Tutorial zu Kapitel WT:III

III. Dokumentsprachen

- HTML
- CSS
- XML-Grundlagen
  - XML-Dokumentstruktur
  - Document Type Definition (DTD)
  - Namespaces
- XML-Schema
- XSL-Familie
  - XPath
  - XSLT

Die nach folgenden Erklärungen basieren auf W3C XML Schema 1.1 Structures, W3C XML Schema 1.1 Datatypes und W3C XML Schema Primer. Vereinfachungen an verschiedenen Stellen sollen das grundsätzliche Verständnis erleichtern.

XML-Schema

Aufbau eines XML-Schemas mit Zielnamensraum: Rahmen

```
<?xml version="1.0"?>
<xs:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.upb.de/webtec"
    xmlns:myns="http://www.upb.de/webtec"
    elementFormDefault="qualified"
    attributeFormDefault="qualified">
  ⋮
</xs:schema>
```

`xmlns:xs="..."`
Deklaration Schema-Namensraum für Vokabular des Schema-Dokumentes

`targetNamespace="..."`
Festlegung Zielnamensraum (optional; falls nicht vorhanden: anonymer Namensraum)

`xmlns:myns="..."`
Deklaration Zielnamensraum für Referenzen (benannte eigene Datentypen, global deklarierte Elemente und Attribute)

`elementFormDefault="qualified"`
`attributeFormDefault="qualified"`
Festlegung, ob Namen lokal deklarierter Element-/Attribut-Namen in Zielnamensraum gehören (`"qualified"`) oder zum anonymen Namensraum (`"unqualified"`, Defaultwert)

XML-Schema

Aufbau einer XML-Schemainstanz (Schema mit Zielnamensraum): Rahmen

```
<?xml version="1.0"?>
<myns:myElementName
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.upb.de/webtec
                        http://www.upb.de/webtec/myschema.xsd"
    xmlns:myns="http://www.upb.de/webtec"
    ...>
  ⋮
</myns:myElementName>
```

`xmlns:xsi="..."`
Deklaration Schemainstanz-Namensraum für Vokabular zur Anbindung des Schema-Dokumentes

`xsi:schemaLocation="..."`
Bindung Namensraum an Schema-Datei (auch mehrere Paare möglich)

`xmlns:myns="..."`
Deklaration Namensraum für eigene Elemente und Attribute (war Zielnamensraum in Schema-Datei)

`...`
Weitere Attribute für `myElementName` wie in Schema-Datei festgelegt

XML-Schema

Aufbau eines XML-Schemas ohne Zielnamensraum: Rahmen

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" >
  ⋮
</xs:schema>
```

`xmlns:xs="..."`
Deklaration Schema-Namensraum für Vokabular des Schema-Dokumentes

Ohne Festlegung Zielnamensraum: anonymer Namensraum ist Zielnamensraum

`elementFormDefault="qualified"`
`attributeFormDefault="qualified"`
Festlegung wäre ohne Wirkung, da anonymer Namensraum als Zielnamensraum verwendet

XML-Schema

Aufbau einer XML-Schemainstanz (Schema ohne Zielnamensraum): Rahmen

```
<?xml version="1.0"?>
<myElementName
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.upb.de/xsd/myschema.xsd"
    ...>
  ⋮
</myElementName>
```

`xmlns:xsi="..."`
Deklaration Schemainstanz-Namensraum für Vokabular zur Anbindung des Schema-Dokumentes

`xsi:noNamespaceSchemaLocation="..."`
Bindung Instanz an Schema-Datei (nur eine Schema-Datei möglich)

`...`
Weitere Attribute für `myElementName` wie in Schema-Datei festgelegt

XML-Schema

Deklarationen von Elementtypen in XML Schema

Deklarationen von Elementtypen erfolgen in einer XML Schema Datei

- in Elementinstanzen des Schema-Elementes `element` **ohne** Inhalt

  ```
  <xs:element name="myElementName" ... />
  ```
  (Kombi-Tag)
- in Elementinstanzen des Schema-Elementes `element` **mit** Inhalt

  ```
  <xs:element name="myElementName" ...> ... </xs:element>
  ```

mit Angabe eines Namens im Attribut `name` und Festlegung eines Datentyps.

Die Festlegung eines Datentyps erfolgt

- durch Referenzierung auf einen benannten Datentyp mit dem Attribut `type`

  ```
  <xs:element name="myElementName" type="xs:string"  ... />
  ```
- oder durch Definition eines anonymen (d.h. unbenanten) Datentyps als Inhalt.

  ```
  <xs:element name="myElementName" ...>
      <xs:complexType> ... </xs:complexType>
  </xs:attribute>
  ```
  (In Elementdeklarationen mit `complexType` oder `simpleType`.)

XML-Schema

Deklarationen von Attributen in XML Schema

Deklarationen von Attributen erfolgen in einer XML Schema Datei

- in Elementinstanzen des Schema-Elementes `attribute` **ohne** Inhalt

  ```
  <xs:attribute name="myAttributeName" ... />
  ```
  (Kombi-Tag)
- in Elementinstanzen des Schema-Elementes `element` **mit** Inhalt

  ```
  <xs:attribute name="myAttributeName" ...> ... </xs:attribute>
  ```

mit Angabe eines Namens im Attribut `name` und Festlegung eines Datentyps.

Die Festlegung eines Datentyps erfolgt

- durch Referenzierung auf einen benannten Datentyp mit dem Attribut `type`

  ```
  <xs:attribute name="myAttributeName" type="xs:string"  ... />
  ```
- oder durch Definition eines anonymen (d.h. unbenanten) Datentyps als Inhalt.

  ```
  <xs:attribute name="myAttributeName" ...>
      <xs:simpleType> ... </xs:simpleType>
  </xs:attribute>
  ```
  (In Attributdeklarationen **nur** mit `simpleType`.)

XML-Schema

Datentyp-Definitionen in XML Schema

Datentyp-Definitionen erfolgen in einer XML Schema Datei

- für einfache Datentypen ohne Attribute (reine Zeichenketten)

  in Elementinstanzen des Schema-Elementes `simpleType`

  ```
  <xs:simpleType name="mySimpleTypeName" ...> ... </xs:simpleType>
  ```

  bzw.

  ```
  <xs:simpleType ...> ... </xs:simpleType>
  ```

- für komplexe Datentypen (explizite Kindelemente, gemischter Inhalt, leerer Inhalt, einfache/vordefinierte Datentypen plus Attribute)

  in Elementinstanzen des Schema-Elementes `complexType`

  ```
  <xs:complexType name="myComplexTypeName" ...> ... </xs:complexType>
  ```

  bzw.

  ```
  <xs:complexType ...> ... </xs:complexType>
  ```

Die Definition eines anonymen Datentyps (als Kindelement von `element` oder `attribute`) erfolgt immer ohne Angabe eines Namens mit Attribut `name`.

XML-Schema

Position von Deklarationen/Definitionen in XML Schema Dateien

Deklarationen mit `element, attribute` und

Datentyp-Definitionen mit `simpleType, complexType` sind

- global

  (`element, attribute, simpleType, complexType` sind Instanzen als direkte Kindelemente von `schema`.)

  Ziel: Möglichkeit zur Wiederverwendung als Kindelement/Attribut/Datentyp, Wurzelelement

- lokal

  (Deklarationen/Datentyp-Definitionen sind geschachtelt in andere `element, attribute, simpleType, complexType` Instanzen.)

  Ziel: Einmalige Verwendung, daher Deklaration/Definition an Ort und Stelle der Verwendung

➜ Nur globale Datentyp-Definitionen erhalten einen Namen mit dem Attribut `name`.

XML-Schema

Datentyp-Definitionen in XML Schema (Fortsetzung)

Verwendung von Attributen und Elementen in Datentyp-Definitionen

Für die Inhaltsmodelle "explizite Kindelemente" und "gemischter Inhalt" werden

- die Kindelemente in einer Elementinstanz der Schema-Elemente `all`, `sequence` und `choice` angegeben mit
- Schachtelung von `all`, `sequence` und `choice`, falls nötig.

Für alle Inhaltsmodelle werden

- die Attribute unmittelbar vor dem schließenden Tag `complexType` angegeben.

Kindelemente und Attribute können

- an Ort und Stelle deklariert werden oder
  `<xs:element name="myElementName" ... > ... </xs:element>` bzw.
  `<xs:attribute name="myAttributeName" ... > ... </xs:attribute>`
- mit dem Attribut `ref` durch Referenzierung auf eine globale Deklaration angegeben werden.
  `<xs:element ref="myElementName" .../>` bzw.
  `<xs:attribute ref="myAttributeName" .../>`

XML-Schema

Datentyp-Definitionen in XML Schema (Fortsetzung)

Anzahlconstraints für Kindelemente

- Anzahlconstraints für Kindelemente werden mit den Attributen `minOccurs` und `maxOccurs` angegeben.
- Die Attribute `minOccurs` und `maxOccurs` dürfen direkt bei den Kindelementen in `element` und auch bei `all`, `sequence` und `choice` angegeben werden.
- Die Attribute `minOccurs` und `maxOccurs` sind nicht erlaubt im öffnenden Tag einer globalen Elementdeklaration.

Mögliche Werte bei `element`, `sequence` und `choice`

- `minOccurs`: 0,1,2,3,. . . ; Defaultwert: 1
- `maxOccurs`: 0,1,2,3,. . . und `unbounded`; Defaultwert: 1

Mögliche Werte bei `all`

- `minOccurs`: 0,1; Defaultwert: 1
- `maxOccurs`: 0,1; Defaultwert: 1

Der Wert bei `minOccurs` muss kleiner sein als der bei `maxOccurs`.

XML-Schema

Datentyp-Definitionen in XML Schema (Fortsetzung)

Anzahlconstraints für Kindelemente: Entsprechung für DTD

DTD	XML-Schema
ohne	`minOccurs="1" maxOccurs="1"`
?	`minOccurs="0" maxOccurs="1"`
+	`minOccurs="1" maxOccurs="unbounded"`
*	`minOccurs="0" maxOccurs="unbounded"`

(Angaben in grau können entfallen, da Default-Werte.)

DTD-Deklaration `<!ELEMENT A (B,(C|D+)?)*>` entspricht (`B,C,D` mit einfachem Inhalt)

```
<xs:element name="A">
  <xs:complexType>
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:element name="B" type="xs:string" minOccurs="0" />
      <xs:choice minOccurs="0">
        <xs:element name="C" type="xs:string" />
        <xs:element name="D" type="xs:string" maxOccurs="unbounded" />
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

(Lokale Deklaration der Kindelemente)

XML-Schema

Datentyp-Definitionen in XML Schema (Fortsetzung)

Constraints für Attribute

- Vorgaben für die Verwendung von Attributen werden mit den Attributen `default`, `fixed` und `use` gemacht.
- Die Attribute `default`, `fixed` und `use` werden in einem Element `attribute` angegeben.
- Die Attribute `default` (Defaultwert-Angabe) und `fixed` (nur ein fester Wert) sind nicht gleichzeitig erlaubt.
- Bei Verwendung des Attributes `default` ist nur die Möglichkeit `use="optional"` erlaubt.
- Das Attribut `use` ist nicht erlaubt im öffnenden Tag einer globalen Attributdeklaration.

Mögliche Werte

- `use`: `optional` und `required`; Defaultwert: `optional`
- `default`, `fixed`: Wert gemäß Datentyp

XML-Schema

Datentyp-Definitionen in XML Schema (Fortsetzung)

Defaultwerte für Attribute: Entsprechung für DTD

DTD	XML-Schema (Deklaration/Verwendung)	(Verwendung)
`#REQUIRED`		`use="required"`
`#IMPLIED`		`use="optional"`
`"Wert"`	`default="Wert"`	`use="optional"`
`#FIXED "Wert"`	`fixed="Wert"`	`use="optional"`

(Angaben in grau können entfallen, da Default-Werte.)

DTD-Deklaration `<!ATTLIST A att CDATA #REQUIRED>` entspricht (`A` mit leerem Inhalt)

```
<xs:element name="A">
  <xs:complexType>
    <xs:attribute name="att" type="xs:string" use="required" />
  </xs:complexType>
</xs:element>
```

(Lokale Deklaration des Attributes)

XML-Schema

Element-Referenzierung in XML Schema

Bei der Angabe von Attributen oder Kindelementen durch Referenzierung auf globale Deklarationen mit `ref` ist der Zielnamensraum der Deklarationen zu beachten.

Wurde ein Zielnamensraum festgelegt, muss die Referenzierung mit einem qualifizierten Namen erfolgen.

- Deklaration ohne Zielnamensraum:

  ```
  <xs:element ref="myElementName" minOccurs="unbounded" />
  ```
  bzw.
  ```
  <xs:attribute ref="myAttributeName" use="required" />
  ```

- Deklaration mit Zielnamensraum:

  ```
  <xs:element ref="myNs:myElementName" minOccurs="unbounded" />
  ```
  bzw.
  ```
  <xs:attribute ref="myNs:myAttributeName" use="required" />
  ```
  Der Zielnamensraum muss also für diese Verwendung deklariert worden sein, im Beispiel mit Bindung an das Präfix `myNs`.

XML-Schema

Datentyp-Referenzierung in XML Schema

Bei der Verwendung eigener benannter Datentypen ist bei der Referenzierung mit `type` der Zielnamensraum der Definition zu beachten.

Wurde ein Zielnamensraum festgelegt, muss die Referenzierung mit einem qualifizierten Namen erfolgen.

- Definition ohne Zielnamensraum:

  ```
  <xs:element name="myElementName" type="myType" />
  ```

- Definition mit Zielnamensraum:

  ```
  <xs:element name="myElementName" type="myns:myType" />
  ```

  Der Zielnamensraum muss also für diese Verwendung deklariert worden sein, im Beispiel mit Bindung an das Präfix `myns`.

- Verwendung von Datentypen aus XML Schema:

  ```
  <xs:element name="myElementName" type="xs:schemaType" />
  ```

  Die Namen dieser Datentypen gehören zum Schema-Namensraum.

(Analog lassen sich die Namen einfacher Datentypen für die Deklaration von Attributen nutzen.)

XML-Schema

Inhaltsmodelle

Dateirahmen für Beispiele mit Festlegung eines Zielnamensraumes

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.upb.de/webtec"
    xmlns:myns="http://www.upb.de/webtec"
    elementFormDefault="qualified"
    attributeFormDefault="qualified">
  ⋮
</xs:schema>
```

Dateirahmen für Beispiele ohne Festlegung eines Zielnamensraumes

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" >
  ⋮
</xs:schema>
```

XML-Schema

Inhaltsmodelle (Fortsetzung)

(a) einfacher Inhalt [W3C] :

1. Verwendung eines Built-in-Datentyps

```
<xs:element name="myElementName" type="xs:decimal"/>
```

Vergleichbare DTD-Definition

```
<!ELEMENT Elementname (#PCDATA)>
```

2. Verwendung eines Built-in-Datentyps plus Attribut

```
<xs:element name="myElementName"
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="source" type="xs:NMTOKEN"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

XML-Schema

Inhaltsmodelle (Fortsetzung)

(a) einfacher Inhalt [W3C] :

3. Definition eines benannten einfachen Datentyps (nur global möglich)

```
<xs:simpleType name="mySimpleTypeName">
  <xs:restriction base="xs:positiveInteger">
    <xs:minInclusive value="8"/>
    <xs:maxInclusive value="72"/>
  </xs:restriction>
</xs:simpleType>
```

Verwendung in Elementdeklaration bei festgelegtem Zielnamensraum:

```
<xs:element name="myElementName" type="myns:mySimpleTypeName"/>
```

Verwendung in Elementdeklaration ohne festgelegten Zielnamensraum:

```
<xs:element name="myElementName" type="mySimpleTypeName"/>
```

Beachte:
Je nach Festlegung eines Zielnamensraumes und dessen Verwendung innerhalb des Schema-Dokumentes ist eine Qualifizierung bei Verwendung des definierten Datentyps im Attribut `type` notwendig oder auch nicht.

XML-Schema

Inhaltsmodelle (Fortsetzung)

(a) einfacher Inhalt [W3C] :

4. Definition eines benannten einfachen Datentyps plus Attribut

```
<xs:complexType name="myAttributedSimpleTypeName">
  <xs:simpleContent>
    <xs:extension base="myns:mySimpleTypeName">
      <xs:attribute name="source" type="xs:NMTOKEN"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

<xs:simpleType name="mySimpleTypeName">
  <xs:restriction base="xs:positiveInteger">
    <xs:minInclusive value="8"/>
    <xs:maxInclusive value="72"/>
  </xs:restriction>
</xs:simpleType>
```

Der hohe Aufwand wird nur nötig, wenn wie hier ein **eigener** einfacher Datentyp die Basis der Erweiterung um ein Attribut sein soll und der Datentyp `mySimpleTypeName` auch anderweitig genutzt werden soll. Ansonsten verwenden wir eine Deklaration anlog zu Fall 2.

XML-Schema

Inhaltsmodelle (Fortsetzung)

(b) explizite Kindelemente [W3C] :

1. Verwendung lokaler Datentypdefinition mit Elementdeklarationen durch Built-in-Datentypen für Kindelemente

```
<xs:element name="myElementName">
  <xs:complexType>
    <xs:sequence minOccurs="0">
      <xs:element name="myChildName1" type="xs:string"
                                              maxOccurs="unbounded"/>
      <xs:element name="myChildName2" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Vergleichbare DTD-Definition (außer Anzahl-Constraints)

```
<!ELEMENT myElementName (myChildName1+, myChildName2)?>
```

XML-Schema

Inhaltsmodelle (Fortsetzung)

(b) explizite Kindelemente [W3C] :

2. Verwendung lokaler Datentypdefinition plus Attribut

```
<xs:element name="myElementName">
  <xs:complexType>
    <xs:choice>
      <xs:element name="myChildName1" type="xs:string"
                                              maxOccurs="unbounded"/>
      <xs:element name="myChildName2" type="xs:string"
                                              minOccurs="0"/>
    </xs:choice>
    <xs:attribute name="myAttributeName" type="xs:string"
                                              use="optional"/>
  </xs:complexType>
</xs:element>
```

Vergleichbare DTD-Definition (außer Anzahl-Constraints)

```
<!ELEMENT myElementName (myChildName1+ | myChildName2?)>
<!ATTLIST myElementName myAttributeName CDATA #IMPLIED>
```

XML-Schema

Inhaltsmodelle (Fortsetzung)

(b) explizite Kindelemente [W3C] :

3. Definition eines benannten komplexen Datentyps plus Attribut (nur global)

```
<xs:complexType name="myComplexTypeName">
  <xs:choice>
    <xs:element name="myChildName1" type="xs:string"
                                          maxOccurs="unbounded"/>
    <xs:element name="myChildName2" type="xs:string"
                                          minOccurs="0"/>
  </xs:choice>
  <xs:attribute name="myAttributeName" type="xs:string"
                                          use="optional"/>
</xs:complexType>
```

Verwendung in Elementdeklaration bei festgelegtem Zielnamensraum:

```
<xs:element name="myElementName" type="myns:myComplexTypeName"/>
```

Verwendung in Elementdeklaration ohne festgelegten Zielnamensraum:

```
<xs:element name="myElementName" type="myComplexTypeName"/>
```

XML-Schema

Inhaltsmodelle (Fortsetzung)

(b) explizite Kindelemente [W3C] :

4. Definition eines benannten komplexen Datentyps mit Verwendung anderer global deklarierter Elemente und Attribute bei festgelegtem Zielnamensraum

```
<xs:complexType name="myComplexTypeName">
  <xs:choice>
    <xs:element ref="myns:myChildName1" maxOccurs="unbounded"/>
    <xs:element ref="myns:myChildName2" minOccurs="0"/>
  </xs:choice>
  <xs:attribute ref="myns:myAttributeName" use="optional"/>
</xs:complexType>

<xs:element name="myChildName1" type="xs:string"/>
<xs:element name="myChildName2" type="xs:string"/>

<xs:attribute name="myAttributeName" type="xs:string"/>
```

Beachte die Qualifizierung bei Verwendung von `ref`.

Beachte die Position der Attribute `minOccurs`, `maxOccurs` und `use` bei der Verwendung der Elemente bzw. Attribute und nicht bei der Deklaration.

XML-Schema

Inhaltsmodelle (Fortsetzung)

(b) explizite Kindelemente [W3C] :

5. Definition eines benannten komplexen Datentyps mit Verwendung anderer global deklarierter Elemente und Attribute ohne festgelegten Zielnamensraum

```
<xs:complexType name="myComplexTypeName">
  <xs:choice>
    <xs:element ref="myChildName1" maxOccurs="unbounded"/>
    <xs:element ref="myChildName2" minOccurs="0"/>
  </xs:choice>
  <xs:attribute ref="myAttributeName" use="optional"/>
</xs:complexType>

<xs:element name="myChildName1" type="xs:string"/>
<xs:element name="myChildName2" type="xs:string"/>

<xs:attribute name="myAttributeName" type="xs:string"/>
```

Beachte die fehlende Qualifizierung bei Verwendung von `ref` in diesem Fall.

Beachte die Position der Attribute `minOccurs`, `maxOccurs` und `use` bei der Verwendung der Elemente bzw. Attribute und nicht bei der Deklaration.

XML-Schema

Inhaltsmodelle (Fortsetzung)

(c) gemischter Inhalt [W3C] :

1. Verwendung lokaler Datentypdefinition mit Elementdeklarationen durch Built-in-Datentypen

```
<xs:element name="myElementName">
  <xs:complexType mixed="true">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="myChildName1" type="xs:string"/>
      <xs:element name="myChildName2" type="xs:string"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
```

Vergleichbare DTD-Definition (außer Anzahl-Constraints)

```
<!ELEMENT myElementName (#PCDATA | myChildName1 | myChildName2) *>
```

Die Verwendung von `mixed="true"` ist in allen Definitionen von komplexen Datentypen möglich außer bei Definition von `simpleContent` wie in a) Bsp. 2 und Bsp. 4.

XML-Schema

Inhaltsmodelle (Fortsetzung)

(d) beliebiger Inhalt [W3C] :

In XML-Schema erlaubt der Datentyp `anyType` beliebigen Elementinhalt **und beliebige Attribute**.

1. Verwendung des Built-in-Datentyps

   ```
   <xs:element name="myElementName" type="xs:anyType"/>
   ```

   alternativ

   ```
   <xs:element name="myElementName"/>
   ```

   Vergleichbare DTD-Definition

   ```
   <!ELEMENT myElementName ANY>
   ```

XML-Schema

Inhaltsmodelle (Fortsetzung)

(d) beliebiger Inhalt [W3C] :

2. Verwendung des Built-in-Datentyps plus Attributdeklaration

```
<xs:element name="myElementName"
  <xs:complexType>
    <xs:complexContent>
      <xs:restriction base="xs:anyType">
        <xs:attribute name="myAttributeName" type="xs:decimal"/>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
```

Ziel ist hier die Einschränkung des Attributes `myAttributeName` auf einen bestimmten Datentyp und eine bestimmte Nutzung. Ohne Deklaration sind für alle Attribute beliebige Werte möglich.

XML-Schema

Inhaltsmodelle (Fortsetzung)

(d) beliebiger Inhalt [W3C] :

3. Definition eines benannten komplexen Datentyps plus Attribut (nur global)

```
<xs:complexType name="myAnyType"/>
  <xs:complexContent>
    <xs:restriction base="xs:anyType">
      <xs:attribute name="myAttributeName" type="xs:decimal"/>
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
```

Verwendung in Elementdeklaration mit festgelegtem Zielnamensraum:

```
<xs:element name="myElementName" type="myns:myAnyType"/>
```

Verwendung in Elementdeklaration ohne festgelegten Zielnamensraum:

```
<xs:element name="myElementName" type="myAnyType"/>
```

XML-Schema

Inhaltsmodelle (Fortsetzung)

(e) leerer Inhalt [W3C] :

1. Verwendung lokaler Datentypdefinition

```
<xs:element name="myElementName">
  <xs:complexType>
  </xs:complexType>
</xs:element>
```

Vergleichbare DTD-Definition

```
<!ELEMENT myElementName EMPTY>
```

2. Verwendung lokaler Datentypdefinition plus Attribut

```
<xs:element name="myElementName">
  <xs:complexType>
    <xs:attribute name="myAttributeName" type="xs:string"/>
  </xs:complexType>
</xs:element>
```

XML-Schema

Inhaltsmodelle (Fortsetzung)

(e) leerer Inhalt [W3C] :

3. Definition eines benannten komplexen Datentyp (nur global)

```
<xs:complexType name="myEmptyType">
</xs:complexType>
```

Verwendung in Elementdeklaration mit festgelegtem Zielnamensraum:

```
<xs:element name="myElementName" type="myns:myEmptyType"/>
```

Verwendung in Elementdeklaration ohne festgelegten Zielnamensraum:

```
<xs:element name="myElementName" type="myEmptyType"/>
```

4. Definition eines benannten komplexen Datentyps plus Attribut (nur global)

```
<xs:complexType name="myEmptyType"/>
  <xs:attribute name="myAttributeName" type="xs:string"/>
</xs:complexType>
```

XML-Schema

Attributdeklarationen

Attribute können nur einfache Datentypen haben. Sie werden meist aus den verfügbaren einfachen Built-in-Datentypen [W3C] oder selbstdefinierten einfachen Datentypen durch Einschränkung der Wertemenge abgeleitet.

1. Verwendung des Built-in-Datentyps

   ```
   <xs:attribute name="myAttributeName" type="xs:string"/>
   ```

2. Verwendung anonymer Datentyp-Definition

   ```
   <xs:attribute name="myAttributeName"/>
     <xs:simpleType>
       <xs:restriction base="xs:positiveInteger">
         <xs:minInclusive value="8"/>
         <xs:maxInclusive value="72"/>
       </xs:restriction>
     </xs:simpleType>
   </xs:attribute>
   ```

3. Verwendung eines benannten einfachen Datentyps bei festgelegtem Zielnamensraum:

   ```
   <xs:attribute name="myAttributeName" type="myns:mySimpleTypeName"/>
   ```

   Verwendung in Elementdeklaration ohne festgelegten Zielnamensraum:

   ```
   <xs:attribute name="myAttributeName" type="mySimpleTypeName"/>
   ```

XML-Schema

Beispiele zur Definition von Datentypen für Attribute

1. Angabe von Minimal-, Maximalwerten

```
<xs:simpleType name="mySimpleTypeName1">
  <xs:restriction base="xs:positiveInteger">
    <xs:minInclusive value="8"/>
    <xs:maxInclusive value="72"/>
  </xs:restriction>
</xs:simpleType>
```

2. Angabe von Minimal-, Maximalwerten

```
<xs:simpleType name="mySimpleTypeName2">
  <xs:restriction base="xs:date">
    <xs:minExclusive value="1999-05-31"/>
    <xs:maxExclusive value="2014-05-31"/>
  </xs:restriction>
</xs:simpleType>
```

XML-Schema

Beispiele zur Definition von Datentypen für Attribute (Fortsetzung)

3. Angabe von Pattern für die Zeichenkette

```
<xs:simpleType name="mySimpleTypeName3">
  <xs:restriction base="xs:string">
    <xs:pattern value="CODE [0-9]9-[A-Z]2"/>
  </xs:restriction>
</xs:simpleType>
```

4. Angabe von Pattern für die Zeichenkette

```
<xs:simpleType name="mySimpleTypeName4">
  <xs:restriction base="xs:decimal">
    <xs:pattern value="[1-9][0-9]*|0"/>
  </xs:restriction>
</xs:simpleType>
```

XML-Schema

Beispiele zur Definition von Datentypen für Attribute (Fortsetzung)

5. Angabe der genauen Länge der Zeichenkette

```
<xs:simpleType name="mySimpleTypeName5">
  <xs:restriction base="xs:decimal">
    <xs:length value="10"/>
  </xs:restriction>
</xs:simpleType>
```

6. Angabe von Minimal-, Maximallänge

```
<xs:simpleType name="mySimpleTypeName6">
  <xs:restriction base="xs:QName">
    <xs:minLength value="3"/>
    <xs:maxLength value="12"/>
  </xs:restriction>
</xs:simpleType>
```

XML-Schema

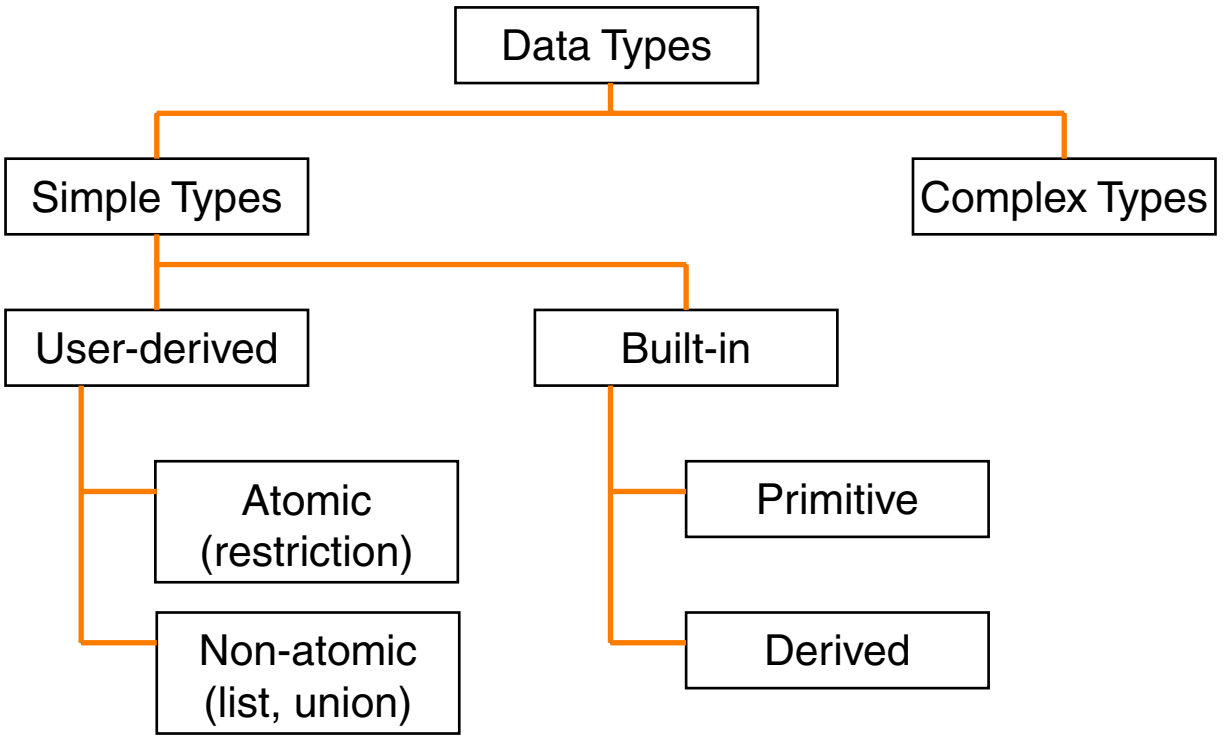
Beispiele zur Definition von Datentypen für Attribute (Fortsetzung)

7. Aufzählung der möglichen Werte für die Zeichenkette

```
<xs:simpleType name="mySimpleTypeName7">
  <xs:restriction base="xs:string">
    <xs:enumeration value="alpha"/>
    <xs:enumeration value="beta"/>
    <xs:enumeration value="gamma"/>
    <xs:enumeration value="delta"/>
    <xs:enumeration value="epsilon"/>
    <xs:enumeration value="zeta"/>
    <xs:enumeration value="eta"/>
    <xs:enumeration value="theta"/>
    <xs:enumeration value="iota"/>
    <xs:enumeration value="kappa"/>
    <xs:enumeration value="lambda"/>
  </xs:restriction>
</xs:simpleType>
```

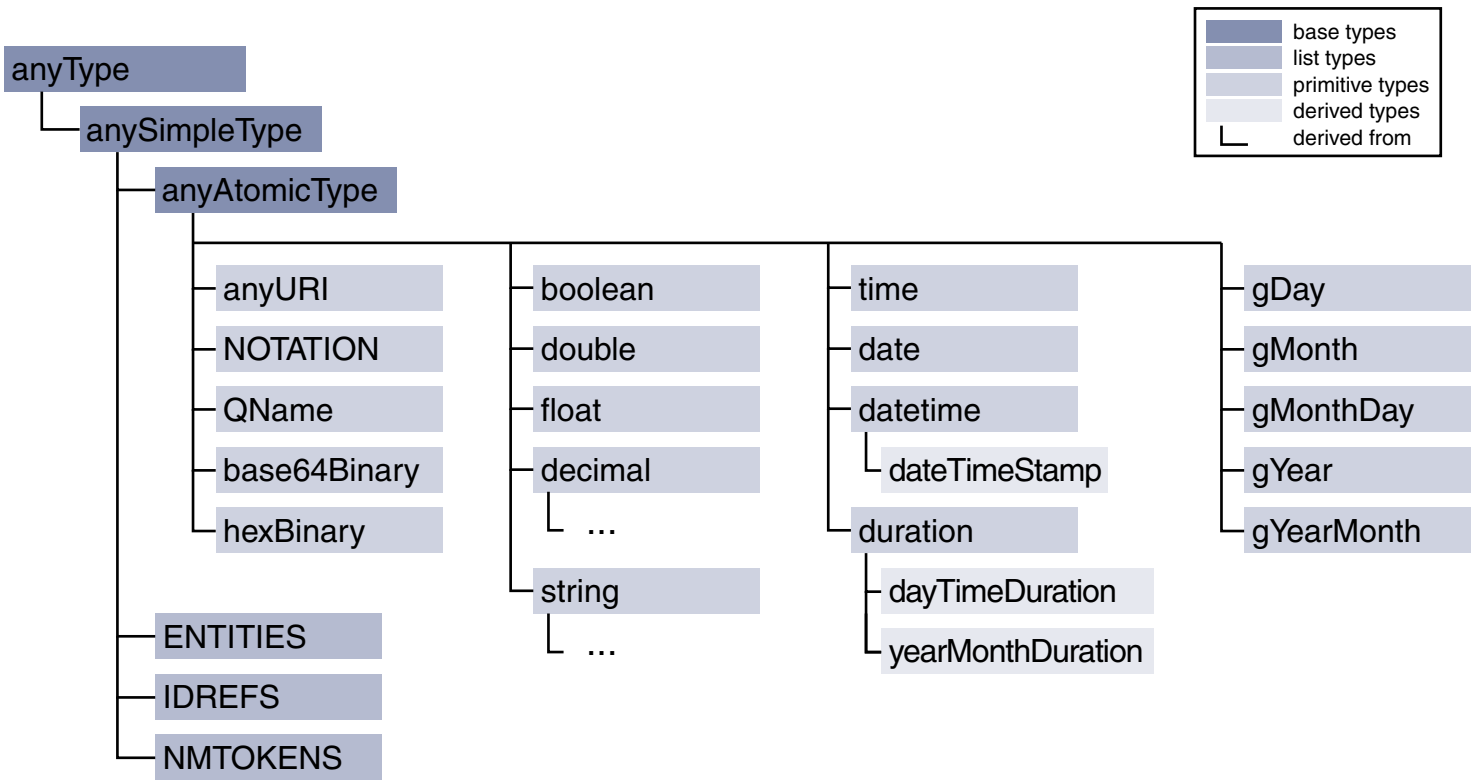
XML-Schema Datentypen

Datentyp Hierarchie



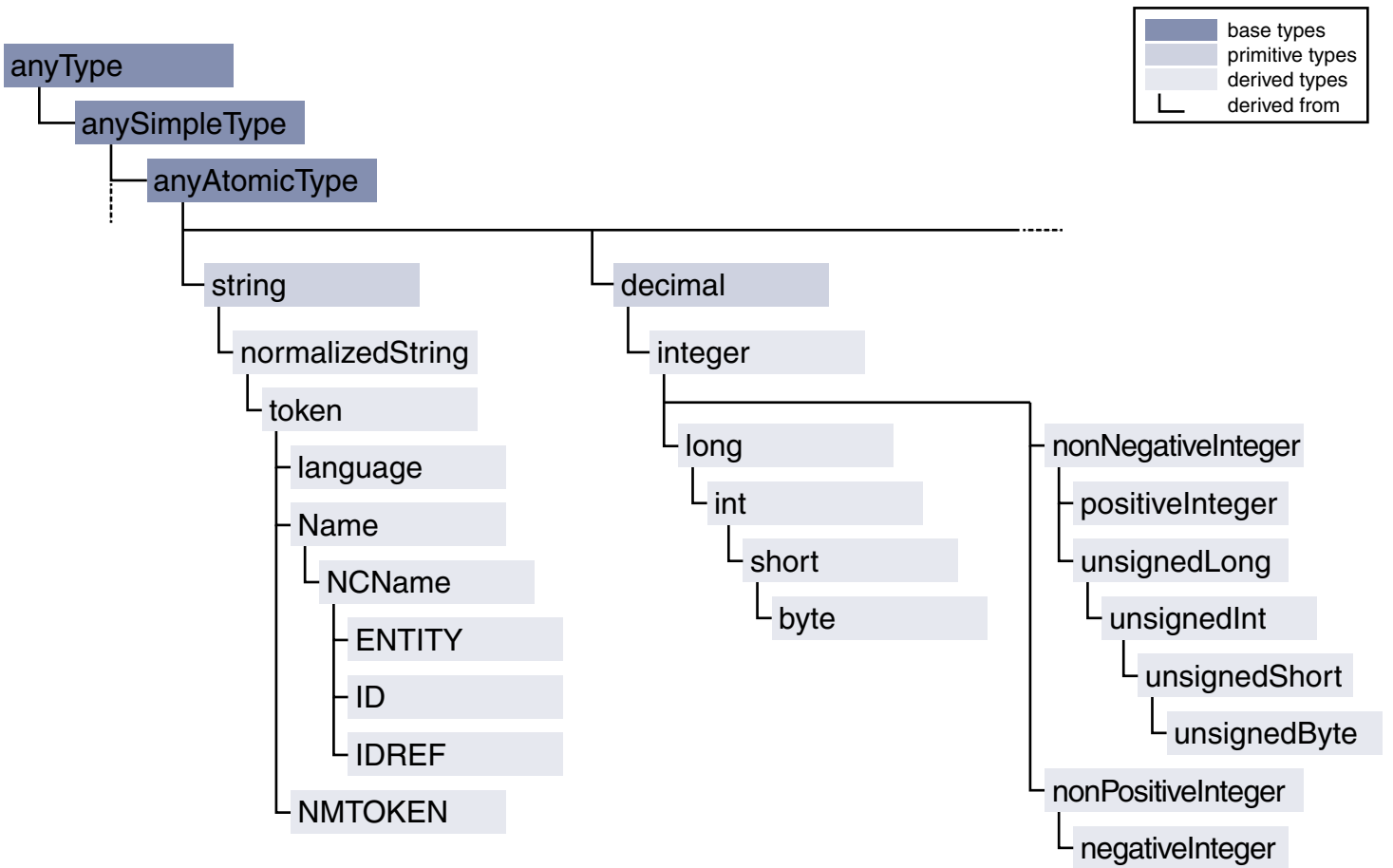
XML-Schema Datentypen

Datentyp Hierarchie: vordefinierte einfache Datentypen



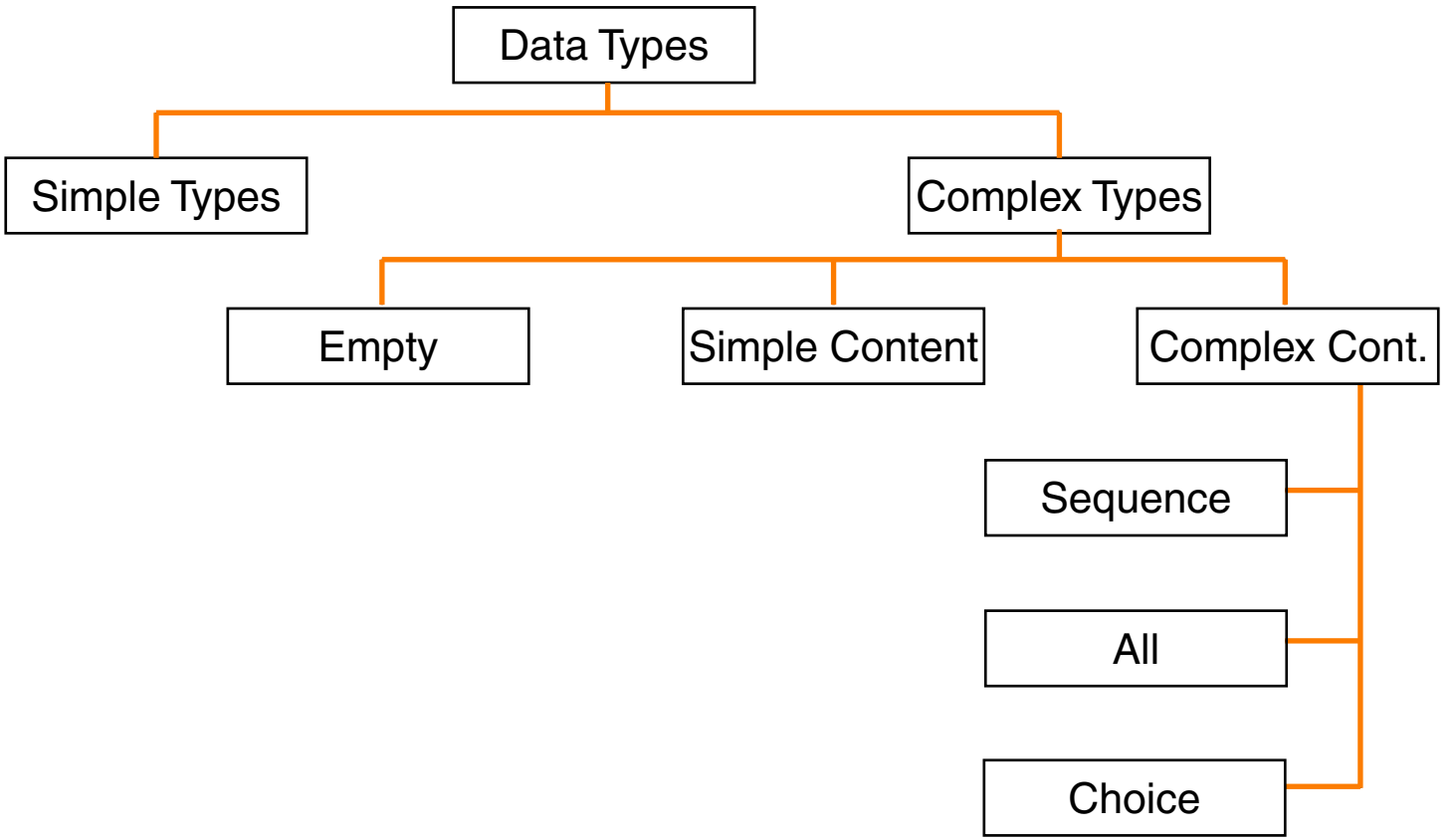
XML-Schema Datentypen

Datentyp Hierarchie: vordefinierte einfache Datentypen (Fortsetzung)



XML-Schema Datentypen

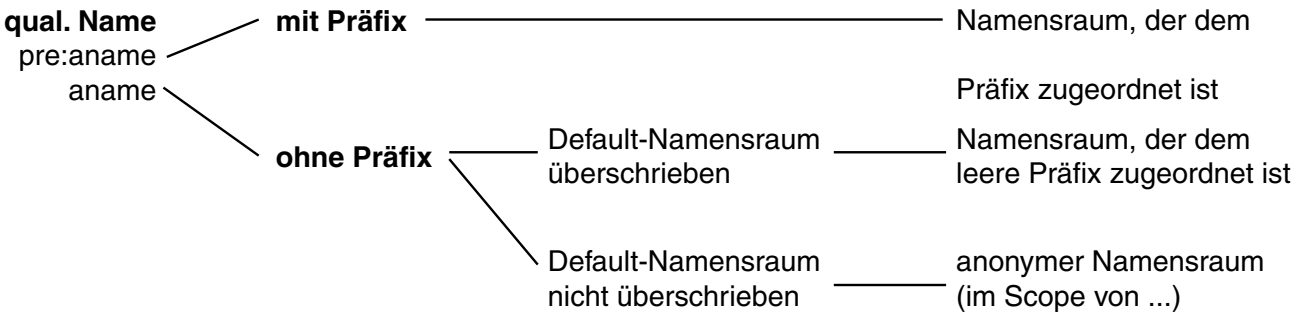
Datentyp Hierarchie: Definition komplexer Datentypen



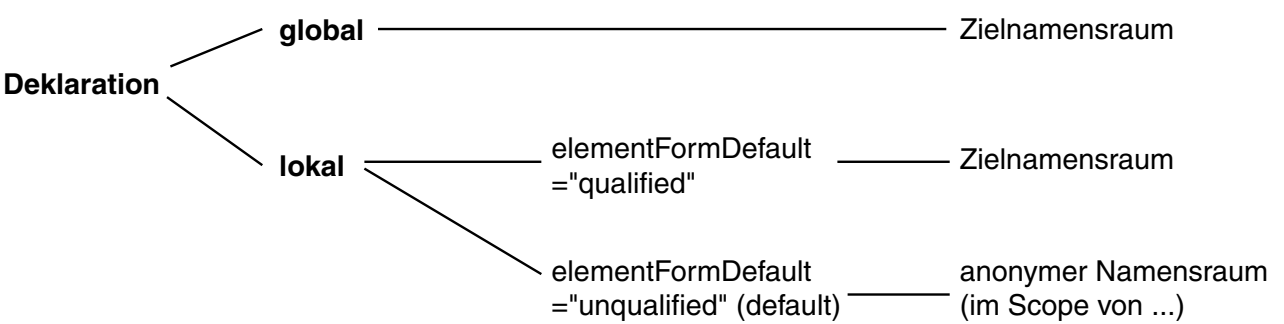
XML-Schema

Zuordnung von Element-, Datentypnamen zu Namensräumen

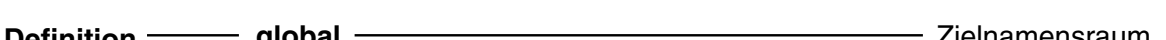
In welchem Namensraum wird ein qualifizierter Name (kein Attributname) gesucht???



In welchen Namensraum wird ein neuer Elementname gesetzt???



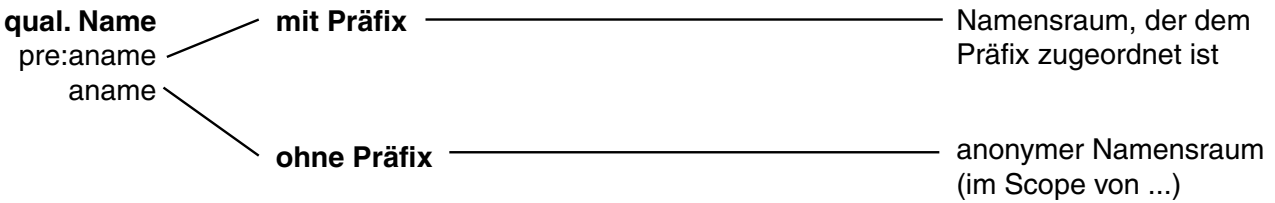
In welchen Namensraum wird ein neuer Datentypname gesetzt???



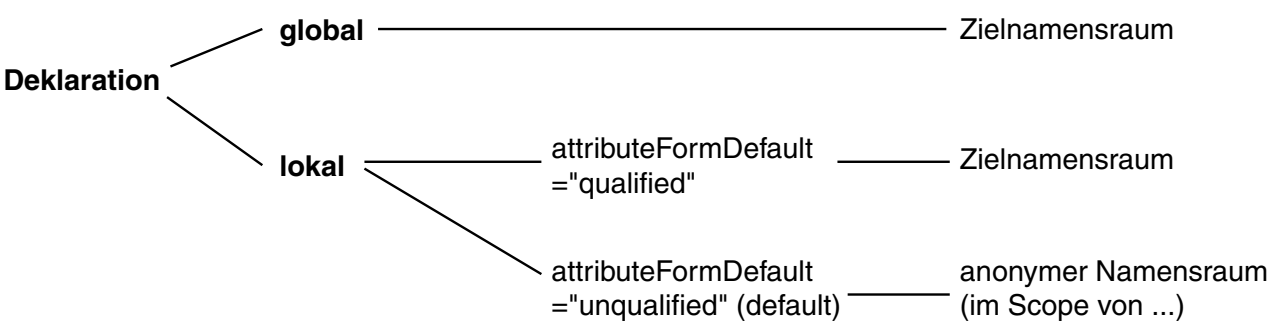
XML-Schema

Zuordnung von Attributnamen zu Namensräumen

In welchem Namensraum wird ein qualifizierter Attributname gesucht???



In welchen Namensraum wird ein neuer Attributname gesetzt???



XML-Schema Entwurfsprinzipien

XML-Schema: Festlegung einer Auszeichnungssprache

Elementtypen, Attribute, Datentypen:
Beschreibung von Deklaration/Definition und Verwendung

- Elementtyp
  - Deklaration: Festlegung von Name und Datentyp
  - Verwendung: Nennung als Kindelement und Anzahlconstraints (`minOccirs, maxOccurs`)
- Attribut
  - Deklaration: Festlegung von Name und Datentyp
  - Verwendung: Nennung als Attribut und Verwendungsvorgaben (`use, dafault, fixed`)
- Datentyp
  - Definition: Festlegung Wertebereich (falls nicht Built-in-Datentyp)
  - Verwendung: Nennung als Datentyp

XML-Schema Entwurfsprinzipien

XML-Schema: Festlegung einer Auszeichnungssprache (Fortsetzung)

Entwurfsmurter 1a:
Deklaration/Definition und Verwendung an ein-und-derselben Stelle angeben

- Elementtyp: Nennung als Kindelement mit Deklaration an Ort und Stelle und Festlegung der Anzahlconstraints
- Attribut: Nennung als Attribut mit Deklaratio an Ort und Stelle und Festlegung der Verwendungsvorgaben
- Datentyp: Nennung als Datentyp und Festlegung Wertebereich an Ort und Stelle als anonymer Datentyp, falls nicht Built-in

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="book" >
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string" />
        <xs:element name="author" type="xs:string" />
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded" >
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string" />
              <xs:element name="since" type="xs:date" />
              <xs:element name="qualification" type="xs:string" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

XML-Schema Entwurfsprinzipien

XML-Schema: Festlegung einer Auszeichnungssprache (Fortsetzung)

Entwurfsmurter 1b: Wiederverwendung von Elementtypen und Attributen

Trennung von Deklaration und Verwendung für Elementtypen und Attrinute

- Elementtyp: Deklaration global mit `element` und `name`
  Referenzierung an Stelle der Verwendung mit `element` und `ref`
- Attribut: Deklaration global mit `attribute` und `name`
  Referenzierung an Stelle der Verwendung mit `attribute` und `ref`

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="book" >
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="title"/>
        <xs:element ref="author"/>
        <xs:element ref="character" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute ref="isbn" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="title" type="xs:string" />
  <xs:element name="author" type="xs:string" />
  <xs:element name="character">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="name"/>
        <xs:element ref="since"/>
        <xs:element ref="qualification"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="name" type="xs:string" />
  <xs:element name="since" type="xs:date" />
  <xs:element name="qualification" type="xs:string" />
  <xs:attribute name="isbn" type="xs:string"/>
</xs:schema>
```

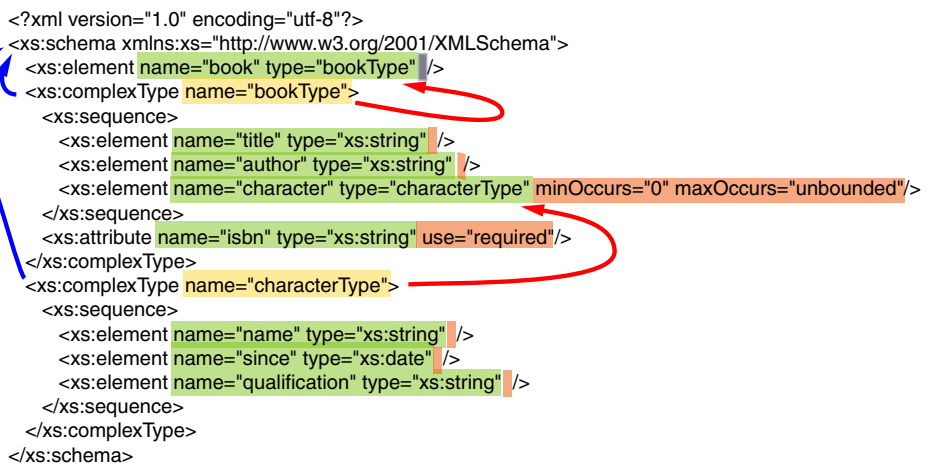
XML-Schema Entwurfsprinzipien

XML-Schema: Festlegung einer Auszeichnungssprache (Fortsetzung)

Entwurfsmurter 2: Wiederverwendung von Datentypen

Trennung von Definition und Verwendung für Datentypen

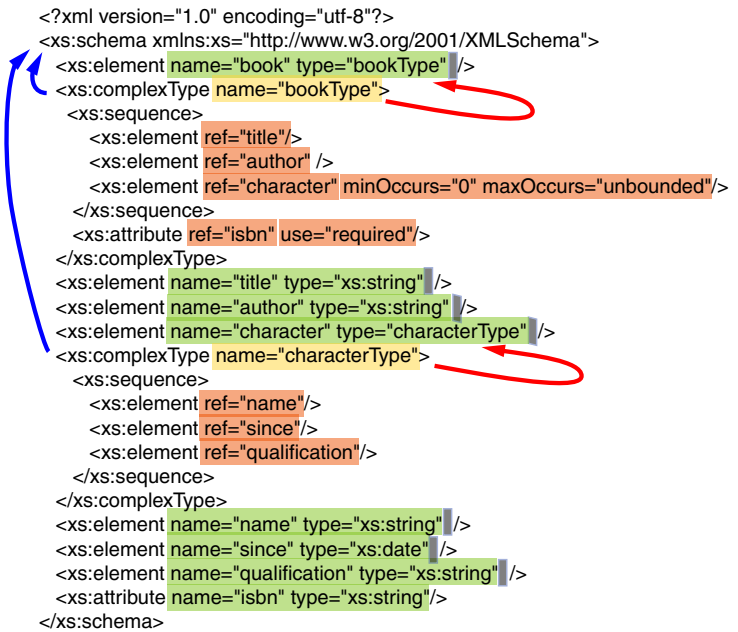
- Datentyp:
  Definition global mit `complexType/simpleType` und `name`
  Referenzierung an Stelle der Verwendung mit `type` (oder `base`)



XML-Schema Entwurfsprinzipien

XML-Schema: Festlegung einer Auszeichnungssprache (Fortsetzung)

Kombination der Entwurfsmuster nach Bedarf



```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="book" type="bookType" />
  <xs:complexType name="bookType">
    <xs:sequence>
      <xs:element ref="title"/>
      <xs:element ref="author" />
      <xs:element ref="character" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="isbn" use="required"/>
  </xs:complexType>
  <xs:element name="title" type="xs:string" />
  <xs:element name="author" type="xs:string" />
  <xs:element name="character" type="characterType" />
  <xs:complexType name="characterType">
    <xs:sequence>
      <xs:element ref="name"/>
      <xs:element ref="since"/>
      <xs:element ref="qualification"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="name" type="xs:string" />
  <xs:element name="since" type="xs:date" />
  <xs:element name="qualification" type="xs:string" />
  <xs:attribute name="isbn" type="xs:string"/>
</xs:schema>
```

- Global deklarierte Elementtypen können als Wurzelelemente in Instanzdokumenten verwendet werden.
- Referenzierung von Elementtypen, Attributen und Datentypen erfolgt über qualifizierte Namen. Dazu ist ggf. eine Namensraumdeklaration mit Präfixbindung und die Verwendung dieses Präfixes nötig.

XML-Schema Entwurfsprinzipien

Entwurfsprinzip 1a: Russian Doll Version 1

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.upb.de/cs/webis2009"
           elementFormDefault="qualified">
  <xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="to" type="xs:string" maxOccurs="unbounded"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
      </xs:sequence>
      <xs:attribute name="sent" type="xs:dateTime" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Zielnamensraum:
Namen global deklarierter Elemente und Attribute und Namen benannter Datentypen
`elementFormDefault="qualified"` auch Namen lokal deklarierter Elemente
`attributeFormDefault="qualified"` auch Namen lokal deklarierter Attribute

Q. Zu welchen Namensräumen gehören die Namen deklarierter Elementtypen und Attribute?

XML-Schema Entwurfsprinzipien

Entwurfsprinzip 1a: Russian Doll Version 2

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.upb.de/cs/webis2009">
  <xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="to" type="xs:string" maxOccurs="unbounded"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
      </xs:sequence>
      <xs:attribute name="sent" type="xs:dateTime" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Q. Zu welchen Namensräumen gehören die Namen deklarierter Elementtypen und Attribute?

XML-Schema Entwurfsprinzipien

Entwurfsprinzip teilweise Russian Doll, teilweise Verwendung globaler Deklarationen

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.upb.de/cs/webis2009"
           xmlns:tns="http://www.upb.de/cs/webis2009">
  <xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element re="tns:to" maxOccurs="unbounded"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
      </xs:sequence>
      <xs:attribute ref="tns:sent" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="to" type="xs:string"/>
  <xs:attribute name="sent" type="xs:dateTime"/>
</xs:schema>
```

Q. Zu welchen Namensräumen gehören die Namen deklarierter Elementtypen und Attribute?

XML-Schema Entwurfsprinzipien

Entwurfsprinzip Russian Doll, teilweise Verwendung globaler Deklarationen, Verwendung benannter Datentypen

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.upb.de/cs/webis2009"
           xmlns:tns="http://www.upb.de/cs/webis2009">

  <xs:complexType name="noteType">
    <xs:sequence>
      <xs:element re="tns:to" maxOccurs="unbounded"/>
      <xs:element name="from" type="xs:string"/>
      <xs:element name="body" type="xs:string"/>
    </xs:sequence>
    <xs:attribute ref="tns:sent" use="required"/>
  </xs:complexType>
  <xs:element name="note" type="tns:noteType"/>
  <xs:element name="to" type="xs:string"/>
  <xs:attribute name="sent" type="xs:dateTime"/>
</xs:schema>
```

Q. Zu welchen Namensräumen gehören die Namen deklarierter Elementtypen und Attribute?

XML-Schema Entwurfsprinzipien

XML-Schema Dokumente ohne Zielnamensraumvereinbarung (d.h. anonymer Namensraum = Zielnamensraum)

Modellierung nach Russian Doll Prinzip

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="since" type="xs:date"/>
              <xs:element name="qualification" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Modellierung mit benannten Datentypen

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="book" type="bookType"/>
  <xs:complexType name="bookType">
    <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="author" type="xs:string"/>
      <xs:element name="character" type="characterType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="isbn" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="characterType">
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="since" type="xs:date"/>
      <xs:element name="qualification" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

Modellierung mit globalen Element-/Attribiutdeklarationen

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="title"/>
        <xs:element ref="author"/>
        <xs:element ref="character" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute ref="isbn" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="title" type="xs:string"/>
  <xs:element name="author" type="xs:string"/>
  <xs:element name="character">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="name"/>
        <xs:element ref="since"/>
        <xs:element ref="qualification"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="since" type="xs:date"/>
  <xs:element name="qualification" type="xs:string"/>
  <xs:attribute name="isbn" type="xs:string"/>
</xs:schema>
```

Modellierung mit benannten Datentypen und mit globalen Element-/Attribiutdeklarationen

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="book" type="bookType"/>
  <xs:complexType name="bookType">
    <xs:sequence>
      <xs:element ref="title"/>
      <xs:element ref="author" />
      <xs:element ref="character" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="isbn" use="required"/>
  </xs:complexType>
  <xs:element name="title" type="xs:string"/>
  <xs:element name="author" type="xs:string"/>
  <xs:element name="character" type="characterType"/>
  <xs:complexType name="characterType">
    <xs:sequence>
      <xs:element ref="name"/>
      <xs:element ref="since"/>
      <xs:element ref="qualification"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="since" type="xs:date"/>
  <xs:element name="qualification" type="xs:string"/>
  <xs:attribute name="isbn" type="xs:string"/>
</xs:schema>
```

Legende: Name aus globaler Dekl./Def. | Name aus lokaler Dekl. | anonym (lokal) def. Datentyp | Namensreferenz

XML-Schema Entwurfsprinzipien

XML-Schema Dokumente mit Zielnamensraumvereinbarung Fall 1

Modellierung nach Russian Doll Prinzip

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="unqualified" attributeFormDefault="unqualified">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="since" type="xs:date"/>
              <xs:element name="qualification" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Modellierung mit globalen Element-/Attribiutdeklarationen

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="unqualified" attributeFormDefault="unqualified"
        xmlns:tns="http://www.upb.de/Sample">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="tns:title"/>
        <xs:element ref="tns:author"/>
        <xs:element ref="tns:character" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute ref="tns:isbn" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="title" type="xs:string"/>
  <xs:element name="author" type="xs:string"/>
  <xs:element name="character">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="tns:name"/>
        <xs:element ref="tns:since"/>
        <xs:element ref="tns:qualification"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="since" type="xs:date"/>
  <xs:element name="qualification" type="xs:string"/>
  <xs:attribute name="isbn" type="xs:string"/>
</xs:schema>
```

Modellierung mit benannten Datentypen

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="unqualified" attributeFormDefault="unqualified"
        xmlns:tns="http://www.upb.de/Sample">
  <xs:element name="book" type="tns:bookType"/>
  <xs:complexType name="bookType">
    <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="author" type="xs:string"/>
      <xs:element name="character" type="tns:characterType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="isbn" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="characterType">
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="since" type="xs:date"/>
      <xs:element name="qualification" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

Modellierung mit benannten Datentypen und mit globalen Element-/Attribiutdeklarationen

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="unqualified" attributeFormDefault="unqualified"
        xmlns:tns="http://www.upb.de/Sample">
  <xs:element name="book" type="tns:bookType"/>
  <xs:complexType name="bookType">
    <xs:sequence>
      <xs:element ref="tns:title"/>
      <xs:element ref="tns:author" />
      <xs:element ref="tns:character" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="tns:isbn" use="required"/>
  </xs:complexType>
  <xs:element name="title" type="xs:string"/>
  <xs:element name="author" type="xs:string"/>
  <xs:element name="character" type="tns:characterType"/>
  <xs:complexType name="characterType">
    <xs:sequence>
      <xs:element ref="tns:name"/>
      <xs:element ref="tns:since"/>
      <xs:element ref="tns:qualification"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="since" type="xs:date"/>
  <xs:element name="qualification" type="xs:string"/>
  <xs:attribute name="isbn" type="xs:string"/>
</xs:schema>
```

Legende: Name aus globaler Dekl./Def. in Zielnamensraum gesetzt; Name aus lokaler Dekl. in anon. Namensraum gesetzt; anonym (lokal) def. Datentyp; Namensreferenz mit Präfix gebunden an Zielnamensraum

XML-Schema Entwurfsprinzipien

XML-Schema Dokumente mit Zielnamensraumvereinbarung Fall 2

Modellierung nach Russian Doll Prinzip

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="qualified" attributeFormDefault="qualified">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="since" type="xs:date"/>
              <xs:element name="qualification" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Modellierung mit globalen Element-/Attribiutdeklarationen

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="qualified" attributeFormDefault="qualified"
        xmlns:tns="http://www.upb.de/Sample">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="tns:title"/>
        <xs:element ref="tns:author"/>
        <xs:element ref="tns:character" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute ref="tns:isbn" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="title" type="xs:string"/>
  <xs:element name="author" type="xs:string"/>
  <xs:element name="character">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="tns:name"/>
        <xs:element ref="tns:since"/>
        <xs:element ref="tns:qualification"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="since" type="xs:date"/>
  <xs:element name="qualification" type="xs:string"/>
  <xs:attribute name="isbn" type="xs:string"/>
</xs:schema>
```

Modellierung mit benannten Datentypen

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="qualified" attributeFormDefault="qualified"
        xmlns:tns="http://www.upb.de/Sample">
  <xs:element name="book" type="tns:bookType"/>
  <xs:complexType name="bookType">
    <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="author" type="xs:string"/>
      <xs:element name="character" type="tns:characterType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="isbn" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="characterType">
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="since" type="xs:date"/>
      <xs:element name="qualification" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

Modellierung mit benannten Datentypen und mit globalen Element-/Attribiutdeklarationen

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="qualified" attributeFormDefault="qualified"
        xmlns:tns="http://www.upb.de/Sample">
  <xs:element name="book" type="tns:bookType"/>
  <xs:complexType name="bookType">
    <xs:sequence>
      <xs:element ref="tns:title"/>
      <xs:element ref="tns:author" />
      <xs:element ref="tns:character" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="tns:isbn" use="required"/>
  </xs:complexType>
  <xs:element name="title" type="xs:string"/>
  <xs:element name="author" type="xs:string"/>
  <xs:element name="character" type="tns:characterType"/>
  <xs:complexType name="characterType">
    <xs:sequence>
      <xs:element ref="tns:name"/>
      <xs:element ref="tns:since"/>
      <xs:element ref="tns:qualification"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="since" type="xs:date"/>
  <xs:element name="qualification" type="xs:string"/>
  <xs:attribute name="isbn" type="xs:string"/>
</xs:schema>
```

Legende: Name aus globaler Dekl./Def. in Zielnamensraum gesetzt; Name aus lokaler Dekl. in Zielnamensraum gesetzt; anonym (lokal) def. Datentyp; Namensreferenz mit Präfix gebunden an Zielnamensraum

XML-Schema Entwurfsprinzipien

XML-Instanzdokument zu XML-Schema Dokumente ohne Zielnamensraumvereinbarung (d.h. anonymer Namensraum = Zielnamensraum)

Modellierung nach Russian Doll Prinzip

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="since" type="xs:date"/>
              <xs:element name="qualification" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Instanzdokument

```
<?xml version="1.0" encoding="utf-8"?>
<book xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="sample.xsd"
        isbn="0836217462">
  <title>Being a Dog Is a Full-Time Job</title>
  <author>Charles M. Schulz</author>
  <character>
    <name>Snoopy</name>
    <since>1950-10-04</since>
    <qualification>extroverted beagle</qualification>
  </character>
  <character>
    <name>Peppermint Patty</name>
    <since>1966-08-22</since>
    <qualification>bold, brash and tomboyish</qualification>
  </character>
</book>
```

Legende: Name in Zielnamensraum Name in anon. Namnesraum

XML-Schema Entwurfsprinzipien

XML-Schema Dokumente mit Zielnamensraumvereinbarung

Modellierung nach Russian Doll Prinzip

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="unqualified" attributeFormDefault="unqualified">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="since" type="xs:date"/>
              <xs:element name="qualification" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Instanzdokument

```
<?xml version="1.0" encoding="utf-8"?>
<tns:book xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.upb.de/Sample   sample.xsd"
        xmlns:tns="http://www.upb.de/Sample"
        isbn="0836217462">
  <title>Being a Dog Is a Full-Time Job</title>
  <author>Charles M. Schulz</author>
  <character>
    <name>Snoopy</name>
    <since>1950-10-04</since>
    <qualification>extroverted beagle</qualification>
  </character>
  <character>
    <name>Peppermint Patty</name>
    <since>1966-08-22</since>
    <qualification>bold, brash and tomboyish</qualification>
  </character>
</tns:book>
```

Verwendung von "http://www.upb.de/Sample" als Default-Namensraum (Bindung an leeres Präfix) nicht sinnvoll, da auch Elementnamen des anonymen Namensraumes verwendet werden.

Modellierung nach Russian Doll Prinzip

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="qualified" attributeFormDefault="qualified">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="since" type="xs:date"/>
              <xs:element name="qualification" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Instanzdokument

```
<?xml version="1.0" encoding="utf-8"?>
<tns:book xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.upb.de/Sample   sample.xsd"
        xmlns:tns="http://www.upb.de/Sample"
        tns:isbn="0836217462">
  <tns:title>Being a Dog Is a Full-Time Job</tns:title>
  <tns:author>Charles M. Schulz</tns:author>
  <tns:character>
    <tns:name>Snoopy</tns:name>
    <tns:since>1950-10-04</tns:since>
    <tns:qualification>extroverted beagle</tns:qualification>
  </tns:character>
  <tns:character>
    <tns:name>Peppermint Patty</tns:name>
    <tns:since>1966-08-22</tns:since>
    <tns:qualification>bold, brash and tomboyish</tns:qualification>
  </tns:character>
</tns:book>
```

Verwendung von "http://www.upb.de/Sample" als Default-Namensraum (Bindung an leeres Präfix) möglich, spart alle "tns:" bei Elementen. Beachte, dass "isbn" ohne qulifizierendes Präfix als Attribut immer im anonymen Namensraum gesucht würde. Daher wird zusätzlich eine Bindung des Namensraumes an ein Präfix "tns:" benötigt, um das Attribut "isbn" korrekt qualifizieren zu können.

Legende: Name in Zielnamensraum Name in anon. Namnesraum

XML-Schema Entwurfsprinzipien

XML-Schema Dokumente mit Zielnamensraumvereinbarung

Modellierung nach Russian Doll Prinzip

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="unqualified" attributeFormDefault="qualified">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="since" type="xs:date"/>
              <xs:element name="qualification" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Instanzdokument

```
<?xml version="1.0" encoding="utf-8"?>
<tns:book xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.upb.de/Sample   sample.xsd"
        xmlns:tns="http://www.upb.de/Sample"
        tns:isbn="0836217462">
  <title>Being a Dog Is a Full-Time Job</title>
  <author>Charles M. Schulz</author>
  <character>
    <name>Snoopy</name>
    <since>1950-10-04</since>
    <qualification>extroverted beagle</qualification>
  </character>
  <character>
    <name>Peppermint Patty</name>
    <since>1966-08-22</since>
    <qualification>bold, brash and tomboyish</qualification>
  </character>
</tns:book>
```

Verwendung von "http://www.upb.de/Sample" als Default-Namensraum (Bindung an leeres Präfix) nicht sinnvoll, da auch Elementnamen des anonymen Namensraumes verwendet werden.

Modellierung nach Russian Doll Prinzip

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.upb.de/Sample"
        elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="character" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="since" type="xs:date"/>
              <xs:element name="qualification" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Instanzdokument

```
<?xml version="1.0" encoding="utf-8"?>
<tns:book xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.upb.de/Sample   sample.xsd"
        xmlns:tns="http://www.upb.de/Sample"
        isbn="0836217462">
  <tns:title>Being a Dog Is a Full-Time Job</tns:title>
  <tns:author>Charles M. Schulz</tns:author>
  <tns:character>
    <tns:name>Snoopy</tns:name>
    <tns:since>1950-10-04</tns:since>
    <tns:qualification>extroverted beagle</tns:qualification>
  </tns:character>
  <tns:character>
    <tns:name>Peppermint Patty</tns:name>
    <tns:since>1966-08-22</tns:since>
    <tns:qualification>bold, brash and tomboyish</tns:qualification>
  </tns:character>
</tns:book>
```

Verwendung von "http://www.upb.de/Sample" als Default-Namensraum (Bindung an leeres Präfix) möglich, spart alle "tns:". Beachte, dass "isbn" ohne qulifizierendes Präfix als Attribut immer im anonymen Namensraum gesucht wird.

Legende: Name in Zielnamensraum Name in anon. Namnesraum