

# XML - Einführung

## Start

Allgemeines

Warum XML?

## Regeln

- Regeln I
- Regeln II
- Regeln III
- Regeln IV

XML abstrakt

Die X-Familie

Wohlgeformt&Gültig

Modell und Schema

XML als Baum

## XPath

- Allgemeines
- XPath-Achsen
- Syntax & Beispiele I
- Beispiele II

XML erzeugen

## Rekapitulation

- Fehlersuche
- Fragen & Antworten

Ende

## Allgemeines vorweg

Start

### Allgemeines

Warum XML?

Regeln

Regeln I

Regeln II

Regeln III

Regeln IV

XML abstrakt

Die X-Familie

Wohlgeformt&Gültig

Modell und Schema

XML als Baum

XPath

Allgemeines

XPath-Achsen

Syntax & Beispiele I

Beispiele II

XML erzeugen

Rekapitulation

Fehlersuche

Fragen & Antworten

Ende

- XML steht für: **eXtensible Markup Language**  
= erweiterbare Auszeichnungssprache
- XML ist eine Weiterentwicklung aus SGML
- XML ist eine Metasprache, die einen Satz Regeln vorgibt, nach denen Auszeichnungssprachen (ein Vokabular, ein „tag set“) entwickelt werden können
- XML ist wie HTML, nur anders
  - XML ist eine Metasprache (s.o.)
  - xHTML ist eine Anwendung von XML
  - Die Richtlinien der TEI sind eine Anwendung von XML
  - Beliebige lokale / eigene Vokabularien / Modelle können eine Anwendung von XML sein: Definiere deine eigenen Tags!

## Warum sollte man XML benutzen?

- Start
- Allgemeines
- Warum XML?**
- Regeln
  - Regeln I
  - Regeln II
  - Regeln III
  - Regeln IV
- XML abstrakt
- Die X-Familie
- Wohlgeformt&Gültig
- Modell und Schema
- XML als Baum
- XPath
  - Allgemeines
  - XPath-Achsen
  - Syntax & Beispiele I
  - Beispiele II
- XML erzeugen
- Rekapitulation
  - Fehlersuche
  - Fragen & Antworten
- Ende

- XML ist ein allgemeiner, offener Standard des W3C seit 1998
- XML ist unabhängig von Plattformen (Betriebssystemen) oder Programmen
- XML ist einfach
- XML ist „plain text“ (Zukunftssicherheit)
- XML ist flexibel
- XML ist dokumentnah
- XML beschreibt abstrakte Datenstrukturen
- XML ist mächtig (XML tut nichts, XML beschreibt Daten, *mit XML* kann man viel tun)
- XML umfasst eine ganze Familie von begleitenden Standards
- XML wird von einer breiten Softwarepalette unterstützt
- XML ist ein weit verbreiteter Standard für die Beschreibung und den Austausch von Daten. XML ist die Grundlage vieler Anwendungsstandards („XML is everywhere“)
- XML ist in zunehmendem Maße die Grundlage von Online-Ressourcen

## Ein paar grundsätzliche Regeln I

- Start
- Allgemeines
- Warum XML?
- Regeln**
  - Regeln I**
  - Regeln II
  - Regeln III
  - Regeln IV
- XML abstrakt
- Die X-Familie
- Wohlgeformt&Gültig
- Modell und Schema
- XML als Baum
- XPath
  - Allgemeines
  - XPath-Achsen
  - Syntax & Beispiele I
  - Beispiele II
- XML erzeugen
- Rekapitulation
  - Fehlersuche
  - Fragen & Antworten
- Ende

- Ein XML-Dokument besteht aus Textdaten und Tags. Es gibt namensgleiche öffnende (`<tag>`) und schließende (`</tag>`) Tags. Diese bilden zusammen mit den Textdaten ein „Element“. Es gibt auch leere Elemente (`<tag/>`). Elementnamen sind „case sensitive“ (`<tag>`, `<Tag>`).
- Tags werden in spitzen Klammern (`<, >`) notiert. Die spitzen Klammern dürfen in den Textdaten nicht vorkommen und werden durch Entities ersetzt: `&lt;` - `&gt;`; Das gleiche gilt für `&` → `&amp;`;

`<text>Textdaten. 10 &gt; 9</text><leeres_element/>`

## Ein paar grundsätzliche Regeln II

- Start
- Allgemeines
- Warum XML?
- Regeln**
  - Regeln I
  - Regeln II**
  - Regeln III
  - Regeln IV
- XML abstrakt
- Die X-Familie
- Wohlgeformt&Gültig
- Modell und Schema
- XML als Baum
- XPath
  - Allgemeines
  - XPath-Achsen
  - Syntax & Beispiele I
  - Beispiele II
- XML erzeugen
- Rekapitulation
  - Fehlersuche
  - Fragen & Antworten
- Ende

- Elemente können geschachtelt werden und müssen sauber geschachtelt sein. Überlappung ist streng verboten!
- Die geschachtelten Elemente bilden einen „Baum“
- Außer den eigentlichen XML-Elementen gibt es eine „xml declaration“ (<?xml ...>) und ggf. weitere „processing instructions“
- Ein XML-Dokument besteht aus einem Prolog und einem XML-Baum (einem – und nur einem! - Wurzelement mit seinen „Kindern“ bzw. „Nachkommen“)

```
<?xml version="1.0">
<vater>
  <kind>text <enkel>text</enkel> text</kind>
</vater>
```

## Ein paar grundsätzliche Regeln III

Start

Allgemeines

Warum XML?

**Regeln**

Regeln I

Regeln II

**Regeln III**

Regeln IV

XML abstrakt

Die X-Familie

Wohlgeformt&Gültig

Modell und Schema

XML als Baum

XPath

Allgemeines

XPath-Achsen

Syntax & Beispiele I

Beispiele II

XML erzeugen

Rekapitulation

Fehlersuche

Fragen & Antworten

Ende

- Elemente können Attribute enthalten. Attribute sind Paare aus Attributname und Attributwert. Der Attributwert steht in Anführungszeichen. Ein Element darf nicht zweimal das gleiche Attribut (Attributnamen) enthalten.

```
<vater hobby="Tennis" hobby2="Fußball">
  <kind>text <enkel>text</enkel> text</kind>
</vater>
```

## Ein paar grundsätzliche Regeln IV

- Start
- Allgemeines
- Warum XML?
- Regeln**
  - Regeln I
  - Regeln II
  - Regeln III
  - Regeln IV**
- XML abstrakt
- Die X-Familie
- Wohlgeformt&Gültig
- Modell und Schema
- XML als Baum
- XPath
  - Allgemeines
  - XPath-Achsen
  - Syntax & Beispiele I
  - Beispiele II
- XML erzeugen
- Rekapitulation
  - Fehlersuche
  - Fragen & Antworten
- Ende

- **Kommentare**
  - <!-- Dies ist ein Kommentar -->
- **Entities**
  - Entities sind Platzhalter für etwas anderes. Sie werden bei der Verarbeitung des XML-Dokuments durch dieses "andere" ersetzt
  - Syntax *&name;* - Beispiel *&gt;*; (für ">")
- **Namespaces**
  - Namensräume geben an, in welcher Domäne (und damit:in welchem Sinne) ein Elementname verwendet wird.
    - `<tei xmlns="http://www.tei-c.org/ns/1.0">...`
    - `<html xmlns="http://www.w3c.org/1999/xhtml">...`
    - `tei:body` ≠ `html:body`

## XML als Datenstruktur

- Start
- Allgemeines
- Warum XML?
- Regeln
  - Regeln I
  - Regeln II
  - Regeln III
  - Regeln IV
- XML abstrakt**
- Die X-Familie
- Wohlgeformt&Gültig
- Modell und Schema
- XML als Baum
- XPath
  - Allgemeines
  - XPath-Achsen
  - Syntax & Beispiele I
  - Beispiele II
- XML erzeugen
- Rekapitulation
  - Fehlersuche
  - Fragen & Antworten
- Ende

- XML-Daten können selbstbeschreibend sein
- XML-Daten sind sequentiell
- XML-Daten sind hierarchisch (Baum)
- XML-Daten können ein Netz sein
- XML kann rekursive Strukturen abbilden
- XML kann komplexe Strukturen abbilden
  - Datenzentriertes XML vs. Dokumentorientiertes XML
- XML kann ein Modell vorgeben (XML kann präskriptiv sein); ein XML-Modell kann allmählich auf bestehenden Daten entwickelt werden (XML kann deskriptiv sein)
- XML (und ein XML-Modell) kann die Semantik von Daten beschreiben. Die Semantik selbst kann aber kaum formal gefasst werden.
- Elemente und Attribute haben unterschiedliche Ausdrucksmöglichkeiten

## Die X-Familie

Start

Allgemeines

Warum XML?

Regeln

Regeln I

Regeln II

Regeln III

Regeln IV

XML abstrakt

**Die X-Familie**

Wohlgeformt&Gültig

Modell und Schema

XML als Baum

XPath

Allgemeines

XPath-Achsen

Syntax & Beispiele I

Beispiele II

XML erzeugen

Rekapitulation

Fehlersuche

Fragen & Antworten

Ende

- **XML** ... beschreibt strukturierte Daten
- **XPath** ... erlaubt die Navigation in XML-Daten
- **XMLSchema** ... beschreibt ein striktes Datenmodell
- **XSL** ... eXtensible Style Language
  - XSLT ... transformiert XML-Dokumente
  - XSL-FO ... beschreibt eine formatierte Ausgabe (z.B. für den Druck)
- **XLink** ... beschreibt komplexe Links
- **XPointer** ... beschreibt XML-Zeiger
- **XQuery** ... ist eine XML-Datenbankabfragesprache
- **XForms** ... beschreibt Eingabeformulare

## Wohlgeformt und gültig

Start

Allgemeines

Warum XML?

Regeln

Regeln I

Regeln II

Regeln III

Regeln IV

XML abstrakt

Die X-Familie

**WohlgeformtGültig**

Modell und Schema

XML als Baum

XPath

Allgemeines

XPath-Achsen

Syntax & Beispiele I

Beispiele II

XML erzeugen

Rekapitulation

Fehlersuche

Fragen & Antworten

Ende

- Man nennt ein XML-Dokument „wohlgeformt“ („well formed“), wenn es die Regeln von XML einhält. Ein wohlgeformtes Dokument kann geparsed, (z.B. mit XPath) navigiert und mit anderen XML-Standards verarbeitet werden

- Man nennt ein XML-Dokument „gültig“ („valid“), wenn es den Regeln eines Schemas entspricht. Strukturen und Inhalte eines gültigen XML-Dokuments können *leichter* kontrolliert, konsistent gehalten und verarbeitet werden. Ein Schema beschreibt ein Modell **explizit**.

## DTD als Beispiel für ein Schema

- Start
- Allgemeines
- Warum XML?
- Regeln
  - Regeln I
  - Regeln II
  - Regeln III
  - Regeln IV
- XML abstrakt
- Die X-Familie
- Wohlgeformt&Gültig
- Modell & Schema**
- XML als Baum
- XPath
  - Allgemeines
  - XPath-Achsen
  - Syntax & Beispiele I
  - Beispiele II
- XML erzeugen
- Rekapitulation
  - Fehlersuche
  - Fragen & Antworten
- Ende

- Es gibt verschiedene "Schema-Sprachen"
  - z.B. DTD (Document Type Definition), XML Schema, RelaxNG (REgular LAnguage description for Xml New Generation)
- Ein Schema beschreibt, welche Elemente und Attribute (ggf. Attributwerte) es in einem Modell gibt und welche Elemente wo (worin) vorkommen dürfen ("Inhaltsmodelle")
- Beispiel:
 

```

<!ELEMENT vater (kind)>
<!ATTLIST vater
  hobby CDATA #IMPLIED
  hobby2 CDATA #IMPLIED>
<!ELEMENT kind (#PCDATA | enkel)* >
<!ELEMENT enkel (#PCDATA) >
      
```
- ... aber wo kriege ich meine DTD eigentlich her?

## XML ist ein Baum!

- Start
- Allgemeines
- Warum XML?
- Regeln
  - Regeln I
  - Regeln II
  - Regeln III
  - Regeln IV
- XML abstrakt
- Die X-Familie
- Wohlgeformt&Gültig
- Modell und Schema
- XML als Baum**
- XPath
  - Allgemeines
  - XPath-Achsen
  - Syntax & Beispiele I
  - Beispiele II
- XML erzeugen
- Rekapitulation
  - Fehlersuche
  - Fragen & Antworten
- Ende

- Der Baum hat ein (!) Wurzel-Element und weitere Kindelemente (die wieder Kindelemente haben). Die Elemente im Baum und andere definierte Einheiten nennt man auch "Knoten" ("nodes").
- Alles ist ein Knoten. Es gibt verschiedene Knoten-Typen (z.B. Processing-Instruction, Element-Knoten, Attribut-Knoten, Text-Knoten)
- In der Navigation werden einzelne Elemente oder Knoten oder Sätze von Knoten ("node sets") angesteuert.

- Wie viele Elemente, wie viele Knoten sind das:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE vater SYSTEM "Vater-Kind.dtd">
<vater id="v0001">
    <kind>text <enkel>text</enkel> text</kind>
</vater>
```

## XPath - allgemeines

- Start
- Allgemeines
- Warum XML?
- Regeln
  - Regeln I
  - Regeln II
  - Regeln III
  - Regeln IV
- XML abstrakt
- Die X-Familie
- Wohlgeformt&Gültig
- Modell und Schema
- XML als Baum
- XPath**
  - Allgemeines**
  - XPath-Achsen
  - Syntax & Beispiele I
  - Beispiele II
- XML erzeugen
- Rekapitulation
  - Fehlersuche
  - Fragen & Antworten
- Ende

- XPath bietet eine Syntax, um Elemente, Knoten und Knoten-Sätze in einem XML-Dokument anzusteuern
- Die Adressierung von Elementen kann mit Bedingungen verknüpft werden
- XPath stellt weitere Funktionen zur Verfügung um z.B. Knotensätze zu zählen, Positionen abzufragen oder Zeichenketten (strings) zu bearbeiten
- XPath-Ausdrücke können absolute oder relative Pfade beschreiben. Relative Pfade sind abhängig von der gegenwärtigen Position (dem "selbst" bzw. "Kontextknoten")

## XPath - Achsen

Start

Allgemeines

Warum XML?

Regeln

Regeln I

Regeln II

Regeln III

Regeln IV

XML abstrakt

Die X-Familie

Wohlgeformt&Gültig

Modell und Schema

XML als Baum

**XPath**

Allgemeines

**XPath-Achsen**

Syntax & Beispiele I

Beispiele II

XML erzeugen

Rekapitulation

Fehlersuche

Fragen & Antworten

Ende

- In der Regel steuert man Elemente über ihre Eltern-Kind-Beziehung und ihre Namen an. Oft ist ihre Position aber genauer anzugeben. Dabei helfen die XPath-Achsen
- parent / ancestor / ancestor-or-self
- child / descendant / descendant-or-self
- preceding / preceding-or-self / preceding-sibling
- following / following-or-self / following-sibling
- attribute

## XPath – Syntax und Beispiele I

- Start
- Allgemeines
- Warum XML?
- Regeln
  - Regeln I
  - Regeln II
  - Regeln III
  - Regeln IV
- XML abstrakt
- Die X-Familie
- Wohlgeformt&Gültig
- Modell und Schema
- XML als Baum
- XPath**
  - Allgemeines
  - XPath-Achsen
  - Syntax & Beispiele I**
  - Beispiele II
- XML erzeugen
- Rekapitulation
  - Fehlersuche
  - Fragen & Antworten
- Ende

/	→ das Dokument; die Wurzel
.	→ das Selbst; der Kontextknoten
*	→ ein beliebiger Name
	→ eine oder-Verknüpfung
@...	→ ein Attribut
[...]	→ eine Bedingung
[not (...)]	→ eine verneinte Bedingung

- `enkel` → Kindelemente des Kontextknotens, die auf den Namen "enkel" hören
- `//enkel` → Elemente mit dem Namen "enkel" in beliebiger Tiefe des Baumes (unterhalb des Kontextknotens)
- `../vater` → Elemente mit Namen "vater" eine Hierarchiestufe über dem Kontextknoten
- `ancestor::vater` → Elemente mit Namen "vater" beliebig viele Hierarchiestufen über dem Kontextknoten
- `preceding::kind` → Elemente mit dem Namen "kind", die im XML-Dokument vor dem Kontextknoten kommen

## XPath – Beispiele II

Start

Allgemeines

Warum XML?

Regeln

Regeln I

Regeln II

Regeln III

Regeln IV

XML abstrakt

Die X-Familie

Wohlgeformt&Gültig

Modell und Schema

XML als Baum

**XPath**

Allgemeines

XPath-Achsen

Syntax & Beispiele I

**Beispiele II**

XML erzeugen

Rekapitulation

Fehlersuche

Fragen & Antworten

Ende

- @name → der Wert des Attributs "name" des Kontextknotens
- @\* → alle Attribute des Kontextknotens
- kind[enkel] → ein Kindelement "kind", das ein Kindelement "enkel" hat
- //kind[@n='3'] → ein Kindelement "kind" in beliebiger Tiefe des Baums mit einem Attribut "n" mit einem Attributwert "3"
- p[contains(person,'Walter')] → ein Kindelement "p", das ein Kindelement "person" hat, das die Zeichenkette "Walter" enthält
- count(//kind) → die Anzahl der Elemente "kind" in beliebiger Tiefe des Baumes
- name(\*) → die Namen aller Kindknoten
- //vater[not (contains(@hobby,'ball')) and not (contains(@hobby2,'ball'))] → alle Väter, die keine Hobbies mit "ball" haben

## Wie entsteht eigentlich mein XML?

- Start
- Allgemeines
- Warum XML?
- Regeln
  - Regeln I
  - Regeln II
  - Regeln III
  - Regeln IV
- XML abstrakt
- Die X-Familie
- Wohlgeformt&Gültig
- Modell und Schema
- XML als Baum
- XPath
  - Allgemeines
  - XPath-Achsen
  - Syntax & Beispiele I
  - Beispiele II
- XML erzeugen**
- Rekapitulation
  - Fehlersuche
  - Fragen & Antworten
- Ende

- **Selber schreiben:**  
Text und Textauszeichnungen werden von Hand eingegeben
- **Vorhandenen Text auszeichnen:**  
Text liegt bereits elektronisch vor und wird sukzessive mit Tags angereichert
- **Übersetzung von typografischen oder Textstrukturen in XML:**  
Formatierungen und Schlüsselbegriffe werden gedeutet und zu expliziten XML-Angaben umgeformt

# Rekapitulation I

Start

Allgemeines

Warum XML?

Regeln

Regeln I

Regeln II

Regeln III

Regeln IV

XML abstrakt

Die X-Familie

Wohlgeformt&Gültig

Modell und Schema

XML als Baum

XPath

Allgemeines

XPath-Achsen

Syntax & Beispiele I

Beispiele II

XML erzeugen

**Rekapitulation**

**Fehlersuche**

Fragen & Antworten

Ende

- Was stimmt hier nicht?

```
<xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE vater SYSTEM "Vater-Kind.dtd">
<vater name=Karl Ludolf Friedrich Lachmann>
  <#kind>Karl <zweiter name>Konrad</enkel> Lachmann</#kind>
</vater>
<vater name="Karl Ludwig Böhmer" name="Boehmer">
  <kind>Johann <2terName>Friedrich<nickname>Regestenböh-
mer</2terName> </nickname> Böhmer</kind>
</vater>
```

## Rekapitulation II

Start

Allgemeines

Warum XML?

Regeln

Regeln I

Regeln II

Regeln III

Regeln IV

XML abstrakt

Die X-Familie

Wohlgeformt&Gültig

Modell und Schema

XML als Baum

XPath

Allgemeines

XPath-Achsen

Syntax & Beispiele I

Beispiele II

XML erzeugen

**Rekapitulation**

Fehlersuche

**Fragen & Antworten**

Ende

- Was ist der Unterschied zwischen HTML, TEI und XML?
- Wer steht hinter XML?
- Was können Elemente enthalten?
- Was können Attribute enthalten?
- Was sind die Vor- und Nachteile von Elementen und Attributen?
- Wie wird ein leeres Element notiert?
- Spielt die Reihenfolge von Elementen eine Rolle?
- Spielt die Reihenfolge von Attributen eine Rolle?
- Ist XML präskriptiv oder deskriptiv?
- Was bedeutet "wohlgeformt" / "well formed"?
- Was bedeutet "gültig" / "valid"?
- Welche Funktion haben Schemata?
- Was können Schemata kontrollieren?
- Welche Funktion hat XPath?
- Welche XPath-Achsen kennen Sie?
- Was steuern Sie hiermit an: `//index[@type='person']`
- Was wird hier zurück geliefert: `count(//@hobby | //@hobby2)`

## Rekapitulation II

Start

Allgemeines

Warum XML?

Regeln

Regeln I

Regeln II

Regeln III

Regeln IV

XML abstrakt

Die X-Familie

Wohlgeformt&Gültig

Modell und Schema

XML als Baum

XPath

Allgemeines

XPath-Achsen

Syntax & Beispiele I

Beispiele II

XML erzeugen

Rekapitulation

Fehlersuche

Fragen & Antworten

Ende

... noch Fragen ... ?

... Meinungen ... ?

... Widerspruch ... ?

... Pause ... ?