Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

DARIAH-DE
Digital Research Infrastructure
for the Arts and Humanities

Referat 11:
Einführung in XPath

DARIAH-DE Tutorial *Digitale Textedition mit TEI*

Redaktion: Christof Schöch (Univ. Würzburg)

Version 1.0, 02/2014

Grundlage der Folien: DH@Oxford 2012

Überblick

Dieses Input-Referat beinhaltet:

1. Was ist XPath? XPath im Kontext

2. XPath „sieht“ XML als Baumstruktur

3. XPath-Ausdrücke
   - Achsen
   - Kontext
   - Knoten-Tests
   - Prädikate
   - Funktionen

Was ist XPath?

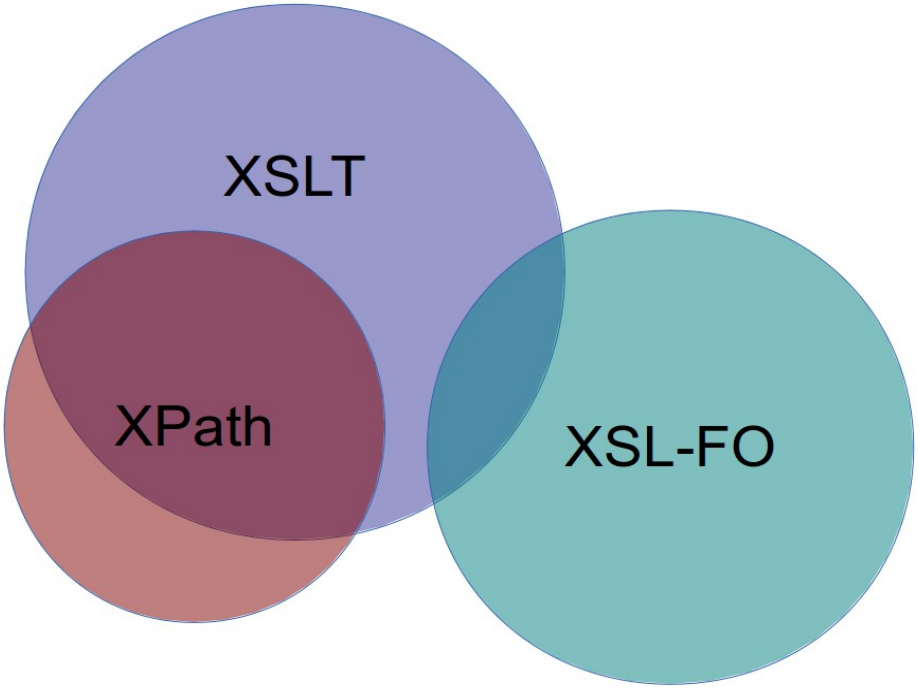
Grundidee

- "XPath is a language for addressing parts of an XML document (XPath Specifications)
- XPath wird in XSLT und XPointer eingesetzt

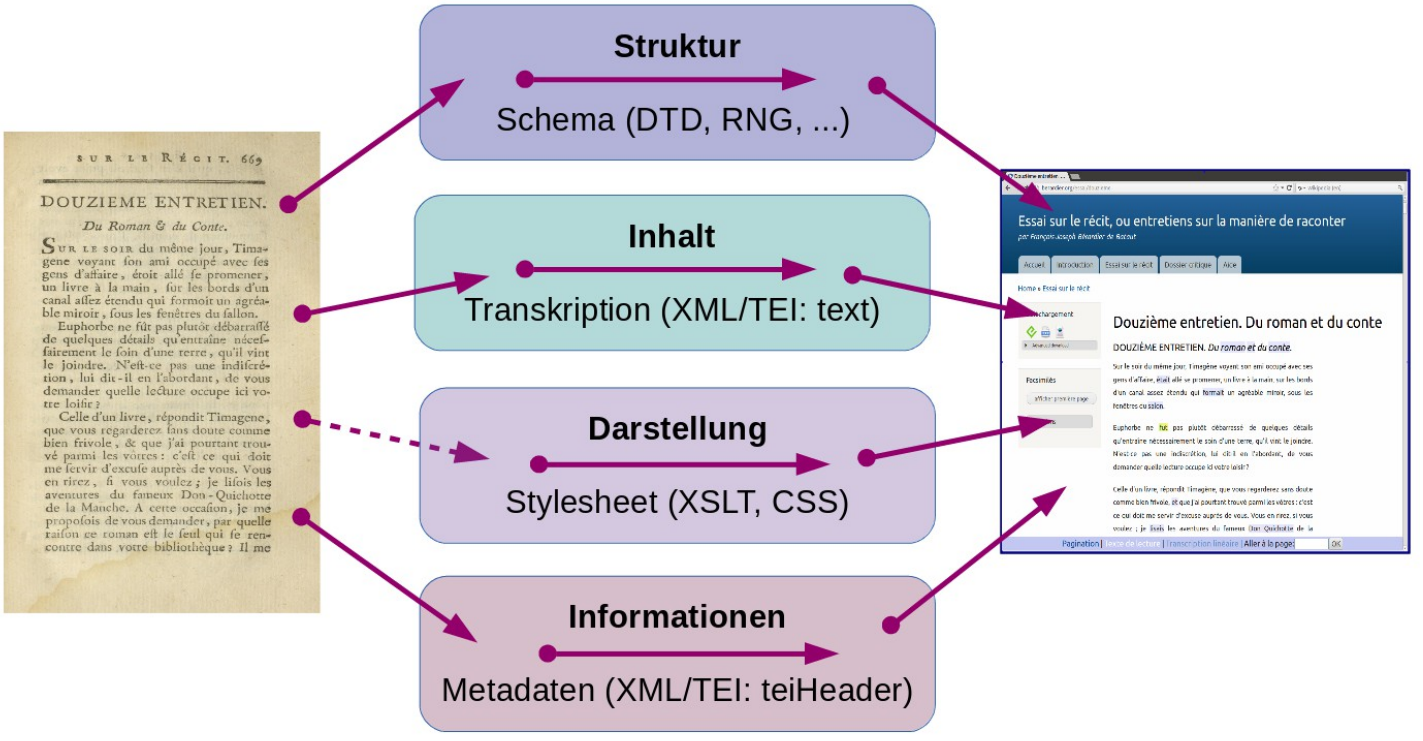
Spezifikation

- Die XPath-Spezifikation wird vom W3C (World Wide Web Consortium) gepflegt
- http://www.w3.org/TR/xpath/

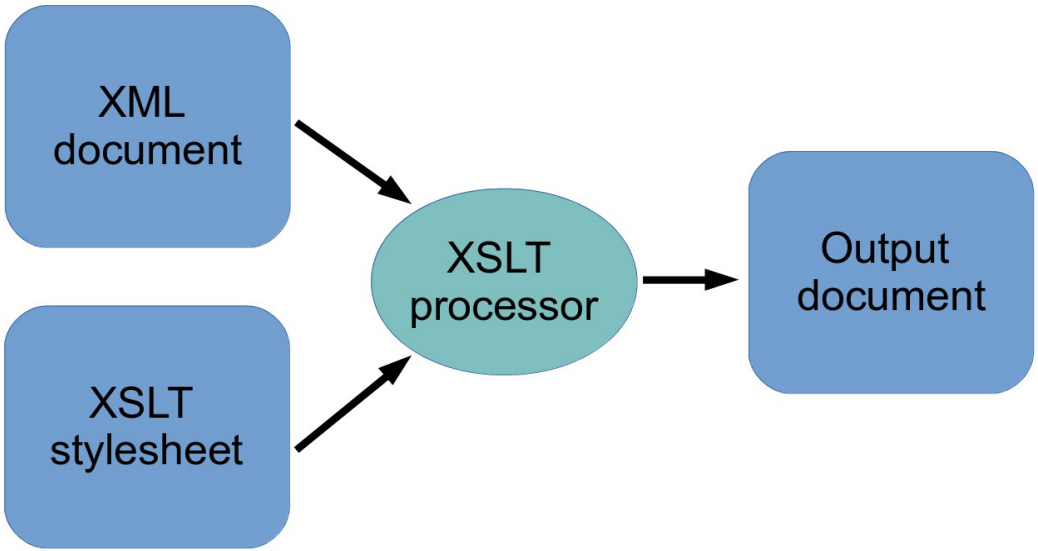
XPath im Kontext (1)



XPath im Kontext (2)



XSLT-Transformation



XPath in XSLT

XPath kommt in XSLT-Elementen als spezielles Attribut vor

```
<xsl:for-each select="[XPath]"/>

<xsl:template match="[XPath]"/>

<xsl:value-of select="[XPath]"/>
```

Beispiel:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="2.0">

<xsl:template match="TVGuide/Channel">
  <xsl:apply-templates select="Name"/>
  <xsl:apply-templates select="Program"/>
</xsl:template>

</xsl:stylesheet>
```

XPath: XML als Baumstruktur

XPath sieht XML-Dokumente als Baumstruktur an

- hierarchische Beziehungen in XML
- Knoten bilden Äste, Verzweigungen, Blätter
- Eltern, Kinder, Geschwister, etc.

Sieben (acht) Verschiedene Arten von Knoten

- (Dokument-Knoten)
- Wurzel-Knoten
- Element-Knoten
- Attribut-Knoten
- Text-Knoten
- Kommentar-Knoten
- Processing Instructions-Knoten
- Namespace-Knoten

Knoten (1)

Dokument-Knoten

- entspricht dem gesamten Dokument
- Wurzel-Element + vorhergehende Processing Instructions und Kommentare

Wurzel-Knoten

- entspricht dem Wurzel-Element
- gesamtes Dokument ohne vorhergehende PI oder Kommentare

Element-Knoten

- beinhalten das gesamte Element
- mit allen Unterelementen, Attributen, Text, usw.

Knoten

Attribut-Knoten

- Beinhaltet ein Attribut einschließlich seines Werts
- Der Element-Knoten ist "parent" des Attributs, aber das Attribut ist kein "child" des entsprechenden Element-Knotens (!!)

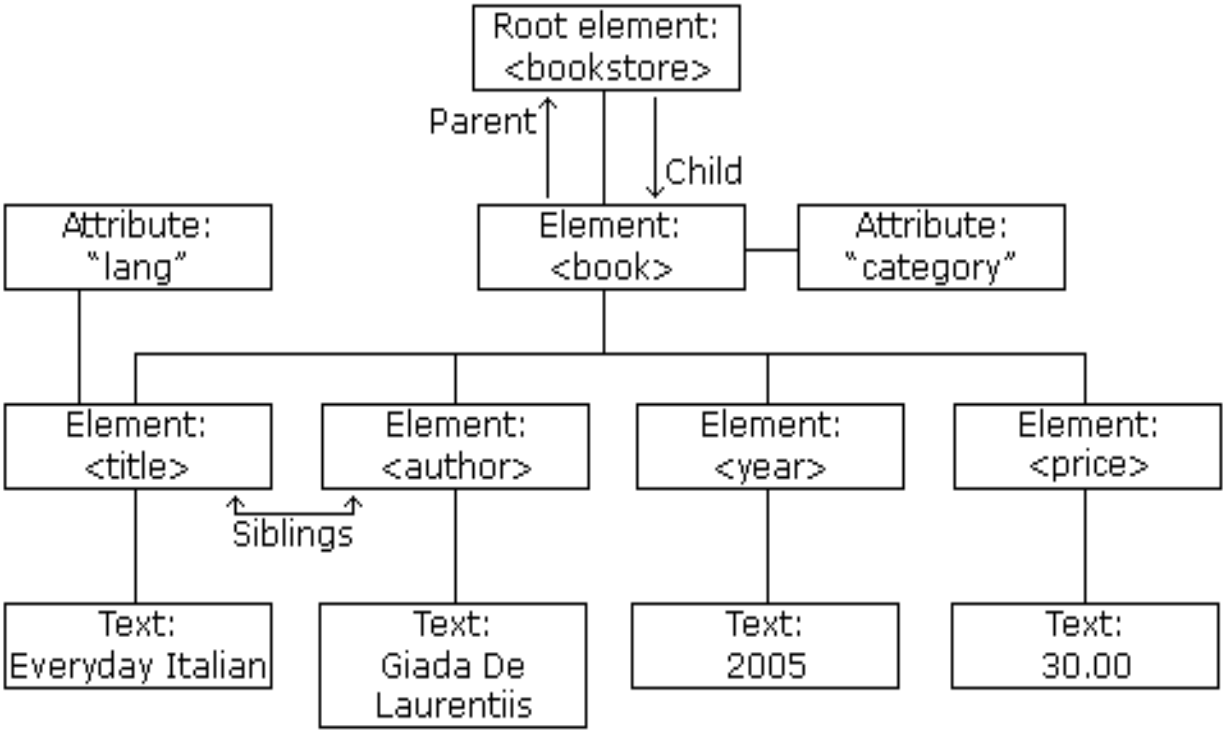
Text-Knoten

- Aller "plain text" innerhalb eines Elements
- Entities werden vorab aufgelöst (und sind nicht als solche ansprechbar)

XML-Baumstruktur (1 - Dokument)

```
<bookstore>
  <book category="COOKING">
     <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>Joan K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
</bookstore>
```

XML-Baumstruktur (1 - Visualisierung)

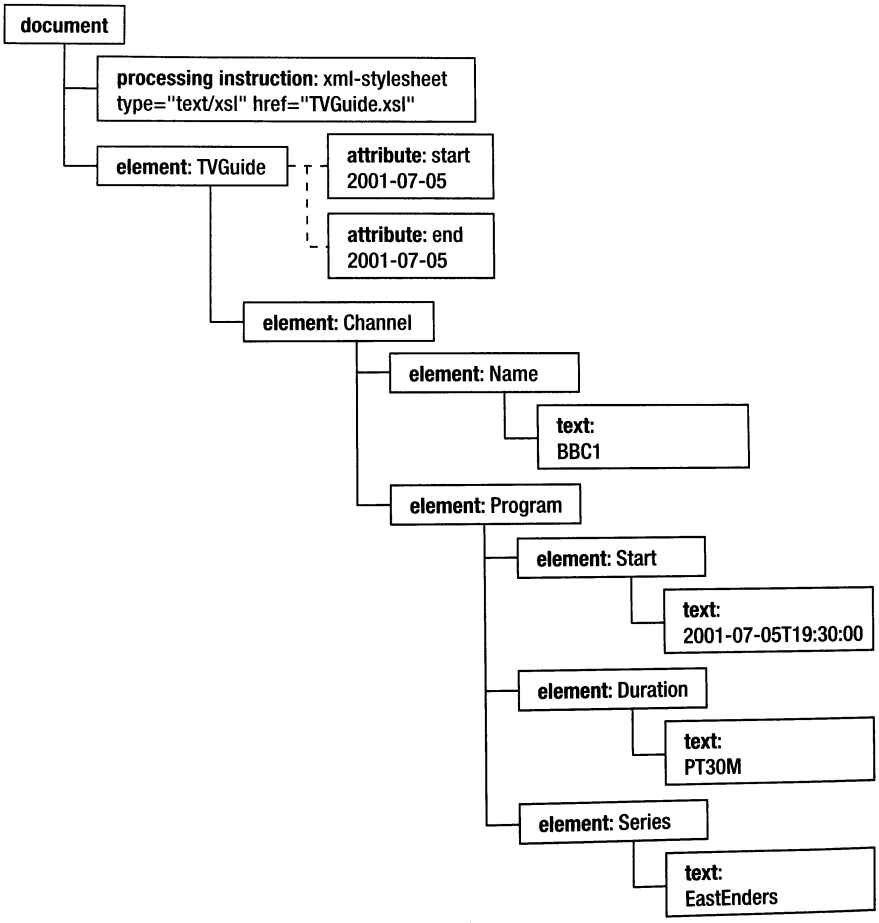


XML-Baumstruktur (2 - Dokument)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="TVGuide.xsl"?>
<TVGuide start="2001-07-05" end="2001-07-05">
 <Channel>
  <Name>BBC1</Name>
  <Program>
   <Start>2001-07-05T19:00:00</Start>
   <Duration>PT30M</Duration>
   <Series>QuestionOfSport</Series>
   <Title></Title>
  </Program>
  <Program rating="5" flag="favorite">
   <Start>2001-07-05T19:30:00</Start>
   <Duration>PT30M</Duration>
   <Series>EastEnders</Series>
  </Program>
 </Channel>
</TVGuide>
```

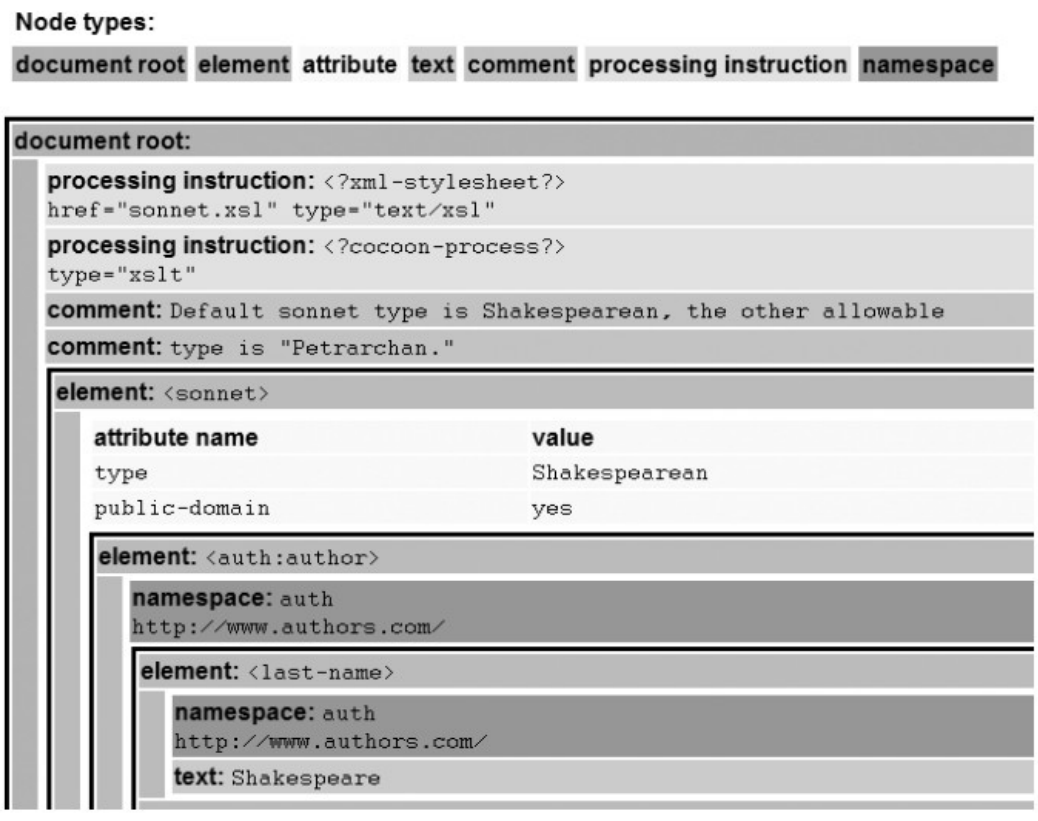
XML-Baumstruktur (2 – Visualisierung)

(Tennison 2005)



XPath: Baumstruktur (inkludierend)

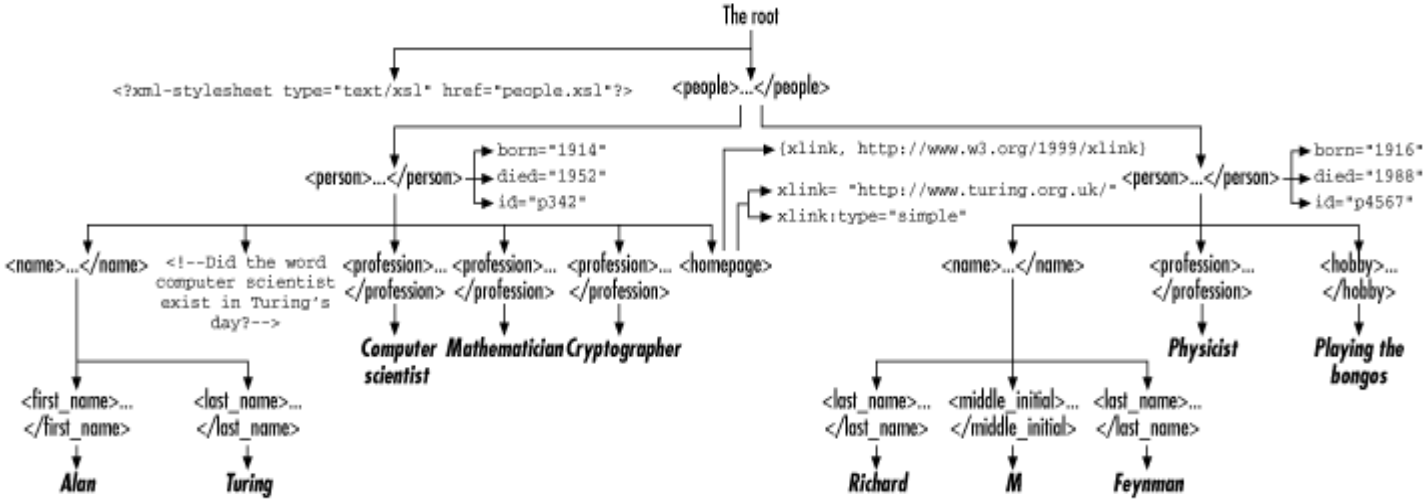
(Tidwell 2009)



XML-Baumstruktur (3 - Dokument)

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="people.xsl"?>
<people>
  <person born="1912" died="1954" id="p342">
    <name>
      <first_name>Alan</first_name>
      <last_name>Turing</last_name>
    </name>
    <!--Did the word computer scientist exist in Turing's day? -->
    <profession>computer scientist</profession>
    <profession>mathematician</profession>
    <profession>cryptographer</profession>
    <homepage xlink:href="http://www.turing.org.uk/"/>
  </person>
  <person born="1918" died="1988" id="p4567">
    <name>
      <first_name>Richard</first_name>
      <middle_initial>&#x4D;</middle_initial>
      <last_name>Feynman</last_name>
    </name>
    <profession>physicist</profession>
    <hobby>Playing the bongoes</hobby>
  </person>
</people>
```

XLM-Baumstruktur (3 - Visualisierung)



XPath-Grundlagen

Zwei Funktionen

- Verweis auf bestimmte Teile eines XML-Dokuments (location reference)
- Teil eines Dokuments auf Übereinstimmung mit Muster überprüfen ("pattern matching")

Zwei Typen von Ausdrücken

- Lokalisierungs-Schritt: ein einzelnes Element (location step)
- Lokalisierungs-Pfad: ein oder mehrere location steps zusammen (location path)

Zwei Typen von Pfaden

- absolute Pfade: gehen vom Dokument-Knoten aus
- relative Pfade: gehen vom aktuellen Kontext aus

XPath-Grundlagen

Lokalisierungs-Schritte

- Prinzip: **Achse + Knotenprüfung + Prädikat** (Prädikat ist optional)
- Syntax: **achse::knotenprüfung[Prädikat]**

Lokalisierungs-Pfad

- besteht aus einem oder mehreren "location steps"
- Definieren Pfad zu einem Teil des XML-Dokuments
- Werden geprüft oder evaluiert

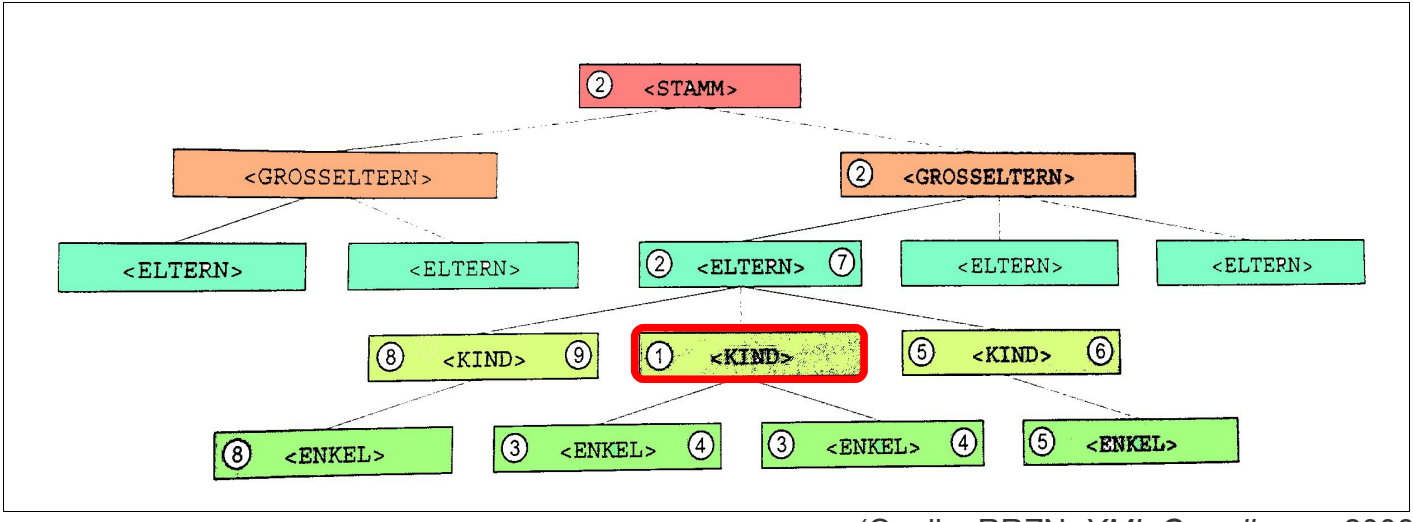
Zwei Schreibweisen

- Ausführliche Schreibweise: "//child::title" oder "child::div/child::p"
- Abgekürzte Schreibweise: "//title" oder "//div/p"
  (Default ist "child")

Kontext

XPath-Ausdrücke werden relativ zu einem Kontext geprüft

- der Kontext ist die Position im Baum, an der der Parser gerade ist
- der Kontext verändert sich ständig!



(Quelle: RRZN, *XML Grundlagen*, 2008)

Kontext

Achsen und Kontext im obigen Beispiel

- self > (1)
- ancestor > (2)
- ancestor-or-self > (2) (1)
- child > (3)
- descendant > (4)
- descendant-or-self > (4) (1)
- following > (5)
- following-sibling > (6)
- parent > (7)

XPath: Achsen (1)

Mit den „Achsen“ kann man einen bestimmten Teil (Knoten) des XML-Baums ansprechen.

- relativ zum Kontext-Knoten
- gibt eine „Richtung“ an

Achsen	Abk.	Funktion: selektiert...
• child::		die direkten Kinder des Kontext-Knotens
• parent::	..	die direkten Eltern des Kontext-Knotens
• self::	.	den Kontext-Knoten selbst
• ancestor::		alle Eltern und Groß-Eltern etc.
• descendant::		alle Kinder und Kindes-Kinder etc. (ohne Attribute!)

XPath: weitere Achsen

Achsen		Funktion: selektiert...
• ancestor-or-self::		den Kontext-Knoten und alle Knoten weiter oben in der Baumstruktur, bis zum Wurzel-Knoten
• descendant-or-self::		den Kontext-Knoten und alle Knoten weiter unten in der Baumstruktur
• attribute	@	die Attribute des Kontext-Knotens

Knoten-Tests

Mit "Knoten-Tests" kann man bestimmte Teile eines Knotens ansprechen

- Beispiel: Ein Element-Knoten beinhaltet auch einen Attribut-Knoten und einen Text-Knoten

```
<title rend="italic" xml:id="123">XML in a Nutshell</title>
```

- `//title` spricht den gesamten Knoten an
- `//title/attribute()` spricht alle Attribut-Knoten in "title" an [2.0]
- `//title/text()` spricht den Text-Knoten in "title" an

Vergleich

- `//title/@xml:id` spricht direkt den Attribut-Knoten "xml:id" an

Liste der Knoten-Tests

node()

comment()

text()

processing-instruction()

Prädikate: weitere Selektion

XPath-Ausdrücke, die mehrere Knoten ansprechen, kann man bspw. mit Zahlenwerten präzisieren

Beispiel:

```
...
<div>
  <head>Titel des ersten Kapitels</head>
  <p>Text des ersten Absatzes.</p>
  <p>Text des zweiten Absatzes.</p>
  <p>Text des dritten Absatzes.</p>
</div>
<div>
  <head>Titel des zweiten Kapitels</head>
  <p>Text des ersten Absatzes.</p>
  <p>Text des zweiten Absatzes.</p>
  <p>Text des dritten Absatzes.</p>
</div>
...
```

Teile der Knoten-Menge: Prädikate

- `//div` alle Element-Knoten "div" (kein Prädikat)
- `//div[1]` der erste Element-Knoten "div"
- `//div/p` alle Element-Knoten "p", die Kinder von allen Element-Knoten "div" sind (kein Prädikat)
- `//div[1]/p` alle Element-Knoten "p", die Kinder des ersten Element-Knoten "div" sind
- `//div[2]/p[3]` der dritte Element-Knoten "p", der Kind des zweiten Element-Knotens "div" ist.
- `//div[2]/p[3]/text()` den Text-Knoten des dritten... (etc.)

Prädikate: Musik-Beispiel (XML)

```
21  ...
22  <album type="hardbop">
23    <titel>Ready for Freddie</titel>
24    <interpret>Freddie Hubbard</interpret>
25  </album>
26  <album type="hardbop">
27    <titel>Giant Steps</titel>
28    <interpret>John Coltrane</interpret>
29  </album>
30  <album type="bebop">
31    <titel>Soultrane</titel>
32    <interpret>John Coltrane</interpret>
33  </album>
34  ...
```

Prädikate: Musik-Beispiel (XPath)

Knoten-Test kombiniert mit Wert

```
//album/titel[text()=“Ready for Freddie“]
```

Ergebnis: Knoten in Zeile 23.

```
//album/interpret[text()=“John Coltrane“]
```

Ergebnis: Knoten in Zeile 28 und 32

```
//album[attribute()=“hardbop“]
```

Ergebnis: Knoten in Zeile 22 bis 29

--- der Knoten-Test attribute() ist nur ab XPath 2.0 verfügbar ---

Funktionen in Prädikaten

XPath beinhaltet mehrere Funktionen, die innerhalb von Prädikaten angewandt werden können

last()

- Wählt statt einer bestimmten Position den letzten Knoten aus
- Beispiel: `//div[2]/p[last()]`
- Ergebnis: der letzte Absatz im zweiten Abschnitt

Attribut-Wert

- Einen Element-Knoten, der für ein bestimmtes Attribut einen bestimmten Wert hat:
- Abstrakt: `{Element}[@{Attribut}="{Wert}"]`
- Konkret: `l[@rhyme="b"]`
- Ergebnis: alle Zeilen ("l") mit Attribut "rhyme" und Reim-Position "c"

Funktionen in Prädikaten (2)

Zählen: count()

- Gibt die Anzahl der Knoten (etc.) aus, die eine bestimmte Definition erfüllen
- Abstrakt: `count({Definition})`
- Konkret: `count(//l[@rhyme="b"])`
- Ergebnis: Die Anzahl der Zeilen, mit Attribut "rhyme" und Reim-Position "b"
- Oder: count(/TEI/body/div/p) => Anzahl der "p"s im Dokument

Also: XPath-Ausdrücke werden „evaluiert“

Evaluation kann verschiedene Ergebnis-Formen haben

„Klassischer Fall“

- einen oder mehrere Knoten (ein "node-set")

Aber auch:

- bool'scher Ausdruck: "true" oder "false"
- Zahl ("floating-point number")
- Zeichenfolge ("string")

Weitere Möglichkeiten von XPath

Mathematische Operationen

- Bspw.: Die Ergebnisse von count() addieren, multiplizieren, etc.

Data types (nur mit „schema-aware processors“)

- XPath 1.0: Knoten-Menge, bool'scher Wert, Zahl, String
- XPath 2.0: weitere Datentypen, wie Datum, Dauer, QName, anyURI, ...

Bool'sche Operatoren

- Ist-Gleich, Ist-Nicht-Gleich, und, oder, größer-als, kleiner-als, ...

XPath 2.0: "Conditional Expressions"

- if, then, else