Aspekte von XML

- Erweiterbarkeit
- Entwurf domänenspezifischer Auszeichnungssprachen
- Selbst beschreibende Daten
- Datenaustausch

Domänenspezifischer Sprachen (Bsp.)

- CML – *Chemical Markup Language*
- MathML – *Mathematical Markup Language*
- SMIL – *Synchronized Multimedia Integration Language*
- MusicML
- OFX – *Open Financial Exchange*
- · · ·

Beispiel: Selbst beschreibende Daten (Bsp.)

⟨PERSON PNUMBER=”p1100” SEX=”M”⟩
   ⟨NAME⟩
      ⟨GIVEN⟩ Judson ⟨/GIVEN⟩
      ⟨SURNAME⟩ McDaniel ⟨/SURNAME⟩
   ⟨/NAME⟩
   ⟨BIRTH⟩
      ⟨DATE⟩ 21 FEB 1834 ⟨/DATE⟩
   ⟨/BIRTH⟩
   ⟨DEATH⟩
      ⟨DATE⟩ 9 DEC 1905 ⟨/DATE⟩
   ⟨/DEATH⟩
⟨/PERSON⟩

XML-Dokumente

- XML-Tags
- XML-Attribute
- XML-Elemente

XML-Tags

- Beispiel: ⟨GREETING⟩, ⟨PERSON⟩, ⟨NAME⟩, ⟨MOLECULE⟩, · · ·
- Bedeutung
  - Wiedergabe der Information
  - Standard-Tag-Sets für bestimmtes Fachgebiet
- Struktur – geschachtelte Tags

XML-Elemente

- XML-Deklaration: ⟨?xml version=”1.0”?⟩
- Root-Element, z.B.: GREETING, PERSON
- Child-Element, z.B.:
  - PERSON: NAME, BIRTH, DEATH
  - NAME: GIVEN, SURNAME
  - BIRTH und DEATH: DATE

XML-Attribute

- Beispiel: ⟨PERSON PNUMBER=”p1100” SEX=”M”⟩

  ⟨PERSON⟩
  ⟨PNUMBER⟩ p1100 ⟨/PNUMBER⟩
  ⟨SEX⟩ M ⟨/SEX⟩
  . . .
  ⟨/PERSON⟩

- Elemente können Unterstruktur haben, Attribute nicht.

DTD – Dokumenttyp-Definition

- Entities
- Elemente
- Attribute

Entities in DTD

- Entity-Referenz
  ⟨!ENTITY *name* “replacement text” ⟩
  vordefinierte externe Entities: <, >, &, ', "
  ⟨!ENTITY apos “'”⟩
- Parameter-Entity
  ⟨!ENTITY % *name* “replacement text” ⟩
  Beispiel:
  ⟨!ENTITY % betreuer “WIMI | LA | TUTOR” ⟩

Elemente in DTD

- Einfache Elemente
  ⟨ELEMENT TITLE (#PCDATA)⟩
- Strukturierte Elemente
  ⟨ELEMENT SWP01 (TITLE,PROFESSOR,MITARBEITER*) ⟩
- Alternation
  ⟨ELEMENT MITARBEITER (WIMI | LA | TUTOR) ⟩
- Wiederholung
  ⟨ELEMENT TERM (%betreuer;)* ⟩

Attribute in DTD

- Deklaration:
  ⟨!ATTLIST *Element_name Attribute_name Type Default_Value*

  ⟨!ATTLIST PERSON PNUMBER ID #REQUIRED
  SEX CDATA #REQUIRED
- Type:
  CDATA, Enumerated, ID, $\cdots$
- Default_Value:
  #REQUIRED, #IMPLIED, $\cdots$

CSS – Cascading Stylesheets

- CSS-Regeln, z.B.:
  - PERSON
    {font-size: 18pt; background-color: white; color: black; display: block}
  - NAME, BIRTH, DEATH {font-size: 14pt; display: block}
  - GIVEN, SURNAME {font-size: 12pt}
  - DATE {font-size: 12pt}
- Stylesheets mit Dokumenten verbinden, z.B.:
  ⟨?xml-stylesheets type="text/css" href="person.css"? ⟩

XSL – Extensible Style Language

- Transformations- und Formatierungssprache
- Transformation: XML-Repräsentation $\rightarrow$ XML-Repräsentation
  - E-Commerce
  - Datenaustausch
- Formatierung

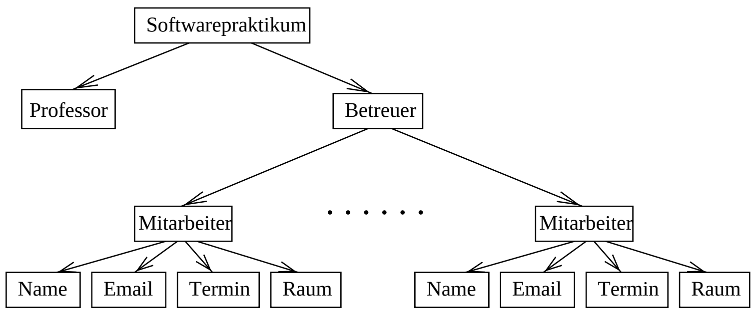
XSL-Transformation

- Der Webbrowser transformiert das XML-Document mit dem Stylesheet.
  (Internet Explorer 5.0, Netscape 6)
- Der Server transformiert das XML-Document mit dem Stylesheet und sendet das transformierte Dokument an den Webbrowser.
  (servlet-kompatibler Webserver)
- Ein drittes Programm transformiert das XML-Dokument mit dem Stylesheet in ein anderes Format (z.B. HTML).
  (XT von James Clark)

Daten strukturieren mit XML

- Daten analysieren
- Dokumenttyp definieren
- XML-Dokument erstellen
- Stylesheet fuer die Darstellung des Dokuments erstellen

Beispiel: Team des Softwarepraktikums



Die Struktur des Teams

Dokumenttyp-Definition: swp01.dtd

```
<!ELEMENT SWP01 (TITLE,PROFESSOR,MITARBEITER+)>

<!ELEMENT TITLE (#PCDATA)>
<!ELEMENT PROFESSOR (NAME)>
<!ELEMENT MITARBEITER (NAME,EMAIL,TERMIN,RAUM)>

<!ELEMENT NAME (#PCDATA)>
<!ELEMENT EMAIL (#PCDATA)>
<!ELEMENT TERMIN (#PCDATA)>
<!ELEMENT RAUM (#PCDATA)>
```

XML-Dokument erstellen: swp01.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
                 href="swp01.xsl"?>
<!DOCTYPE SWP01 SYSTEM "swp01.dtd">
<SWP01>
 <TITLE> Softwarepraktikum Sommersemester 2
 </TITLE>
 <PROFESSOR>
   <NAME> Martin Gogolla </NAME>
 </PROFESSOR>
 <MITARBEITER>
   <NAME> Ralf Kollmann </NAME>
   <EMAIL> kollmann@tzi.de </EMAIL>
   <TERMIN> Fr. 13-15 </TERMIN>
   <RAUM> MZH7200 </RAUM>
 </MITARBEITER>
 <MITARBEITER>
   <NAME> Andreas Lattner </NAME>
   <EMAIL> adl@tzi.de </EMAIL>
   <TERMIN> Do. 13-15 </TERMIN>
   <RAUM> MZH1400 </RAUM>
 </MITARBEITER>
 <MITARBEITER>
   <NAME> Stefan Prelle </NAME>
   <EMAIL> prelle@tzi.de </EMAIL>
```



```
   <TERMIN> Mi. 8-10 </TERMIN>
   <RAUM> MZH5290 </RAUM>
 </MITARBEITER>
 <MITARBEITER>
   <NAME> Mark Richters </NAME>
   <EMAIL> mr@tzi.de </EMAIL>
   <TERMIN> Mi. 8-10 </TERMIN>
   <RAUM> MZH5290 </RAUM>
 </MITARBEITER>
 <MITARBEITER>
   <NAME> Hui Shi </NAME>
   <EMAIL> shi@tzi.de </EMAIL>
   <TERMIN> Mi. 8-10 </TERMIN>
   <RAUM> MZH6240 </RAUM>
 </MITARBEITER>
 <MITARBEITER>
   <NAME> </NAME>
   <EMAIL> </EMAIL>
   <TERMIN> Mi. 10-12 </TERMIN>
   <RAUM> MZH7260 </RAUM>
 </MITARBEITER>
 <MITARBEITER>
   <NAME> Anke Werner </NAME>
   <EMAIL> anke@tzi.de </EMAIL>
   <TERMIN> Mi. 8-10 </TERMIN>
   <RAUM> MZH6240 </RAUM>
 </MITARBEITER>
</SWP01>
```

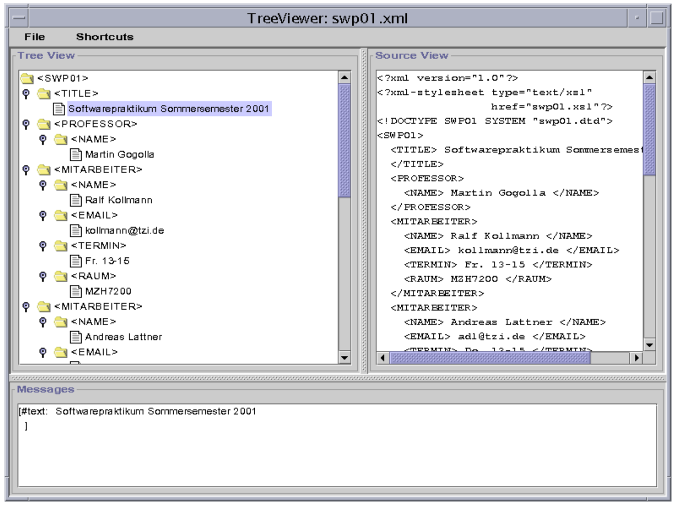

XML-Parser Xerces: Eine Java-Anwendung

Source http://xml.apache.org/
Xerces-J-bin.1.3.1.tar.gz

CLASSPATH .:/home/dbs/lehre/swp2001/tools/xerces-1_3_1/xerces.jar:
/home/dbs/lehre/swp2001/tools/xerces-1_3_1/xercesSamples.jar

Start java sax.SAXCount -v swp01.xml
java ui.TreeView swp01.xml
java dom.traversal.TreeWalkerView swp01.xml

Beispiel: TreeView swp01.xsl



Stylesheet: swp01.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org
                        /1999/XSL/Transform">

  <xsl:template match="/">
    <html>
    <head>
      <title>
        <xsl:value-of select="/SWP01/TITLE"/
      </title>
    </head>
    <body bgcolor="#ffffff">
      <xsl:apply-templates select="SWP01"/>
    </body>
    </html>
  </xsl:template>

  <xsl:template match="SWP01">
    <h1>
      <font face="Helvetica, Arial" size="48
       color="blue">
      <xsl:value-of select="TITLE"/>
      </font>
      <br/>
    <a href="http://www.rational.com">
```



```
      <img border="0" src="/home/shi/Lehre
                  /SWP01/XT/UML.gif"/></a>
    <a href="http://www.javasoft.com">
      <img border="0" src="/home/shi/Lehre
             /SWP01/XT/javalogos.gif"/></a>
    </h1>
    <hr/>
    <h1>
      Professor: <xsl:value-of select="PROFE
    </h1>
    <table>
      <th>Tutor</th>
      <th>EMail</th>
      <th>Termin</th>
      <th>RAUM</th>
    <xsl:apply-templates select="MITARBEITER
    </table>
  </xsl:template>

  <xsl:template match="MITARBEITER">
    <tr>
      <td><xsl:value-of select="NAME"/></td>
      <td><xsl:value-of select="EMAIL"/></td
      <td><xsl:value-of select="TERMIN"/></t
      <td><xsl:value-of select="RAUM"/></td>
    </tr>
  </xsl:template>
</xsl:stylesheet>
```


Konverter XT: Eine Java-Anwendung

Source http://www.jclark.com/xml/xt.html
http://www.jclark.com/xml/xp/index.html

CLASSPATH .:/home/dbs/lehre/swp2001/XML/XT/xt.jar:
/home/dbs/lehre/swp2001/XML/XT/sax.jar:
/home/dbs/lehre/swp2001/XML/XP/xp.jar

Start java -Dcom.jclark.xsl.sax.parser=com.jclark.xml.sax.Driver
com.jclark.xsl.sax.Driver mydoc.xml mydoc.xsl mydoc.html

Transformation: swp01.html