Freie Universität Berlin

Aufbau von XML-Dokumenten

Malgorzata Mochol
Freie Universität Berlin
Institut für Informatik
Netzbasierte Informationssysteme
mochol@inf.fu-berlin.de

Organisatorisches

News

- 1. Übung: XML-Syntax, Namensräume

  → heute (Mi.) 14:15 – 15:45, SR 005

  → Do. 10:15-11:45, SR 005

Freie Universität Berlin

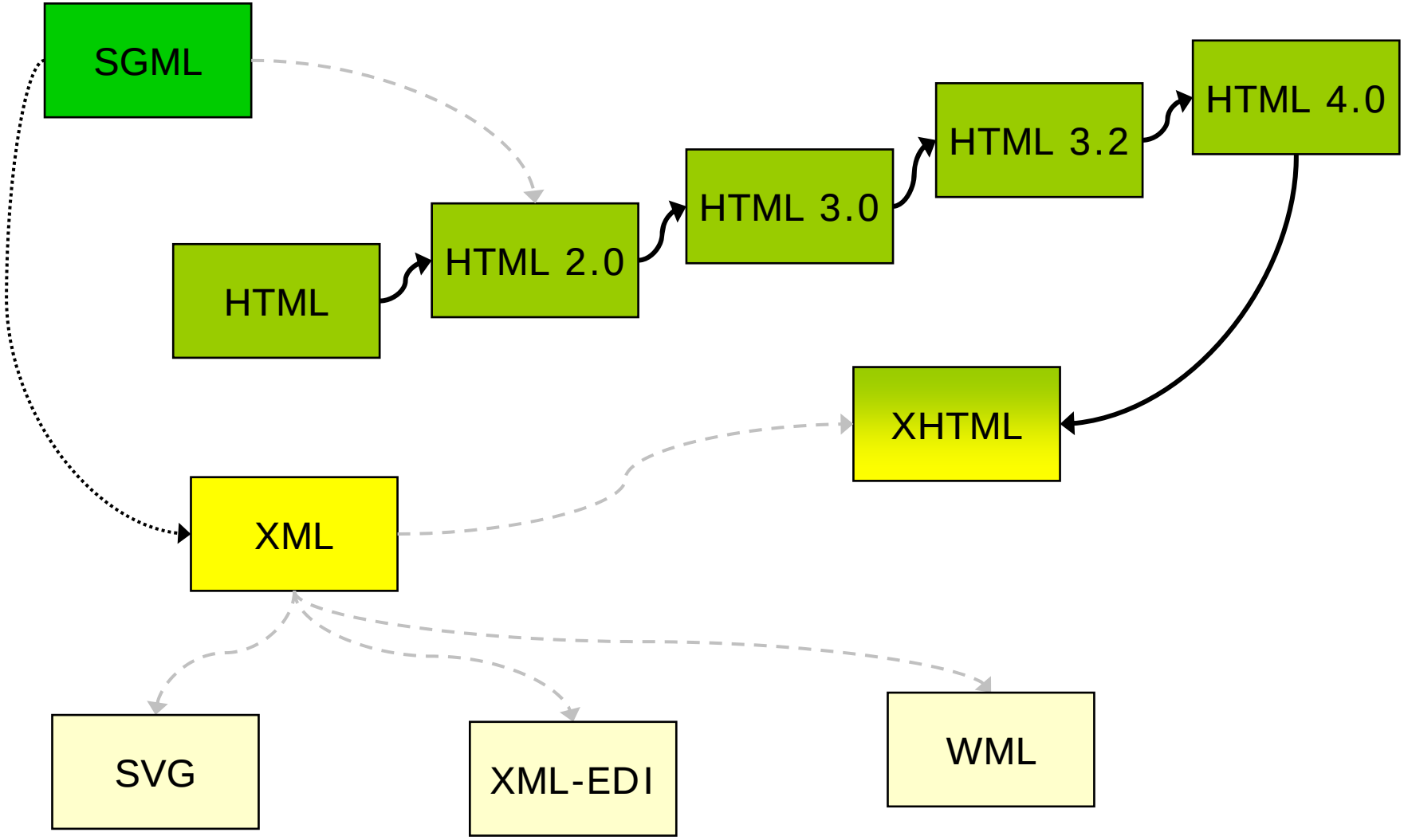
Aufbau von XML-Dokumenten

Heutige Vorlesung

Freie Universität Berlin

- Wiederholung: Was ist XML?
- XML-Syntax
  - Elemente
  - Attribute
  - Deklaration
  - ...
- Namensräume

SGML, HTML, XML, XHTML,...



Quelle: R.Tolksdorf: HTML & XHTML – die Sprachen des Webs. Informationen aufbereiten und präsentieren im Internet. dpunkt.verlag,5.Auflage, 2003

Wiederholung: Was ist XML?

Freie Universität Berlin

- XML ist eine Methode, um strukturierte Daten in einer Textdatei darzustellen.
- XML sieht ein wenig wie HTML aus.
- XML ist Text, aber nicht zum Lesen.
- XML ist eine Familie von Techniken.
- XML ist neu, aber nicht so neu.
- XML überführt HTML in XHTML
- XML ist modular
- XML ist lizenzfrei und plattformunabhängig

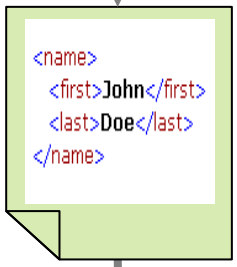
XML-Syntax: XML-Dokumente

Was ist ein XML-Dokument?

Inhalt: Text oder Daten

kodiert als

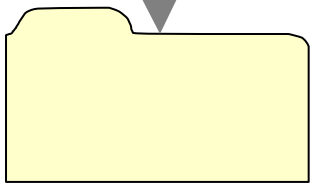
XML-Dokument



Objekt, das Syntaxregeln von XML entspricht (wohlgeformt ist)

gespeichert in

XML-Datei(en)



Grundbausteine von XML

- **Elemente**: strukturieren das XML-Dokument
- **Attribute**: Zusatzinformationen zu Elementen
- **XML-Deklaration**: Informationen für Parser

```
<?xml version="1.0" encoding="UTF-8"?>
<name id="1232345">
  <first>John</first>
  <middle>Fitzgerald Johansen</middle>
  <last>Doe</last>
</name>
```

- **Namensräume**: lösen Namenskonflikte auf und geben Elementen eine Bedeutung

Freie Universität Berlin

XML-Syntax
Grundbausteine: XML-Elemente

Grundbausteine: XML-Elemente

- Beispiel: <first>John</first>

- besteht aus:
  - einem Anfangs-Tag (engl. *start tag*)

    hier: *<first>*
  - einem dazugehörigen Ende-Tag (engl. *end tag*):

    hier: *</first>*
  - einem Inhalt:

    hier: *„John“*
  - alles zusammen bildet ein Element

- Elemente haben einen Namen: hier „first“

Inhalt von Elementen

Freie Universität Berlin

1. unstrukturierter Inhalt:
   - einfacher Text ohne Kind-Elemente
2. strukturierter Inhalt:
   - Sequenz von > Ø Kind-Elementen
3. gemischter Inhalt:
   - enthält Text mit mind. einem Kind-Element
4. leerer Inhalt

1. Unstrukturierter Inhalt

Freie Universität Berlin

- Beispiel: <first>John</first>
- einfacher Text **ohne Kind-Elemente**

  Kind-Element: Element, das im Inhalt eines Elementes vorkommt
- unstrukturierter Inhalt auch als *Parsed Character Data (PCDATA)* bezeichnet:
  - character data: einfache Zeichenkette
  - parsed: Zeichenkette wird vom Parser analysiert, um Ende-Tag zu identifizieren
  - Normalisierung: u.a. Zeilenumbruch (CR+LF) ➔ #xA

Anmerkung: Auf den Folien schreibe ich der besseren Lesbarkeit wegen *Kind-Element* statt *Kindelement* !

PCDATA

Freie Universität Berlin

- Reservierte Symbole < und & in PCDATA nicht erlaubt.
- Symbole wie /, (,), {, }, [,], % allerdings erlaubt
- statt < und & **Entity References** & bzw. < benutzen
- Entity References in XML:

&	⇨	&
<	⇨	<
>	⇨	>
'	⇨	'
"	⇨	"

CDATA

Freie Universität Berlin

- Unstrukturierten Inhalt mit vielen reservierten Symbolen besser als Character Data (CDATA) darstellen.
- Beispiel:

```
<formula>
  <![CDATA[ X < Y & Y < Z ]]>
</formula>
```

- Inhalt: String zwischen inneren Klammern [ ]
  hier: X < Y & Y < Z
- XML-Parser sucht in CDATA lediglich]]>, analysiert den Inhalt aber ansonsten nicht.
- “]]>“ als Inhalt von CDATA nicht erlaubt

Entity Reference und CDATA

```
<formula>
   X &lt; Y &amp; Y &lt; Z
</formula>
```

```
<formula>
   X < Y & Y < Z
</formula>
```

```
<formula>
   <![CDATA[ X < Y & Y < Z ]]>
</formula>
```

Freie Universität Berlin

- Beispiel:

```
<name>
        <first>John</first>
        <last>Doe</last>
</name>
```

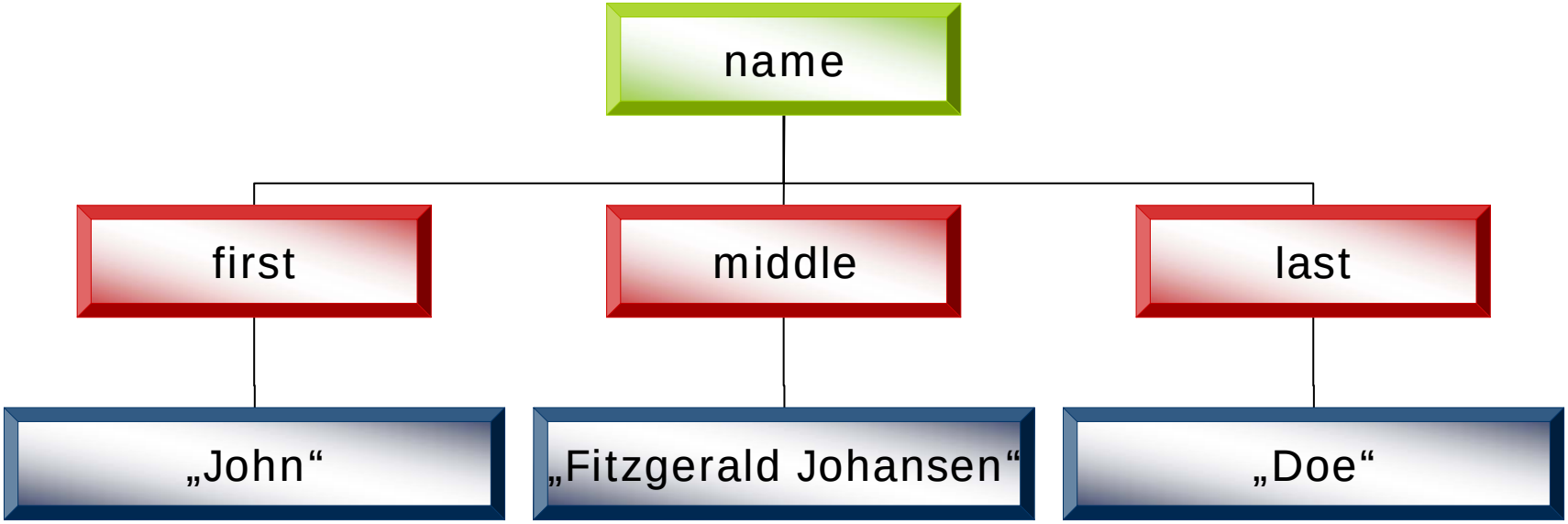
- Sequenz von **> 0 Kind-Elementen**:

  hier: *<first>John</first>* und *<last>Doe</last>*
- kein Text vor, nach oder zwischen den Kind-Elementen
- **Kind-Elemente immer geordnet**:

  Reihenfolge, so wie sie im XML-Dokument erscheinen
- Elemente können **beliebig tief geschachtelt** werden.

Baumstruktur von XML

```
<name>
   <first>John</first>
   <middle>Fitzgerald Johansen</middle>
   <last>Doe</last>
</name>
```



3. Gemischter Inhalt

Freie Universität Berlin

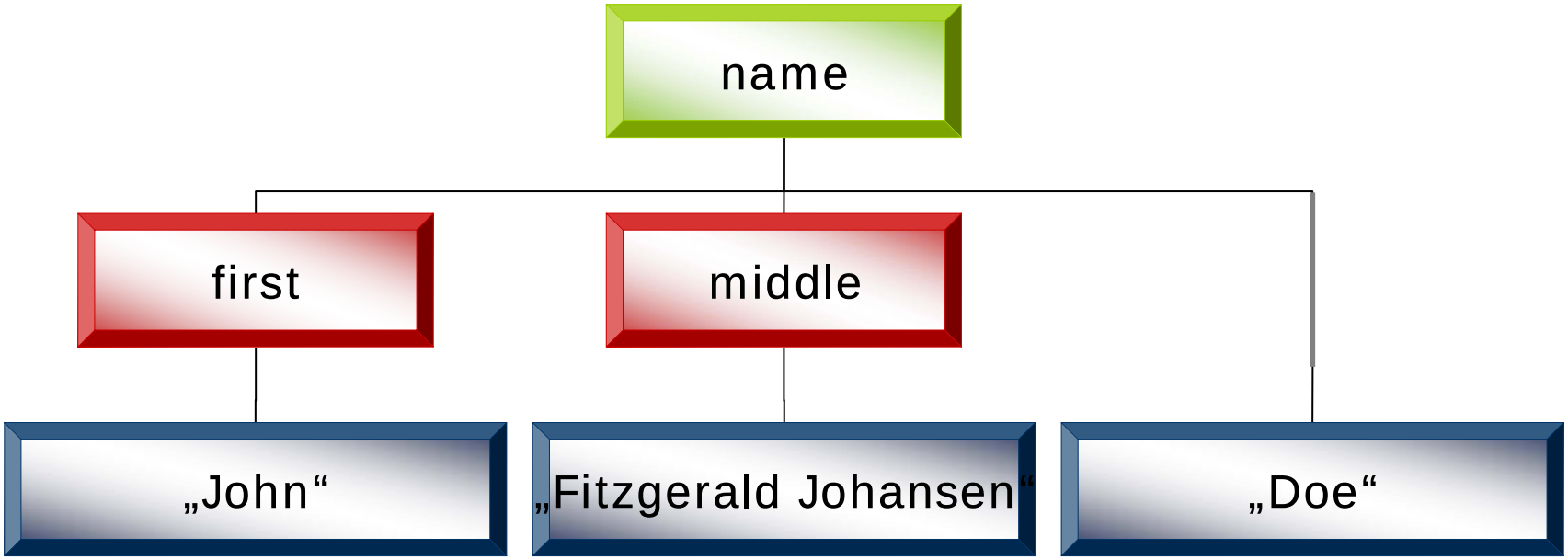
- Englisch: **mixed content**
- enthält Text mit mindestens einem Kind-Element
- Beispiel:

```
<section>
  Text
  <subsection> ... </subsection>
  Text
</section>
```

- Wie unterscheidet ein Parser strukturierten und gemischten Inhalt, wenn Text = leerer String?
- Antwort: Nur mit zugehöriger DTD oder XML-Schema möglich!

Baumstruktur von XML

```
<name>
   <first>John</first>
   <middle>Fitzgerald Johansen </middle>
   Doe
</name>
```



4. Leerer Inhalt

Freie Universität Berlin

- Beispiel:

```
<name>
   <first>John</first>
   <middle></middle>
   <last>Doe</last>
</name>
```

- weder Text noch Kind-Element
- <middle></middle> auch **leeres Element** genannt
- Abkürzung: **selbstschließendes Tag** (engl.: *self-closing tag*) <middle/> :

```
<name>
   <first>John</first>
   <middle/>
   <last>Doe</last>
</name>
```

Warum leere Elemente?

Freie Universität Berlin

```
<name>
   <first>John</first>
   <last>Doe</last>
</name>
```

vs.

```
<name>
   <first>John</first>
   <middle/>
   <last>Doe</last>
</name>
```

- leeres Element evtl. von DTD oder XML-Schema vorgeschrieben
- einfacher später mit Inhalt zu füllen
- leeres Element kann Attribute haben:

  `<middle status="unknown"></middle>` oder

  `<middle status="unknown"/>`

Freie Universität Berlin

XML-Syntax
Grundbausteine: XML-Attribute

Grundbausteine: XML-Attribute

Freie Universität Berlin

```
<name id="1232345" nickname="Shiny John">
  <first>John</first>
  <last>Doe</last>
</name>
```

- Attribut: Name-Wert-Paar
  - name="wert" oder name='wert' aber ~~name= "wert~~‘
- Attribut-Wert:
  - immer PCDATA: keine Kind-Elemente, kein < und &
  - "we"rt" und 'we'rt' ebenfalls nicht erlaubt
  - Normalisierung: u.a. Zeilenumbruch ➔ #xA
- Beachte: Reihenfolge der Attribute belanglos

Attribut statt Element

- Jedes Attribut auch als Kind-Element darstellbar:

```
<name id="12345">
  <first>John</first>
  <middle>Fitzgerald</middle>
  <last>Doe</last>
</name>
```

id als Attribut

```
<name>
  <id>12345</id>
  <first>John</first>
  <middle>Fitzgerald</middle>
  <last>Doe</last>
</name>
```

id als Kind-Element

Element statt Attribut

- Jedes Kind-Element mit **unstrukturiertem** Inhalt auch als Attribut darstellbar:

```
<name>
  <id>12345</id>
 <first>John</first>
 <middle>Fitzgerald</middle>
 <last>Doe</last>
</name>
```

→

```
<name id="12345"
  first="John"
  middle="Fitzgerald"
  last="Doe" />
```

id, first, middle und last als Kind-Elemente

id, first, middle und last als Attribute

Resultat: leeres Element !

Attribut oder Element?

Freie Universität Berlin

- Attribut kann nur einfachen String (PCDATA) als Wert haben, Element kann beliebig strukturiert sein
- <![CDATA[...]]> in Element-Inhalten erlaubt, nicht aber in Attribut-Werten
- Reihenfolge der Attribute belanglos, diejenige von Elementen nicht

<u>Fazit</u>: Attribute für einfache, unstrukturierte Zusatzinformationen (Metadaten) geeignet.

Beispiel

Freie Universität Berlin

```
<name creation-date="01.05.2008">
  <first>John</first>
  <middle>Fitzgerald Johansen</middle>
  <last>Doe</last>
</name>
```

- Erstellungsdatum creation-date ist eine Zusatzinformation
- falls noch andere Attribute vorhanden: Reihenfolge egal

⇨ Repräsentation als Attribut

- Nachteil: Datum "01.05.2008" unstrukturiert

Reservierte Attribute

Freie Universität Berlin

- xml:lang
  - Sprache des Inhalts
  - Beispiel: Übung 1

- xml:space
  - Leerräume im Inhalt
  - Beispiel: Übung 1

Freie Universität Berlin

XML-Syntax
Grundbausteine: XML-Deklaration

Grundbausteine: XML-Deklaration

Freie Universität Berlin

```
<?xml version="1.0" encoding="UTF-8"?>
<name id="1232345">
  <first>John</first>
  <middle>Fitzgerald Johansen</middle>
  <last>Doe</last>
</name>
```

- enthält Informationen für Parser: z.B. verwendete XML-Version und Kodierung
- wenn vorhanden, dann immer **am Anfang der Datei**
- in XML 1.0 optional, in XML 1.1 obligatorisch

XML-Deklaration

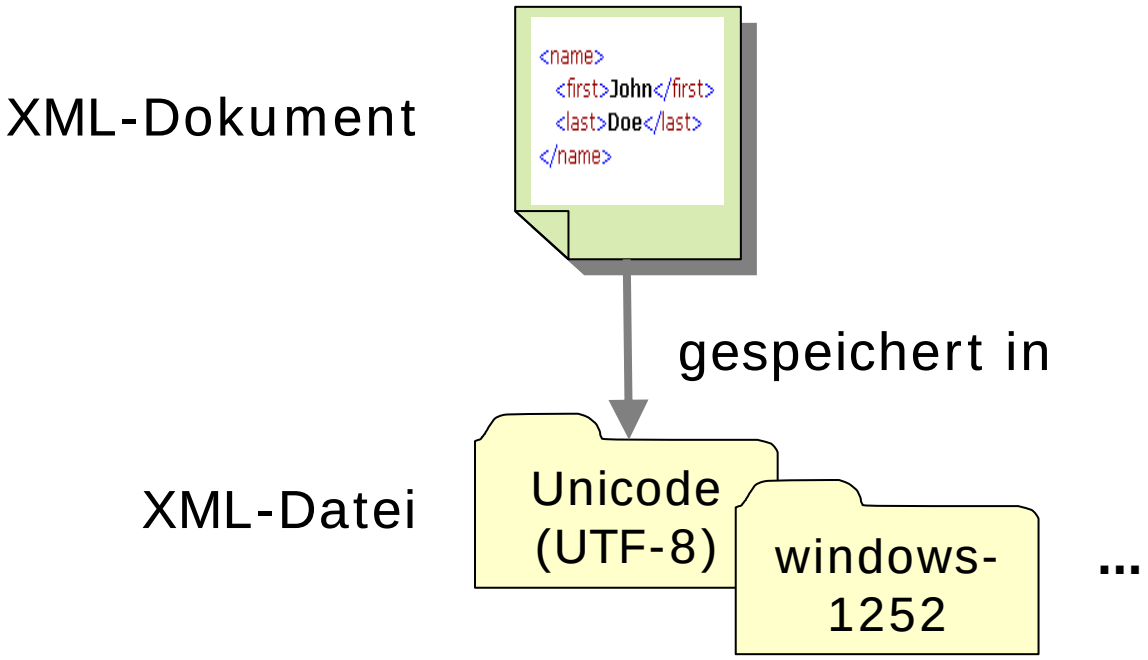
Freie Universität Berlin

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

- Attribut: version
  - verwendete XML-Version: "1.0" oder "1.1"
  - obligatorisch
- Attribut: encoding
  - Kodierung der XML-Datei
  - Optional
- Attribut: standalone
  - Gibt an, ob es eine zugehörige DTD oder ein XML-Schema gibt ("no" – default) oder nicht ("yes").
  - Optiona

Beachte: immer in dieser Reihenfolge!

XML-Deklaration: Kodierung



XML-Deklaration: Kodierung

Freie Universität Berlin

- XML-Parser
  - müssen intern mit Unicode (UTF-8 oder UTF-16) arbeiten
- Unicode
  - kann alle nationalen Zeichen darstellen: insgesamt ca. 65.000 Zeichen
- encoding-Attribut
  - Zeichenkodierung der XML-Datei
  - Fehlt das Attribut, dann wird Kodierung in Unicode angenommen.

Beachte: XML-Parser müssen nur Unicode verarbeiten können!

Freie Universität Berlin

XML-Syntax
Andere Grundbausteine

Weitere Grundbausteine von XML

Freie Universität Berlin

- Kommentare
  - <!-- Kommentar -->
  - -- in Kommentaren nicht erlaubt
- Prozessorinstruktionen
  - Beispiel: <?mysql SELECT * FROM PO?>
  - relativ selten benutzt

Freie Universität Berlin

XML-Syntax
Wohlgeformte XML-Dokumente

Wohlgeformte XML-Dokumente

Freie Universität Berlin

1. Jedes Anfangs-Tag muss ein zugehöriges Ende-Tag haben.
2. Elemente dürfen sich nicht überlappen.
3. XML-Dokumente haben genau ein Wurzel-Element.
4. Element-Namen müssen bestimmten Namenskonventionen entsprechen.
5. XML beachtet grundsätzlich Groß- und Kleinschreibung.
6. XML belässt White Space im Text.
7. Ein Element darf niemals zwei Attribute mit dem selben Namen haben.
8. Alle Attributwerte stehen in Anführungszeichen

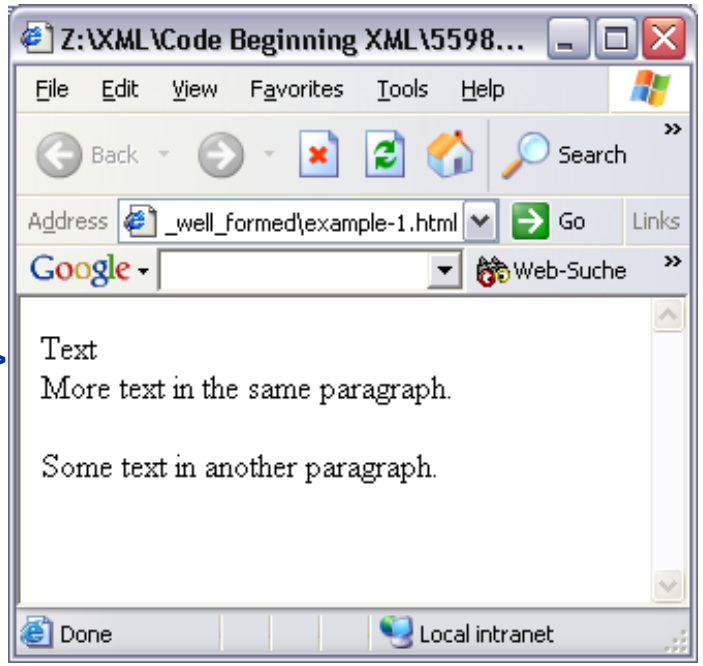
Wie kann aus den Grundbausteinen ein wohlgeformtes XML-Dokument gebildet werden?

Freie Universität Berlin

Jedes Anfangs-Tag muss zugehöriges Ende-Tag haben.

- In HTML gilt diese Regel nicht:

```
<HTML>
 <BODY>
   <P>Text
   <BR>More text in the same paragraph.
   <P>Some text in another paragraph.</P>
 </BODY>
</HTML>
```



- Wo endet das erste P-Element?

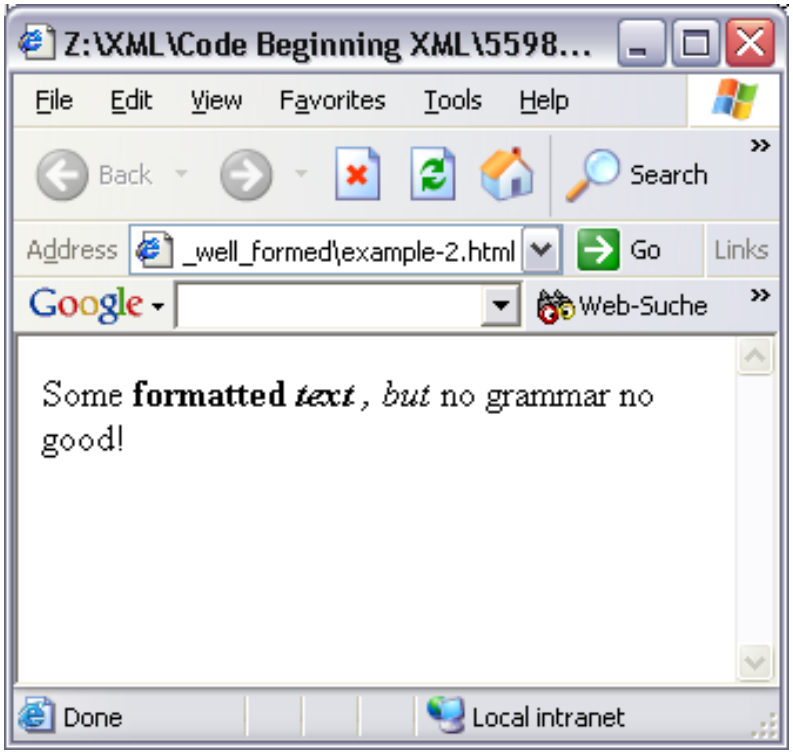
⇨ HTML mehrdeutig

Freie Universität Berlin

Elemente dürfen sich nicht überlappen.

- In HTML gilt diese Regel nicht:

```
<HTML>
 <BODY>
  <P>Some
   <STRONG>formatted
   <EM>text
   </STRONG>, but
   </EM>
   no grammar no good!
  </p>
 </BODY>
</HTML>
```



⇨ HTML unstrukturiert

Freie Universität Berlin

Jedes XML-Dokument hat genau ein Wurzel-Element.

- Also z.B. statt

```
<?xml version="1.0"?>
<name>John</name>
<name>Jane</name>
```

- zusätzliches Eltern-Element einführen:

```
<?xml version="1.0"?>
<names>
   <name>John</name>
   <name>Jane</name>
</names>
```

Freie Universität Berlin

Element- und Attribut-Namen:

- beginnen entweder mit Buchstaben oder _ aber nie mit Zahlen:

  z.B. first, First oder _First
- nach erstem Zeichen zusätzlich Zahlen sowie - und Punkt erlaubt:

  z.B. _1st-name oder _1st.name
- enthalten **keine Leerzeichen**
- enthalten **kein Doppelpunkt**
- beginnen **nicht** mit **xml**, unabhängig davon, ob die einzelnen Buchstaben groß- oder kleingeschrieben sind

Namenskonvention: Beispiele

Freie Universität Berlin

- <résumé> ✓
- <xml-tag> nicht korrekt: beginnt mit „xml“
- <123> nicht korrekt: beginnt mit Zahl
- <fun=xml> nicht korrekt: enthält „=“
  erlaubt wären: _, - und Punkt
- <first name> nicht korrekt: enthält Leerzeichen

Freie Universität Berlin

XML beachtet Groß- und Kleinschreibung.

- Im Gegensatz zu HTML unterscheidet XML also z.B. zwischen <P> und <p>.

Dennoch möglichst nicht gleichzeitig <First> und <first> verwenden!

Hinweis: eine Schriebweise im gesamten Dokument verwenden z.B. <FirstName>

Freie Universität Berlin

XML belässt White Space im Text.

- Beispiel:

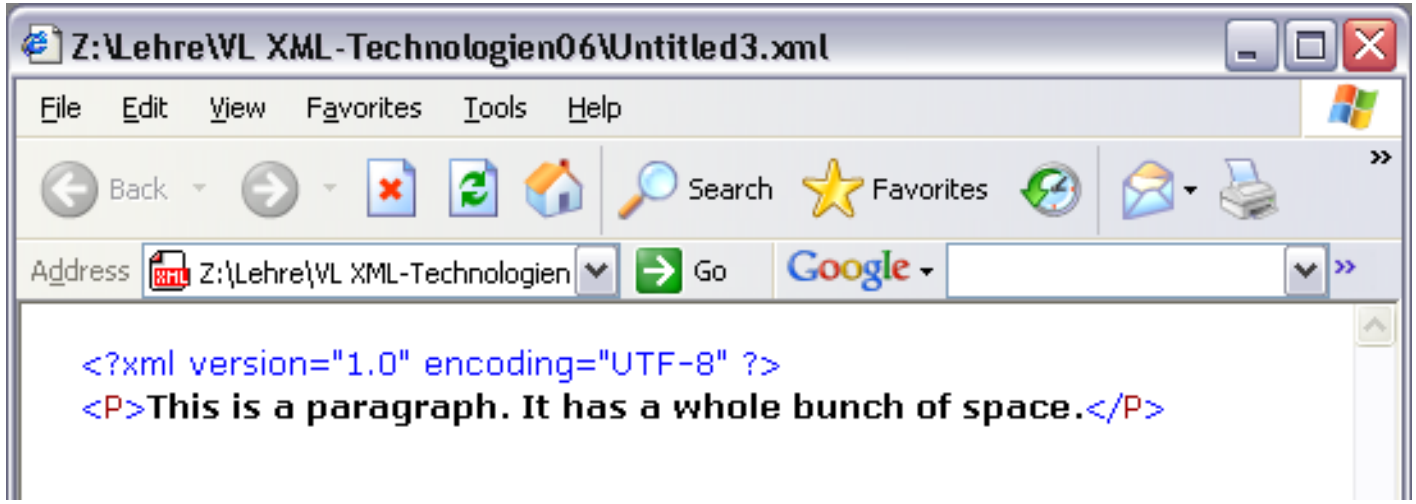
```
<?xml version="1.0" encoding="UTF-8">
<P>This is a paragraph.          It has a whole bunch
   of space.</P>
```

Inhalt des P-Elementes:

```
This is a paragraph.                It has a whole bunch
 of space.
```

White Space in Browsern

- Beachte: Von Browsern wird White Space allerdings nicht angezeigt:



- Grund:
  - XML-Dokumente werden zur Darstellung im Browser in HTML umgewandelt
  - HTML reduziert White Space auf ein Leerzeichen

Freie Universität Berlin

XML-Syntax
XML 1.0 vs. XML1.1

XML-Versionen

Freie Universität Berlin

	XML 1.0	XML 1.1
XML-Deklaration	optional	obligatorisch
Zeilenende (EoL)	LF, CR	LF, CR NEL (IBM-Mainframe)
Unicode in Namen	**exklusive Strategie**: alle Zeichen verboten, die nicht ausdrücklich erlaubt	**inklusive Strategie**: alle Zeichen erlaubt, die nicht ausdrücklich verboten

- ⇨ XML 1.1 interoperabel mit IBM-Mainframes
- ⇨ XML 1.1 aufwärtskompatibel bzgl. zukünftiger Unicode-Versionen
- ⇨ XML 1.1 nicht abwärtskompatibel!

Viel Lärm um XML 1.1!

Freie Universität Berlin

- XML 1.0, 3rd Edition zeitgleich mit XML 1.1 veröffentlicht
  - 3rd Edition = 2nd Edition + Errata zur 2nd Edition
- Änderungen von XML 1.1 nur für IBM-Mainframes und Nicht-ASCII-Zeichen relevant
- XML 1.1 verlangt von Parsern, dass beide Versionen erkannt werden:
  - wenn keine XML-Deklaration oder explizit Version 1.0:
    → Wohlgeformtheit gemäß XML 1.0
  - in allen anderen Fällen:
    → Wohlgeformtheit gemäß XML 1.1
- Mittlerweile:
  - XML 1.0, 4rd Edition
  - XML 1.1, 2nd Edition

Kleiner Online-Test zu XML

- W3 Schools: kostenlose Online-Tutorials zu XML-Technologien

⇨ Online-Test
http://www.w3schools.com/xml/xml_quiz.asp

XML-Editoren

- XML-Dokumente werden normalerweise mit speziellen Editoren erstellt und modifiziert.

- meistbenutzte XML-Editor: XMLSpy von Altova
  - Professional Edition als 30-tägige Testlizenz kostenlos

  ➔ www.xmlspy.com

- XML Plug-in für Eclipse
  - im PC-Pool vorhanden

Namensräume

Namenskonflikte

```
<course>
  <title>Semantic Web</title>
  <lecturers>
    <name>
      <title>Priv.-Doz. Dr. M.S.E</title>
      <first>Otto</first>
      <last>Müller</last>
    </name>
  </lecturers>
  <date>12/11/2004</date>
  <abstract>...</abstract>
</course>
```

- **Namenskonflikt**: gleicher Name, aber unterschiedliche Bedeutung
- z.B. Titel einer Veranstaltung vs. Titel einer Person
- in einem Dokument unterschiedliche Vokabularien

Auflösung durch Präfixe

```
<course:course>
  <course:title>Semantic Web</course:title>
  <course:lecturers>
    <pers:name>
      <pers:title>Priv.-Doz. Dr. M.S.E</pers:title>
      <pers:first>Otto</pers:first>
      <pers:last>Müller</pers:last>
    </pers:name>
  </course:lecturers>
  <course:date>12/11/2004</course:date>
  <course:abstract>...</course:abstract>
</course:course>
```

- Präfixe geben Kontext an: Aus welchem Bereich stammt der Name
  - z.B. pers:title vs. course:title
- ähnliches Vorgehen in Programmiersprachen:
  - z.B. java.applet.Applet

Namensräume

Freie Universität Berlin

course:course

course:title course:abstract

course:lecturers

course:date

pers:name

pers:title pers:first

pers:last

Namensraum (namespace):

- alle Bezeichner mit identischen Anwendungskontext
- Namensräume müssen eindeutig identifizierbar sein.

Namensräume in XML

- WWW: Namensräume müssen global eindeutig sein.
- In XML wird Namensraum mit URI identifiziert.
- Zuerst wird Präfix bestimmter Namensraum zugeordnet, z.B.:

- Anschließend kann das Namensraum-Präfix einem Namen vorangestellt werden: z.B. pers:title
- Beachte: Wahl des Präfixes egal!

Beispiel

```
<course:course xmlns:course="http://www.w3.org/2004/course">
   <course:title>Semantic Web</course:title>
   <course:lecturers>
      <pers:name xmlns:pers="http://www.w3.org/2004/pers">
         <pers:title>Priv.-Doz. Dr. M.S.E</pers:title>
         <pers:first>Steffen</pers:first>
         <pers:last>Staab</pers:last>
      </pers:name>
   </course:lecturers>
   <course:date>12/11/2004</course:date>
   <course:abstract>...</course:abstract>
</course:course>
```

Standard-Namensraum

Freie Universität Berlin

- **xmlns="URI"** statt xmlns:prefix="URI"
- Namensraum-Präfix kann weggelassen werden.

- Standard-Namensraum gilt für das Element, wo er definiert ist.
- Kind-Elemente erben Standard-Namensraum von ihrem Eltern-Element.
- Ausnahme: Standard-Namensraum wird überschrieben
- Beachte: Standardnamensräume gelten nicht für Attribute

Beispiel

```
<course:course xmlns:course="http://www.w3.org/2004/course">
  <course:title>Semantic Web</course:title>
  <course:lecturers>
    <pers:name xmlns:pers="http://www.w3.org/2004/pers">
      <pers:title>Priv.-Doz. Dr. M.S.E</pers:title>
      <pers:first>Otto</pers:first>
      <pers:last>Müller</pers:last>
    </pers:name>
  </course:lecturers>
  <course:date>12/11/2004</course:date>
  <course:abstract>...</course:abstract>
</course:course>
```

Beispiel

```
<course xmlns="http://www.w3.org/2004/course">
  <title>Semantic Web</title>
  <lecturers>
    <pers:name xmlns:pers="http://www.w3.org/2004/pers">
      <pers:title>Priv.-Doz. Dr. M.S.E</pers:title>
      <pers:first>Otto</pers:first>
      <pers:last>Müller</pers:last>
    </pers:name>
  </lecturers>
  <date>12/11/2004</date>
  <abstract>...</abstract>
</course>
```

Beispiel

```
<course xmlns="http://www.w3.org/2004/course">
  <title>Semantic Web</title>
  <lecturers>
    <name xmlns="http://www.w3.org/2004/pers">
      <title>Priv.-Doz. Dr. M.S.E</title>
      <first>Otto</first>
      <last>Müller</last>
    </name>
  </lecturers>
  <date>12/11/2004</date>
  <abstract>...</abstract>
</course>
```

Standard-Namensräume: Gültigkeit

```
<course xmlns="http://www.w3.org/2004/course">
  <title>Semantic Web</title>
  <lecturers>
    <name xmlns="http://www.w3.org/2004/pers">
      <title>Priv.-Doz. Dr. M.S.E</title>
      <first>Otto</first>
      <last>Müller</last>
    </name>
  </lecturers>
  <date>12/11/2004</date>
  <abstract>...</abstract>
</course>
```

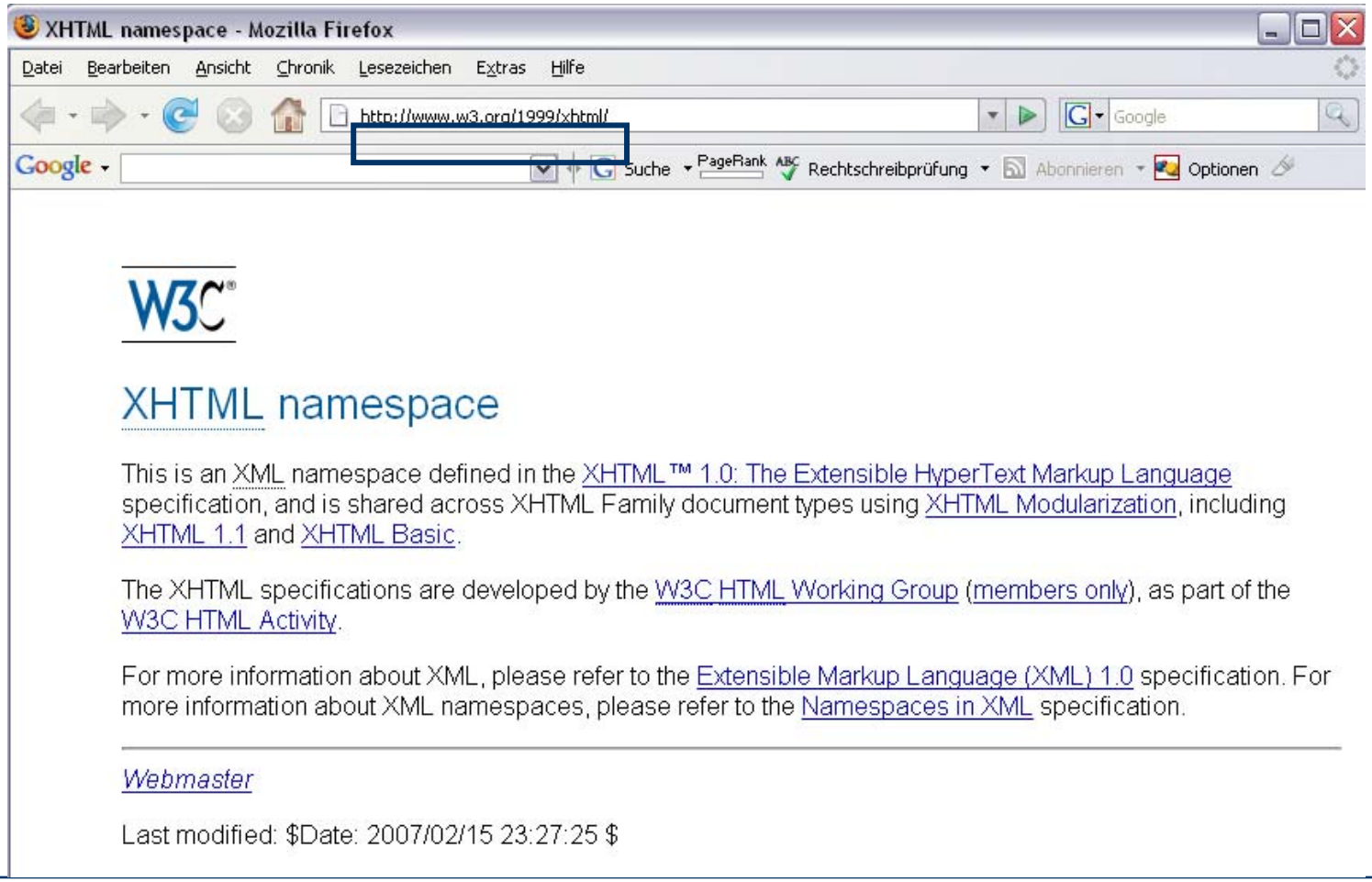
Exkurs: Uniform Resource Identifier

Freie Universität Berlin

- eindeutige Bezeichner für Ressourcen im WWW
- URI kann den physischen Aufenthaltsort einer Resource beschreiben
- Solche URIs werden auch **Uniform Resource Locators (URLs)** genannt.

URIs als Namensraum-Bezeichner

- Beispiel: http://www.w3.org/1999/xhtml bezeichnet den Namensraum für XHTML



URIs als Namensraum-Bezeichner

Freie Universität Berlin

- URI kann (muss aber nicht) Beschreibung des Namensraumes enthalten:
  - z.B. XML-Schema oder Spezifikationen
- URI muss nicht einmal existieren!
- Allerdings ist nur bei existierenden URIs Eindeutigkeit sichergestellt.

Beispiel

```
<Book xmlns="http://www.books-ns.org">
  ...
</Book>
```

- http://www.book-ns.org existiert (noch) nicht
- keine Fehlermeldung, keine Warnung von XML-Parser oder XML–Editor
- dennoch Eindeutigkeit nicht sichergestellt:

  jemand anderes kann gleiche URL für anderen Namensraum verwenden

⇨ neue Namensräume nur mit URIs bezeichnen, die man selbst besitzt

Qualified vs. Unqualified

Freie Universität Berlin

- Element- oder Attribut-Name heißt **namensraumeingeschränkt** (**qualified**), wenn er einem Namensraum zugeordnet ist.

- Einschränkung vom Element-Namensraum:
  1. Standard-Namensraum festlegen
  2. Namensraum-Präfix voranstellen

- Einschränkung vom Attribut-Namensraum:
  1. Namensraum-Präfix voranstellen

Was bedeutet … ?

- HTML:
  - Bedeutung festgelegt (p = Absatz)
- XML:
  - Bedeutung offen
  - kann aber mit Namensraum festlegt werden
  - Beispiel: p stammt aus dem Namensraum für XHTML.

xhtml Abkürzung für Namensraum

```
<xhtml:p xmlns:xhtml="http://www.w3.org/1999/xhtml">…</xhtml:p>
```

Namensraum:
u.a. p = Absatz

Und das war es schon?

Freie Universität Berlin

- Ja!
- Syntax wohlgeformter XML-Dokumente (fast) vollständig vorgestellt
- XML-Syntax also sehr einfach
- gleichzeitig ist XML beliebig erweiterbar
- Und das ist genau die Stärke von XML: einfach und flexibel!

Wie geht es weiter?

Freie Universität Berlin

- ☑ XML-Syntax
  - ☑ Elemente
  - ☑ Attribute
  - ☑ Deklaration
  - ☑ Wohlgeformtheit
- ☑ Namensräume

- Nächste Woche
  - Definition von XML-Sprachen mit DTDs und XML-Schema