<?xml?>

XML

Syntaktische Grundlagen für das Semantic Web

Sprachen des Semantic Web – XML

- e**X**tensible **M**arkup **L**anguage
- Offizielle Recommendation des W3C beschreibt
  - Maschinenlesbare Dokumente
  - Klasse von Objekten → XML Dokumente
  - Verhalten von Software, die XML Dokumente verarbeitet

Sprachen des Semantic Web – XML

- Markup-Sprachen:
  - Versehen von Text-Dokumenten mit zusätzlichen Informationen
  - Auszeichnen / Annotieren →Metadaten
- Tags (engl. Etikett / Schild) zeichnen Teile eines Dokuments aus
- Beispiel HTML:
  - Formatierung z.B. mit <i> (kursiv) und <b> (fettgedruckt)
    - `<i>Diese Vorlesung</i> hat das Thema <b>Semantic Web</b>.`
    - *Diese Vorlesung* hat das Thema **Semantic Web**.
  - HTML hat eine feste Menge von Tags (Vokabular)
  - Programme verwenden **genau** diese Tags

Sprachen des Semantic Web – XML

- XML verwendet wie HTML Tags

- HTML definiert Darstellung/Formatierung
- XML definiert die *logische Struktur* von Dokumenten

- Mit XML kann man (beliebige) Markup-Sprachen definieren
- Beispiel: XHTML = XML basiertes HTML

- Bei XML können die Tags selbst/frei definiert werden

Sprachen des Semantic Web – XML

- Beispiel:

```
<Vorlesung>Diese Vorlesung</Vorlesung>
behandelt das Thema
<Thema>Semantic Web Technologien</Thema>
```

Sprachen des Semantic Web – XML

- XML bietet eine einfache und universell einsetzbare Möglichkeit, Daten zu **speichern**, elektronisch zu **verbreiten** und zu **verarbeiten**.
- XML ist ein *universelles* Daten-Austauschformat
- Praktisch jedes Datenmodell kann nach XML serialisiert werden.

Sprachen des Semantic Web – XML

- Jede Programmiersprache bietet Konstrukte oder Bibliotheken zum Umgang mit XML
- Es gibt eine große Anzahl an XML-Anwendungen und XML Vokabularien
  - Die meisten neu eingeführten Dateiformate basieren auf XML
  - Beispiel: Neues MS Office-Dateiformat
- Bezug zum Semantic Web:
  - Syntax-Definitionen von RDF(S) und OWL

Sprachen des Semantic Web – XML

- Aufbau von XML-Dokumenten
  - XML-Deklaration:

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

  - Optional: DOCTYPE-Definition
  - Öffnen des XML-Wurzelelement (bei (X)HTML: `<html>`)
    - Inhalt
  - Schließen des XML-Wurzelelements (bei (X)HTML: `</html>`)

Sprachen des Semantic Web – XML

- XML-Elemente
  - Beginnen mit einem **Start-Tag**
  - Enden mit einem **End-Tag**
  - Tags sind ***XML!-Namen***, welche in spitze Klammern eingeschlossen sind
  - Alles zwischen Start-Tag und End-Tag nennt man **Inhalt**
  - Beispiel:

```
<Person>Steffen Schloenvoigt</Person>
```

Sprachen des Semantic Web – XML

- XML-Elemente können Daten enthalten, aber auch andere Elemente
- Beliebig tiefe Verschachtelung der Elemente möglich
- XML-Elemente ohne Inhalt können zumeist mit selbstschließenden Tags gekennzeichnet werden
  - Beispiel:

    `<Person></Person> = <Person />`

Sprachen des Semantic Web – XML

- Syntaktisches zu XML:
  - Jedes XML-Dokument hat **genau ein** Wurzelelement
  - Alles andere (weitere Elemente, Texte, Daten) muss sich innerhalb dieses Wurzelelements befinden!
  - Jedes geöffnete Element muss wieder geschlossen werden! (Ausnahme: selbstschließende Elemente)

Sprachen des Semantic Web – XML

- Syntaktisches zu XML-Namen:
  - Müssen mit einem Buchstaben oder mit _ beginnen
  - Dürfen neben Buchstaben u.a. auch Zahlen enthalten
  - Keine Längenbeschränkung
  - Unterscheidung zwischen Groß-/ Kleinschreibung:

    `<Person>` ‡ `<person>` ‡ `<PERSON>`

  - **Empfehlung:** XML-Namen immer klein schreiben
  - Umlaute **nicht** verwenden !

Sprachen des Semantic Web – XML

- XML-Attribute
  - Angabe von Eigenschaften bzw. zusätzlichen Informationen zu einem XML-Element
  - Verwendung innerhalb von Start-Tags oder selbstschließenden Elementen (nicht in End-Tags)
  - Namen der Attribute sind ebenfalls XML-Namen

    → selbe syntaktische Einschränkungen
  - Beispiel:

    ```
    <Person status="Lehrbeauftragter">Steffen Schloenvoigt</Person>
    ```
- Spezielles Attribut `xml:lang`
  - Vordefiniert in XML
  - Spezifiziert die Sprache eines Elements
  - Werte sind die üblichen Ländercodes (de, en, fr, it, …)

Sprachen des Semantic Web – XML

- Beispiel:

```
<?xml version="1.0" encoding="utf-8"?>
<Personen>
   <Person status="Lehrbeauftrager">
      <Name>Steffen Schloenvoigt</Name>
      <Organisation xml:lang="de">
         Hochschule Konstanz (HTWG)
      </Organisation>
      <Organisation xml:lang="en">
         University of Applied Sciences, Constance
      </Organisation>
   </Person>
</Personen>
```

Sprachen des Semantic Web – XML

- Übung – Wo sind die Fehler?

```
<Liste laenge=”4”>
    <ListenEintrag pos=”1”>
        <Element/>Lorem ipsum<element/>
    </Listeneintrag>
    <listeneintrag pos=”3”/>
</Liste>
```

Sprachen des Semantic Web – XML

- Übung – Wo sind die Fehler?

```
<Liste laenge="4">
    <ListenEintrag pos="1">
        <Element/>Lorem ipsum<element/>
    </Listeneintrag>
    <listeneintrag pos="3"/>
</Liste>
```

Sprachen des Semantic Web – XML

- Daten als Elemente oder als Attribut speichern?
  - Oft unklar, was besser ist

```
<Person name="Steffen Schloenvoigt" />
<Person>Steffen Schloenvoigt</Person>
```

  - Faustregel:
    - Falls Daten bereits eine eigene Struktur aufweisen: Verschachtelte Element nehmen!
  - Attribute besitzen **keinerlei** Struktur
  - Attributen kann **nichts hinzugefügt** werden
  - Attribute am besten dann nehmen, wenn etwas *über* ein Element ausgesagt werden soll
  - Attribut-Namen gelten immer nur für das Element, in dem sie definiert sind

Sprachen des Semantic Web – XML

- **Wohlgeformtheit** von XML:
  - Syntaktisch richtiges XML wird als *wellformed/wohlgeformt* bezeichnet. Dazu gehört:
    - XML-Deklaration
    - Wurzelelement
    - Korrekte Verschachtelung
    - Korrekte Benamung
  - Alle XML-Dokumente **müssen** wohlgeformt sein, damit sie von Software problemlos verarbeitet werden können

Sprachen des Semantic Web – XML

- Übung:

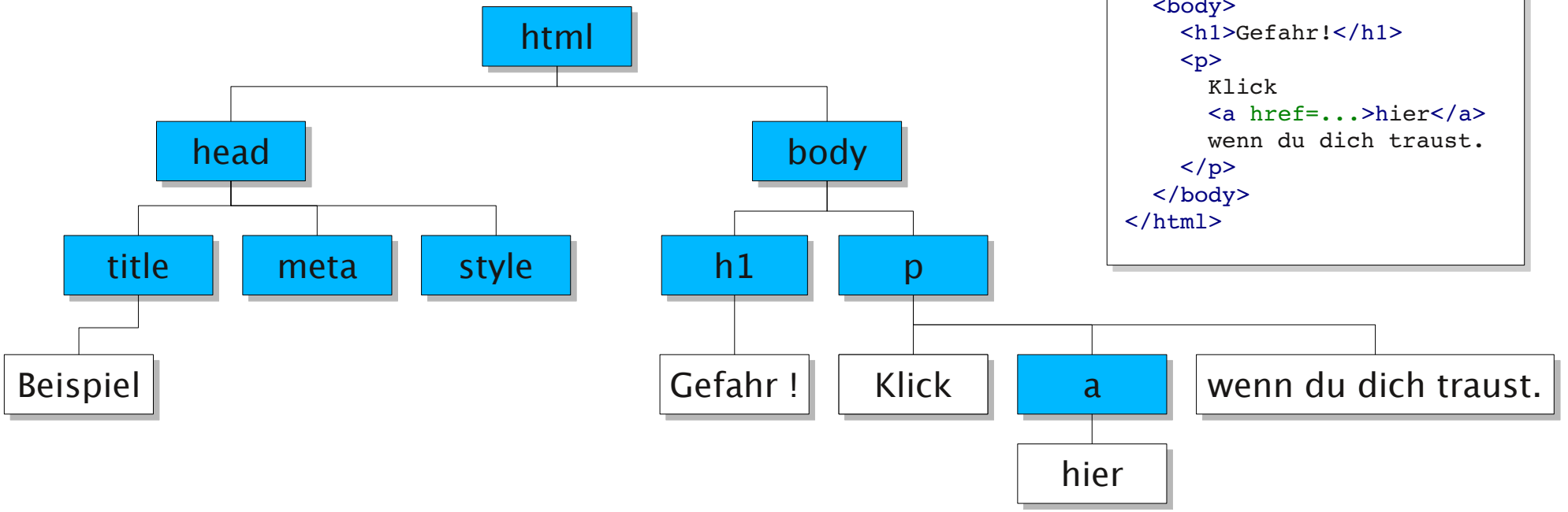
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC
"-//W3C/DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1/transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Kontrollieren Sie Ihr HTML!</title>
</head>
<body>
  <h3>HTML-Check des W3C</h3>
  <b>Kostenloser HTML-Check:<br/>
  <i>http://validator.w3.org/</b></i>
</body>
</html>
```

- Ist dieses Dokument wohlgeformt?

Sprachen des Semantic Web – XML

- Datenmodell von XML: Baumstruktur
- Jeder Tag entspricht einem benamten Knoten
- Jedes verschachtelte Tag entspricht einem Kindknoten
- Beispiel DOM-Baum einer HTML Datei:

```
<html>
  <head>
    <title>
      Beispiel
    </title>
    <meta name="... />
    <style type=... />
  </head>
  <body>
    <h1>Gefahr!</h1>
    <p>
      Klick
      <a href=...>hier</a>
      wenn du dich traust.
    </p>
  </body>
</html>
```



Sprachen des Semantic Web – XML

- Wie kann man die Struktur / das Vokabular eines XML Dokuments vorgeben?

→ DOCTYPE Definition

  - Einfache Möglichkeit, eine kontextfreie Grammatik in Backus-Naur-Form zu definieren

→ XML-Schema Dokumente

  - XML-basiert, neuer und mehr Möglichkeiten als bei DTDs

Sprachen des Semantic Web – XML

- DOCTYPE Definition:
  - Welche Elemente und Attribute sind überhaupt erlaubt
  - Wie dürfen Elemente ineinander verschachtelt werden
  - Welche Attribute sind für welches Element zulässig

- Überprüfung durch spezielle Parser möglich

  → validierender XML-Parser
  - Beispielsweise: W3C (X)HTML Validator
    - http://validator.w3.org/
    - Prüft neben Wohlgeformtheit auch korrekte Verschachtelung

Sprachen des Semantic Web – XML

- Beispiel DTD:

```
<!ELEMENT Personen (Person+)>
<!ELEMENT Person (Name?, Organisation*)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Organisation (#PCDATA)>

<!ATTLIST Person status CDATA #REQUIRED>
<!ATTLIST Organisation xml:lang CDATA “de”>
```

- Modifikatoren:

Modifikator	Auftreten der Kindelemente
+	Mindestens einmal
*	Keinmal oder beliebig oft
?	Keinmal oder genau einmal

Sprachen des Semantic Web – XML

- Beispiel DTD:
  -
```
<!ELEMENT Personen (Person+)>
<!ELEMENT Person (Name?, Organisation*)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Organisation (#PCDATA)>

<!ATTLIST Person status CDATA #REQUIRED>
<!ATTLIST Organisation xml:lang CDATA “de”>
```

- PCDATA: “Parsable Character DATA”
  - Textueller Inhalt
- CDATA: Character Data
  - Beliebige Zeichenketten

Sprachen des Semantic Web – XML

- Beispiel DTD:
  -
```
<!ELEMENT Personen (Person+)>
<!ELEMENT Person (Name?, Organisation*)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Organisation (#PCDATA)>

<!ATTLIST Person status CDATA #REQUIRED>
<!ATTLIST Organisation xml:lang CDATA “de”>
```

- Zusätze zu Attributen:
  - #REQUIRED: Attribut muss unbedingt angegeben werden
  - #FIXED “...”: Feste Vorgabe eines Attribut-Werts
  - #IMPLIED Attribut muss keinen Wert besitzen

Sprachen des Semantic Web – XML

- XML-Entitäten
  - Definition von Abkürzungen, Sonderzeichen
  - Wird durch Name- / Wertpaar spezifiziert
  - `&name;` referenziert den Wert
  - Beispiel:
    - `<!ENTITY htwg "Hochschule Konstanz – Hochschule für Technik, Wirtschaft und Gestaltung" >`
    - Im XML-Dokument kann nun **`&htwg;`** verwendet werden, um den langen Text zu referenzieren.
  - Oft werden auch Sonderzeichen auf diese Weise "escaped"

Entity	Wert
< bzw >	< bzw >
	Leerzeichen
ß	ß
©	©

Sprachen des Semantic Web – XML

- Einbindung von DTDs:
  - Direkt im XML-Dokument nach der XML-Deklaration
  - Form:
    - <!DOCTYPE WurzelElement [...]>
    - Statt "..." die Definition einfügen
  - Alternativ:
    - <!DOCTYPE WurzelElement SYSTEM "definitionfile.dtd">
    - Auslagerung der Definition in separate Datei
      Vorteil: Wiederverwendbar

Sprachen des Semantic Web – XML

XML-Schema-Definition (XSD)

- Soll DTDs mittelfristig ersetzen
- Ähnliche Grundfunktionalitäten wie DTDs
- Viel zusätzliche Funktionalität. Beispielsweise:
  - Angabe der **exakten Anzahl** von erlaubten verschachtelten Elementen (statt +,*,?)
  - **Gruppierung** von Elementen, sodass immer nur eines dieser Elemente auftreten darf
  - XML-Schema-**Datentypen** (integer, string, date …) nutzen und erweitern
  - Einbindung externer Definition → **Modularisierung**
- XML-Schema-Definitionen werden selbst in XML verfasst!!!

  → Infrastuktur für die Verarbeitung von XML kann genutzt werden

Sprachen des Semantic Web – XML

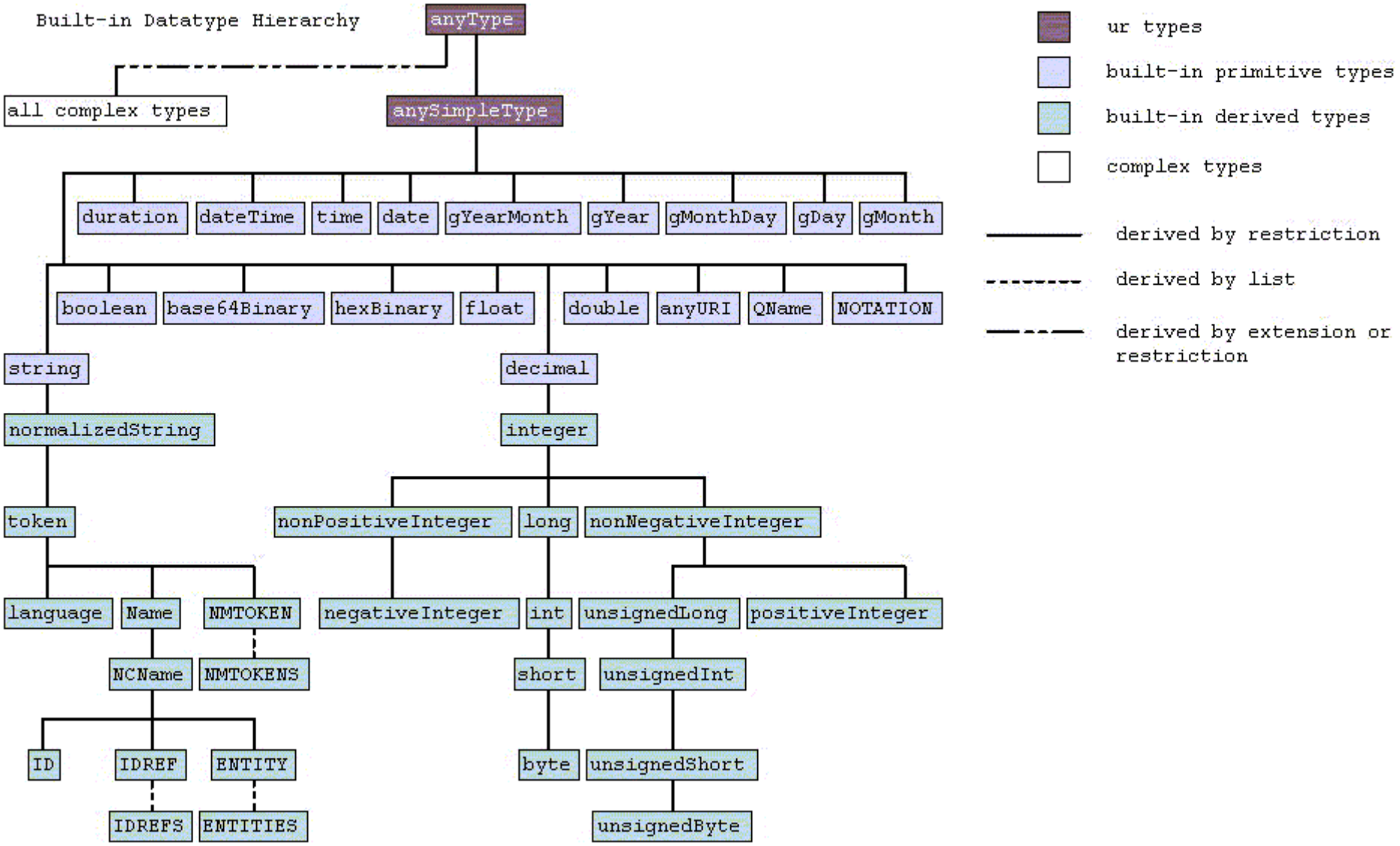
- Beispiel einer XML-Schema-Definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.example.com/api/schema/contact/1.0"
        xmlns="http://www.w3.org/2001/XMLSchema"
        xmlns:tns="http://www.example.com/api/schema/contact/1.0"
        xmlns:general="http://www.example.com/api/schema/general/1.0"
        elementFormDefault="qualified" attributeFormDefault="unqualified">
  <import namespace="http://www.example.com/api/schema/general/1.0"
          schemaLocation="general.xsd"/>

  <simpleType name="ContactMethodType">
    <restriction base="string">
      <enumeration value="Telefon"/>
      <enumeration value="Mobil"/>
      <enumeration value="EMail"/>
      <enumeration value="Brief"/>
    </restriction>
  </simpleType>

  <complexType name="ContactFormRequest">
    <annotation><documentation>Daten eines Kontaktforumlars</documentation></annotation>
    <sequence>
      <element name="salutation" type="string">
        <annotation><documentation>Anrede</documentation></annotation>
      </element>
      <element name="firstName" type="string" minOccurs="0">
        <annotation><documentation>Vorname</documentation></annotation>
      </element>
      <element name="lastName" type="string">
        <annotation><documentation>Nachname</documentation></annotation>
      </element>
      ...
      <element name="preferredContact" type="tns:ContactMethodType">
        <annotation><documentation>Bevorzugte Kontaktmethode</documentation></annotation>
      </element>
      <element name="Address" type="general:Address" minOccurs="0">
        <annotation><documentation>Adresse</documentation></annotation>
      </element>
    </sequence>
  </complexType>
</schema>
```

XSD-Datentypen



Sprachen des Semantic Web – XML

- Übung:

```
<!DOCTYPE kochbuch [
<!ELEMENT kochbuch (rezept)+ >
<!ELEMENT rezept (rezepttyp, titel, zutat+, arbeitsschritt+) >
<!ELEMENT titel (#PCDATA) >
<!ELEMENT zutat (#PCDATA) >
<!ELEMENT arbeitsschritt (#PCDATA) >
<!ATTLIST arbeitsschritt nummer CDATA #REQUIRED >
<!ELEMENT rezepttyp EMPTY >
<!ATTLIST rezepttyp name CDATA #REQUIRED >
]>
```

Sprachen des Semantic Web – XML

- Übung:

```
<!DOCTYPE kochbuch [
<!ELEMENT kochbuch (rezept)+ >
<!ELEMENT rezept (rezepttyp, titel, zutat+, arbeitsschritt+) >
<!ELEMENT titel (#PCDATA) >
<!ELEMENT zutat (#PCDATA) >
<!ELEMENT arbeitsschritt (#PCDATA) >
<!ATTLIST arbeitsschritt nummer CDATA #REQUIRED >
<!ELEMENT rezepttyp EMPTY >
<!ATTLIST rezepttyp name CDATA #REQUIRED >
]>
```

```
<?xml version="1.0" encoding="utf-8"?>
<kochbuch>
  <rezept>
    <rezepttyp name="Torten" />
    <titel>Schwarzwälder Kirschtorte</titel>
    <arbeitsschritt nummer="19">
      Schokostreusel draufmachen
    </arbeitsschritt>
    <zutat>Sahne</zutat>
    <zutat>Schokostreusel</zutat>
  </rezept>
</kochbuch>
```

Sprachen des Semantic Web – XML

- Modularisierung von XML
  - Oft sollen XML-Namen wiederverwendet werden
    → Einbindung externer Vokabularien erwünscht
  - Vermischung von Vokabularien kann zu Überschneidungen führen
    → Namenskonflikte
  - Namenskonflikte entstehen meist durch Mehrdeutigkeit

Sprachen des Semantic Web – XML

- Beispiel:

```
<?xml version="1.0" encoding="utf-8">
<Buch>
   <Titel>Semantic Web Grundlagen</Titel>
   <Autor>
      <Name>Pascal Hitzler</Name>
      <Titel>Dr.</Titel>
   </Autor>
   <Autor>
      <Name>York Sure</Name>
      <Titel>Dr.</Titel>
   </Autor>
</Buch>
```

Sprachen des Semantic Web – XML

- Beispiel:

```
<?xml version="1.0" encoding="utf-8">
<Buch>
   <Titel>Semantic Web Grundlagen</Titel>
   <Autor>
      <Name>Pascal Hitzler</Name>
      <Titel>Dr.</Titel>
   </Autor>
   <Autor>
      <Name>York Sure</Name>
      <Titel>Dr.</Titel>
   </Autor>
</Buch>
```

Sprachen des Semantic Web – XML

- Lösung:
  - XML-Namensräume = Namespaces
  - XML-Namespaces sind vom W3C standardisiert
  - XML-Namen können in sinnvolle Unterräume aufgeteilt/modularisiert werden
  - XML-Namen werden durch **URIs**
    - Eindeutig und
    - Einheitlich

    identifiziert
  - Dadurch (Wieder-/Weiter-)Verwendung in beliebigen XML-Dokumenten möglich

Sprachen des Semantic Web – XML

- URIs – Uniform Resource Identifiers
  - Einheitlicher Bezeichner für Ressourcen
  - Verwendung im WWW, um Webseiten oder andere Dateien zu bezeichnen. Aber auch für Webdienste oder E-Mail.
  - URIs sind aber nicht abhängig vom WWW
  - Ursprünglich Trennung in
    - URLs (Uniform Resource Locators) zur Identifikation von Web-adressierbaren Ressourcen – also alles, was per HTTP oder FTP übertragen werden kann.
    - URNs (Uniform Resource Names), um Ressourcen mittels einem vorhandenem oder frei zu vergebenden Namens zu identifizieren. Ursprünglich für weltweit eindeutige Bezeichner, wie etwa ISBNs
  - Heute wird eigentlich nicht mehr unterschieden
  - Wichtig: Eine URI muss kein Dokument referenzieren! (Kann aber)

Sprachen des Semantic Web – XML

- Syntax von URIs:
  - Allgemein:

```
Schema:FolgenderTeil
```

    - Oft verwendete Schemata: http, ftp, mailto
    - Beispiele:

```
http://example.com/blog/1
ftp://ftp.example.com/pub/rfc/rfc1808.txt
mailto:steffen@schloenvoigt.de
isbn:978-3-540-33993-9
```

Sprachen des Semantic Web – XML

- Syntax von URIs
  - Viele Schemata besitzen einen hierarchischen Aufbau
  - Insbesondere auch http und ftp
  - Syntax:

    schema://[login[:password]@]**server**[:port]/[path][?query]

    - Optionale Login-Parameter
    - Optional TCP Port (per Def. wird der Standarport des Schema verwendet – HTTP:80, FTP:21 …)
    - Pfad innerhalb der Server-Domain
    - Mit Query können mehrere Anfrage-Parameter übergeben werden. Aufbau von Query: param1=value1&param2=value2...

Sprachen des Semantic Web – XML

- Syntax von URIs – Fragment
  - Getrennt durch # kann an URIs ein sogenanntes Fragement angehängt werden:

    schema://server/[path][?query][**#fragment**]

  - Das Fragment kann auf eine Stelle **innerhalb** des Antwortdokuments einer Anfrage referenzieren
  - Der Browser schneidet in der Regel das Fragment ab, bevor er die Anfrage an den Server sendet
  - Im zurückgegebenen Dokument wird an die mit dem Fragment gekennzeichnete Stelle gesprungen
  - URIs mit Fragmenten nennt man URI-Referenz
- URI-Referenzen werden im Semantic Web häufig zur Identifizierung von Ressourcen verwendet.
- URIs können auch relativ zum aktuellen Dokument angegeben werden → Schema, Server usw. entfallen

XML-Namespaces

- Setzen sich zusammen aus einer Menge von XML-Namen, die in einem XML Dokument verwendet werden
- Sie werden folgendermaßen spezifiziert:

```
<ElementName xmlns[:prefix]="URI" >
```

- Angabe erfolgt in einem Start-Tag und ist dann innerhalb dieses Elements gültig

XML-Namespaces

- Meistens werden Namespaces im Wurzel-Element angegeben und sind damit für den Rest des Dokuments gültig
- Optional ist die Angabe eines Präfixes möglich
  - Kann im Geltungsbereich als Abkürzung verwendet werden
  - Wenn kein Präfix → Default Namespace (Es kann nur einen geben!)
- Attribute können nicht mit Namensräumen modularisiert werden
  - Aber: Ihr Name gilt lediglich lokal für das definierende Element. Sie können deswegen mehrfach in einem Dokument verwendet werden, ohne dass es zu Namenskonflikten kommt

Sprachen des Semantic Web – XML

- Beispiel (aus Buch Semantic Web Grundlagen)

```
<?xml version="1.0" encoding="utf-8" ?>
<Buch xmlns="http://www.semanticweb-grundlagen.de/"
      xmlns:aifb="http://www.aifb.uni-karlsruhe.de/">
  <Titel>Semantic Web Grundlagen</Titel>
  <aifb:Autor>
    <aifb:Name>Pascal Hitzler</aifb:Name>
    <aifb:Titel>Dr.</aifb:Titel>
  </aifb:Autor>
  <aifb:Autor>
    <aifb:Name>York Sure</aifb:Name>
    <aifb:Titel>Dr.</aifb:Titel>
  </aifb:Autor>
</Buch>
```

  - Titel ist nun sauber getrennt
  - Die beiden Versionen von Titel lauten nun eigentlich:
    http://www.semanticweb-grundlagen.de/Titel
    http://www.aifb.uni-karlsruhe.de/Titel

Sprachen des Semantic Web – XML

- Weitere Techniken im Zusammenhang mit XML:
  - XSLT: **Transformation** von XML-Dokumenten in beliebige Ausgabeformate
  - XPATH: **Referenzierung** von Stellen in einem XML-Dokument über die Angabe der auf dem Weg zur Stelle liegenden Elemente
  - XQuery: **Anfragesprache**, mit der es möglich ist, Daten aus XML-Dokumenten zu extrahieren
- Dazu gibt es Unmengen an Software, die XML lesen, verarbeiten, ausgeben, transformieren usw. kann.
- Kurz: XML ist eine standardisierte und weitverbreitete Meta-Sprache, die von Maschinen lesbar ist.

Sprachen des Semantic Web – XML

- Warum reicht XML für das SW dann nicht aus?
  - XML-Tags sind nicht viel besser als natürliche Sprache
    - Zumindest aus Sicht des Semantic Web
  - XML-Tags sind einfach nur Wörter
    - Können mehrdeutig sein
    - Beziehung zueinander ist nicht eindeutig definiert
  - Für Menschen ergibt sich meist eine Bedeutung
  - Für Maschinen bleiben Tags ohne Semantik
  - Es fehlt die Möglichkeit, die Bedeutung von Tags auf eine Art zu codieren, die Verarbeitung durch Maschinen ermöglicht.
- XML dient primär als syntaktische Grundlage zur Definition der weiteren Sprachen des Semantic Web

Sprachen des Semantic Web – XML

Noch Fragen ?

Sprachen des Semantic Web – XML

- Literatur:
  - Buch “Semantic Web Grundlagen”, Springer Verlag 2008
    Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, York Sure
    ISBN: 978-3-540-33993-9
  - Extensible Markup Language (XML) 1.0 (third edition)
    F. Yergeau, T. Bray, J. Paoli, C.M.Sperberg-McQueen, E. Maler
    W3C Recommendation, 4th February 2004
    http://www.w3.org/TR/REC-xml
  - Namespaces in XML
    T. Bray, D. Hollander, A. Layman, R. Tobin
    W3C Recommendation, 4th February 2004
    http://www.w3.org/TR/xml-names11
  - XML Schema Part 0: Primer Second Edition
    D.C. Fallside und P. Walmsley
    W3C Recommendation, 28th October 2004
    http://www.w3.org/TR/xmlschema-0/