Vorlesung / Übungen

Multimedia Technologie II

Prof. Dr. Michael Frank / Prof. Dr. Klaus Hering

Sommersemester 2004

HTWK Leipzig, FB IMN

3. Extensible Markup Language (XML)

Wie SGML ist XML eine *Meta-Markup Language*, d.h. sie bietet Mechanismen zur Definition von Markup Languages.

Alle über XML definierten Sprachen (bezeichnet als *XML-Applikationen* oder *Dokumenttypen*) müssen gewissen syntaktischen Grudregeln (z. B. bezüglich der Bildung und Verwendung von *tags*) genügen. Diesen Regeln entsprechende Dokumente heißen *wohlgeformt (well-formed).*

Darüber hinausgehende Regeln zur Definition von XML-Applikationen werden zum Beispiel im Rahmen von *Dokumenttyp-Definitionen (DTD)* gegeben. Diese verkörpern eine Menge von Deklarationen zur Einführung von *Elementnamen* (als Bezeichnung durch *tags* eingegrenzter Dokumentteile), *Attributnamen* (zur Verbindung von Elementen mit Eigenschaften), *Entitynamen* (als Bezeichnung von Dokumentteilen) und *Typnamen* (zur Charakterisierung speziell zu behandelnder Dokumentteile).

XML Schemata stellen eine alternative Definitionsform für Dokumenttypen dar.

Dokument aus der Mathematics Markup Language (MathML)

```
<?xml version="1.0"?>
<math xmlns="http://www.w3.org/TR/REC-MathML/">
  <mi>F</mi><mo>=</mo>
  <mi>G</mi><mo>&InvisibleTimes;</mo>
  <mfrac>
    <mrow>
      <mi>M</mi><mo>&InvisibleTimes;</mo><mi>m</mi>
    </mrow>
    <apply>
      <power><mi>r</mi><mn>2</mn></power>
    </apply>
  </mfrac>
</math>
```

verkörpert Gleichung

$F = GMm / r^2$

(Newton'sches Gravitationsgesetz)

Dokument aus der Scalable Vector Graphics (SVG) Language

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg
    PUBLIC "-//W3C//DTD SVG 20001102//EN"
    http://www.w3.org/TR/2000/CR-SVG-20001102/DTD/svg-20001102.dtd>
<svg>
        <desc>Three shapes</desc>
        <rect fill="green" x="1cm" y="1cm" width="3cm" height="3cm"/>
        <circle fill="red" cx="3cm" cy="2cm" r="4cm"/>
        <polygon fill="blue" points="110,160 50,300 180,290"/>
</svg>
```

verkörpert drei Figuren (Rechteck, Kreis, Polygon)

⇒ vielfältige Interpretationsmöglichkeiten von XML-Dokumenten

⇒ vielversprechende Anwendung von XML zur Spezifikation von Formaten für den Datentransfer zwischen Applikationen

XML beschreibt die Struktur und Semantik, aber nicht die Formatierung von Sprachelementen. Letztere wird über *Style Sheets* festgelegt. Das birgt eine Reihe von Vorteilen:

- gleiche Stilvorgaben können für viele Dokumente verwendet werden
- Änderungen einer Stilvorgabe erfolgen konzentriert an einem Ort und alle damit in Verbindung stehenden Dokumente werdadurch beeeinflußt
- entsprechend dem Verwendungszweck eines Dokuments können Stylesheets ausgetauscht werden
- Experimente in Bezug auf das Layout lassen die Dokumentstruktur völlig unberührt
- keine textliche Vermischung von Notationen zu Inhalt und Form

3.1 Syntaktische Konstrukte

Elemente verkörpern die Grundbausteine von XML. Dabei handelt es sich um in *tags* eingeschlossene Dokumentteile, die eine Mischung aus Text und anderen Elementen enthalten können.

```
<el1>ein reines Textelement</el1>
<outer>Text<inner>Text</inner>Text</outer>
```

Spezialfall eines „leeren“ Elements:

```
<empty/>
```

Elemente können durch *Attribute* näher beschrieben werden. Attributnamen dürfen nicht mehrfach an ein Element gebunden werden.

Container-Element:

<*name* att_1 = "val_1" … att_i = "val_i" > ——— start tag

element_content

(übereinstimmender Name,
Groß- und Kleinschreibung beachten!)

</*name*> ——— end tag

Leeres Element:

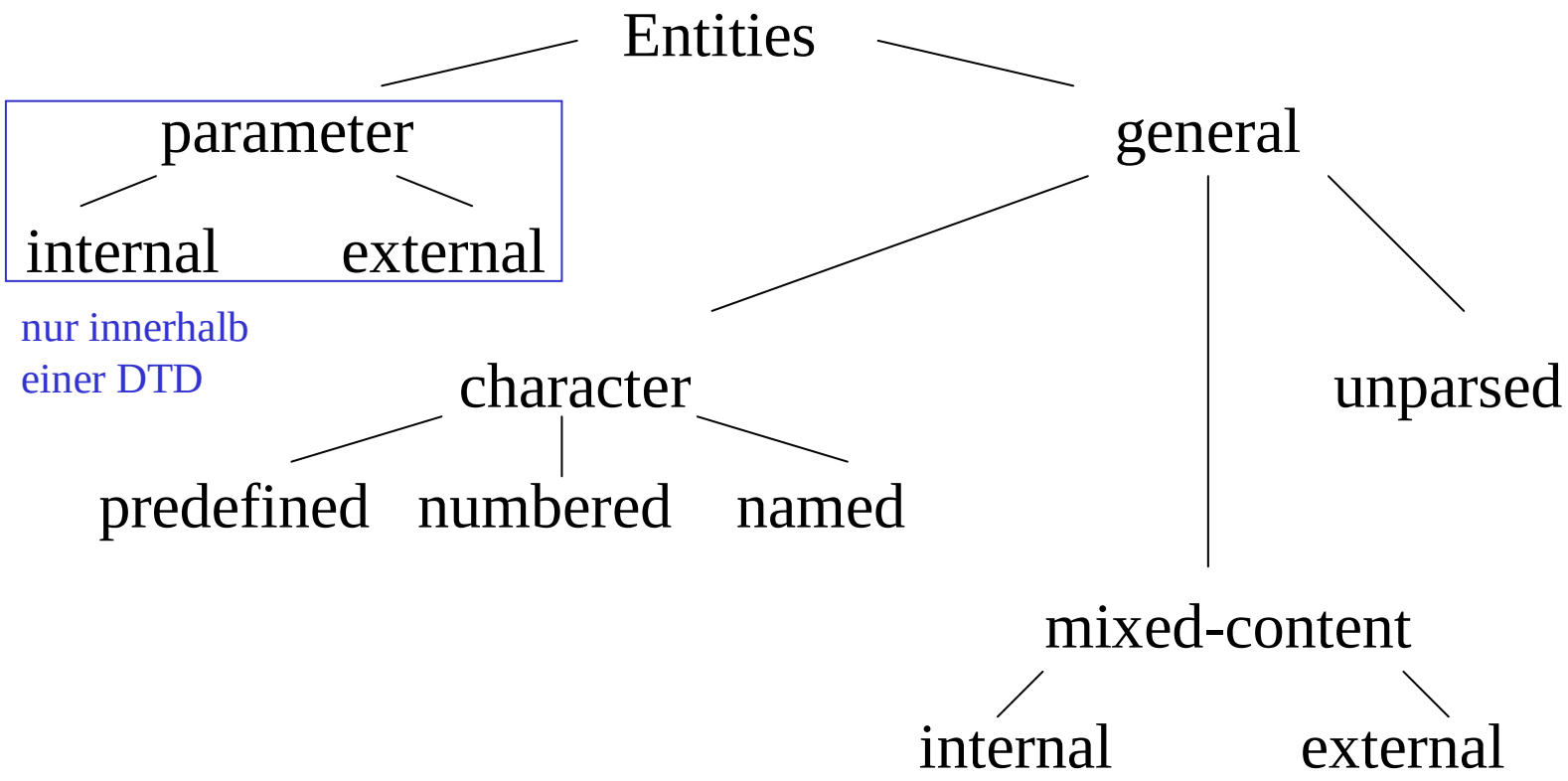
<*name* att_1 = "val_1" … att_i = "val_i" /> ———

Attributlisten können leer sein

Regeln für wohlgeformte Dokumente:

- Von zwei Elementen innerhalb eines Dokuments ist entweder eines vollständig im anderen enthalten oder beide überlappen nicht.
- Die Sonderzeichen < , > , [,] , & treten nicht isoliert im content-Bereich auf (Ausnahme: CDATA-Bereiche)
- Element-Namen müssen mit einem Buchstaben oder _ beginnen. Sie dürfen lediglich Buchstaben, Ziffern oder Zeichen aus der Menge {- , . , _} enthalten (außerdem : im Zusammenhang mit *name spaces*).

Innerhalb von XML-Dokumenten fungieren sogenannte *Entities* als Platzhalter (Abkürzungen) für Dokumententeile. Sofern sie nicht vordefiniert sind, werden Entities im Dokumentprolog oder in einer externen DTD deklariert.



Entity-Deklaration:

<!ENTITY *name* “*value*“>

Bezugnahme:

&*name*; (general entity) % *name*; (parameter entity)

Predefined character entities:

(name,value): (amp,&) (apos,‘) (gt,>) (lt,<) (quot,“)

Bezug auf eine numbered character entity:

&#xa3 (163. Unicode character £)

Named character entities sind explizit zu deklarieren aber einfacher zu merken als numbered character entities. Es gibt eine Reihe vorgefertigter DTDn mit entsprechenden Deklarationen.

Internal mixed-code entity:

```
<!ENTITY IBM “<company>International Business Machines Corporation</company>“>
```

External mixed-code entity:

```
<!ENTITY chap2 SYSTEM “chap02.xml“>
```

Unparsed entity:

```
<!ENTITY picture SYSTEM “person.gif“ NDATA GIF>
```

Konstrukte, die ihren Inhalt vor dem Parser abschirmen:

Comments, CDATA Sections, Processing Instructions

Kommentar:

<!-- *text and markup* -->

CDATA section:

<![CDATA[*text*]]>

CDATA sections verkörpern für den Parser Dokumententeile, die frei von Markup-Konstrukten sind. Die Verwendung dieser Konstrukte bietet sich zum Beispiel zur Darstellung von Programmfragmenten an:

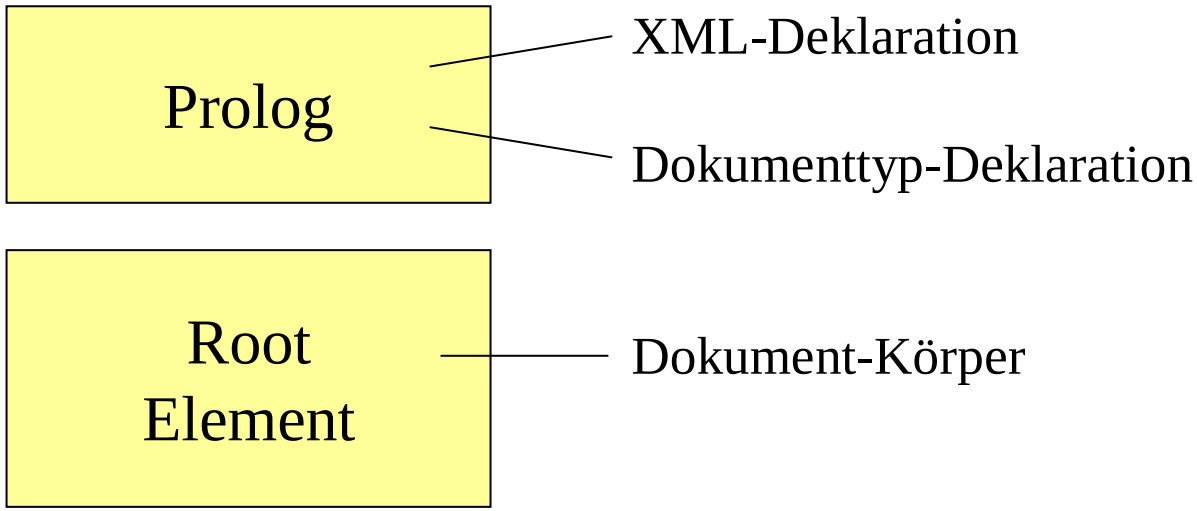
<![CDATA[if (&x < &y)]]>

Processing instructions:

<?*name data*?>

Ein *processing instruction handler*, der den angegebenen Namen akzeptiert, kann die Daten verwenden.

Dokument-Gesamtstruktur:



XML-Deklaration:

Eigenschaftsdefinitionen

<?xml *name*$_1$ = “*val*$_1$“ … *name*$_i$ = “*val*$_i$“ ?>

Eigenschaft Wert

mögliche Eigenschaften:

version

gibt XML-Versionsnummer an

encoding

gibt den verwendeten Zeichensatz an (standardmäßig UTF-8)

standalone

gibt an, ob weitere Files geladen werden müssen (z.B. eine Dokumenttyp-Definition oder externe Dokumentteile), Werte: yes, no

Die Angabe von Eigenschaftsdefinitionen ist optional, allerdings sollte wenigstens die XML-Versionsnummer bereitgestellt werden.

Dokumenttyp-Deklaration:

<!DOCTYPE *root_element_name*
DTD_reference
[*internal_declarations*
]>

external subset

internal subset

- *root_element_name* stimmt mit dem Namen des *tags* überein, welcher zum Einschließen des Dokument-Körpers verwendet wird
- *DTD_reference* verkörpert einen (optionalen) Verweis auf eine externe Datei, die in einer DTD zugelassene Deklarationen enthält
- Die zu einem Dokument gehörende DTD wird durch die Gesamtheit der Deklarationen in *internal subset* und *external subset* bestimmt (beide Teile können leer sein)
- Falls vorhanden, überlagern interne Deklarationen auf das gleiche Objekt bezogene externe Deklarationen

Beispiel:

```
<?xml version="1.0"?>
<!DOCTYPE book
    PUBLIC "-//ORA//DTD DBLITE XML//EN"
    SYSTEM "usr/local/prod/dtds/dblite.dtd"
[
    <!ENTITY chap1 SYSTEM "ch01.xml">
    <!ENTITY chap2 SYSTEM "ch02.xml">
    <!ENTITY xml "<acronym>XML</acronym>">
] >
```

```
<book>
        …
</book>
```

3.2 Dokumenttyp-Definition (DTD)

im folgenden:
Konstruktion einer kompletten Beispiel-DTD

⇒ Dokumenttyp (XML-Applikation) **Katalog**
(in Bezug auf Bücher)
⇒ siehe
P. Spencer: Professional XML Design and Implementation.
Wrox Press, 1999.

Vorgehen:

- TOP-DOWN Betrachtung der Dokumentstruktur
- Einführung entsprechender Elemente
- Erzeugung zugehöriger Deklarationen
  (Elementtyp-, Attributlisten-, Entity-, Notations-Deklarationen)

Informale Beschreibung:

Ein Katalog soll eine Folge von Buchbeschreibungen enthalten. Zu einer Buchbeschreibung sollen Titel, Autoren, Verlag, Seitenzahl, ISBN-Nummer und Preisangaben, ggf. in unterschiedlichen Währungen, gehören.

Einführung des root-Elements: **Katalog**

Die folgende Elementtyp-Deklaration drückt aus, dass ein **Katalog**-Element eine (möglicherweise leere) Folge von **Buch**-Elementen enthält:

```
<!ELEMENT Katalog (Buch)*>
```

kein Vorkommen der eingeklammerten Kategorie oder beliebig viele hintereinander liegende

Einführung des **Buch**-Elements:

Ein **Buch**-Element soll (in dieser Reihenfolge) ein **Titel**-, **Autoren**-, **Verlag**-Element, optional ein **Seiten**-Element, ein **ISBN**-Element und ein oder mehrere **Preis**-Elemente enthalten.

```
<!ELEMENT Buch (Titel, Autoren, Verlag, Seiten?, ISBN, Preis+)>
```

ein oder kein Vorkommen des Seiten-Elements

ein oder mehrere Vorkommen des Preis-Elements

Einführung des **Autoren**-Elements:

Ein **Autoren**-Element soll ein oder mehrere **Autor**-Elemente enthalten.

```
<!ELEMENT Autoren (Autor+)>
```

Schließlich sollen das **Titel**-, **Autor**-, **Verlag**-, **Seiten**-, **ISBN**- und **Preis**-Element reinen Text (*parsed character data, PCDATA)* enthalten. Das wird durch die folgenden Deklarationen ausgedrückt:

```
<!ELEMENT Titel (#PCDATA)>
<!ELEMENT Autor (#PCDATA)>
<!ELEMENT Verlag (#PCDATA)>
<!ELEMENT Seiten (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT Preis (#PCDATA)>
```

Für das **Preis**-Element wird noch ein mit **Waehrung** bezeichnetes Attribut eingeführt, dem als Wert eine Zeichenkette (CDATA) zugeordnet sein soll. Weiter soll dieses Attribut in jedem *start tag* eines Preis-Elements gefordert sein.

```
<!ATTLIST Preis Waehrung CDATA #REQUIRED>
```

Gültiges Beispieldokument des Typs Katalog:

```
<?xml version="1.0"?>
<!DOCTYPE Katalog [
<!ELEMENT Katalog (Buch)*>
<!ELEMENT Buch (Titel, Autoren, Verlag, Seiten?, ISBN, Preis+)>
<!ELEMENT Autoren (Autor+)>
<!ELEMENT Titel (#PCDATA)>
<!ELEMENT Autor (#PCDATA)>
<!ELEMENT Verlag (#PCDATA)>
<!ELEMENT Seiten (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT Preis (#PCDATA)>
<!ATTLIST Preis Waehrung CDATA #REQUIRED>
]>
```

DTD

… Fortsetzung auf der nächsten Folie

```
…
<Katalog>
<Buch>
<Titel>XML in der Praxis</Titel>
<Autoren>
<Autor>Henning Behme</Autor><Autor>Stefan Mintert</Autor>
</Autoren>
<Verlag>Addison-Wesley</Verlag>
<ISBN >3-8273-1636-7</ISBN>
<Preis Waehrung="USD">52.25</Preis><Preis Waehrung=„ATS">700</Preis>
</Buch>
<Buch>
<Titel>Learning XML</Titel>
<Autoren><Autor>Erik T. Ray</Autor></Autoren>
<Verlag>O&apos;Reilly</Verlag>
<Seiten>356</Seiten>
<ISBN>0-596-00046-4</ISBN>
<Preis Waehrung="Euro">47</Preis>
</Buch>
</Katalog>
```