

XML als Datenstruktur

XPath

Wuppertal, 11.2.2020

Wer hat Spaß an purer Logik?

Übersicht

- Wozu XPath?
- XPath als Standard
- Bäume, Hierarchien, Schachteln
- Konzepte und Grundbausteine
- Üben, Üben, Üben

Was ist XPath?

- "XPath is a language for addressing parts of an XML document" (XPath Specifications)
- XPath dient der Navigation in XML-Dokumenten und der Erzeugung von "Rückgaben"
- XPath wird vor allem in anderen X-Technologien verwandt: XSLT, XQuery

Was ist XPath?

- [XPath](#) ist ein W3C-Standard
- [XPath 1.0](#) – 1999
- [XPath 2.0](#) – 2010
- [XPath 3.1](#) – 2017
- Unterstützung durch andere Technologien?
- Was davon brauchen Sie?

Warum brauche ich XPath?

- XPath wird vor allem in anderen X-Technologien verwandt: XSLT, Xquery
- Selbst wenn Sie diese Technologien nicht selbst einsetzen, wollen Sie Sich in Daten zurechtfinden, Dinge finden, Sachen prüfen, Kennzahlen ermitteln etc. XPath verschafft Ihnen Durchblick durch die Daten

Wo findet man XPath?

- Einfacher Einstieg: https://www.w3schools.com/xml/xpath_intro.asp
 - dort auch eine einfache Referenz
- Einfache Referenz: https://www.w3schools.com/xml/xsl_functions.asp
(nur bis 2.0!)
- Vollständige Referenz: <https://maxtoroq.github.io/xpath-ref/>

Einfacher Einstieg

edition.onb.ac.at/okopenko/context:okopenko/methods/sdef:Context/get

TAGEBÜCHER
ANDREAS OKOPENKO

Österreichische
Nationalbibliothek

Startseite Projekt ▾ Okopenko Tagebücher Themen ▾ Register ▾ Zusatzmaterialien

Digitale Edition

gegenwärtiger Stand:
Ich überleg mir jedes Wort.
dreihundertmal, eh ich's aussprech.
Ich denke aber, dass die Worte,
die die 400ste Überlegung
durchhalten, noch besser
würden
wären

Andreas Okopenko (1930–2010) spielte schon als junger Autor eine wichtige Rolle im Wiener Literaturbetrieb der 1950er Jahre und erwies sich als herausragender Netzwerker der literarischen Avantgarde Österreichs. In seinen Tagebüchern lässt sich der frühe Schreib- und Schaffensprozess anhand von poetologischen Reflexionen und literarischen Textentwürfen nachvollziehen. Die Tagebuchaufzeichnungen enthalten zudem eine Vielzahl an Kommentaren zum österreichischen Literaturbetrieb und zur Zeitgeschichte. Die vorliegende [digitale Edition](#) umfasst 29 Tagebücher (1949–1954) aus dem [Nachlass](#), der 2012 vom Literaturarchiv der Österreichischen Nationalbibliothek erworben wurde. Roland Innerhofer, Bernhard Fetz und ihr [Team](#) verantworten die digitale Edition.

Tagebücher Andreas Okopenko. Digitale Edition.

HerausgeberInnen: Roland Innerhofer, Bernhard Fetz, Christian Zolles, Laura Tezarek, Arno Herberth, Desiree Hebenstreit, Holger Englerth, Österreichische Nationalbibliothek und Universität Wien, Wien 2019. [↗](#)

XML als Datenstruktur: XPath
Patrick Sahle



Leopoldina
Nationale Akademie
der Wissenschaften



BERGISCHE
UNIVERSITÄT
WUPPERTAL

<https://edition.onb.ac.at/okopenko/context:okopenko/methods/sdef:Context/get>

Einfacher Einstieg

Persons.xml [C:\PatosWelt\Arbeiten\UniWuppertal\ScieboCloud\Veranstaltungen\WinterSchool2020\Übungsdaten\XPath-Übung\Persons.xml] - <oxygen/> XML Editor (Ausschließlich akademische Nutzung)

Datei Bearbeiten Suchen Projekt Optionen Werkzeuge Dokument Fenster Hilfe

XPath 2.0 `/TEI/text/body/div/listPerson/person`

• ZK_1_01_05_073_V_N_NB_6-3.xml* × • Persons.xml ×

TEI text body div listPerson person

```
1 <?xml version='1.0' encoding='UTF-8' standalone='yes'?>
2 <TEI xmlns="http://www.tei-c.org/ns/1.0" xml:id="aok_REG_Persons">
3   <teiHeader> [48 lines]
52   <text>
53     <body>
54       <div>
55         <head>Personenregister</head>
56         <p>Das Personenregister enthält alle Personen, die in den edierten Tagebüchern von Andrea
57           Okopenko genannt sind. Es sind sowohl reale als auch fiktive Personen erfasst. </p>
58         <listPerson xml:id="okopenko_persons">
59           <person xml:id="Absolon_Kurt" role="Bildende_KünstlerInnen" cert="high"> [13 lines]
73           <person xml:id="Adenauer_Konrad" role="PolitikerInnen" cert="high"> [15 lines]
89           <person xml:id="Adler_Alfred" role="PsychologInnen" cert="high"> [15 lines]
105          <person xml:id="Aichinger_Ilse" role="SchriftstellerInnen" cert="high">
106            <persName ref="http://d-nb.info/gnd/118501232">
107              <reg>Aichinger, Ilse</reg>
108              <surname>Aichinger</surname>
```

Text Raster Autor

Resultate

Beschreibung - 680 Elemente

	XPath Location	Ressource	Sy
cert="high" role="Bildende_KünstlerInnen" xml:id="Absolon_Kurt"	/TEI[1]/text[1]/body[1]/div[1]/listPerson[1]/person[1]	Persons.xml	C

Schachteln, Hierarchien, Bäume

Sie haben es schon gelernt:

- Text ist sequentiell
- Tags + Inhalt = Elemente
- Elemente in Elementen = Verschachtelung
- Ein äußerstes Element = Eine Hierarchie!
- Ein „Baum“?

XML als Baum



XML als Datenstruktur: XPath
Patrick Sahle

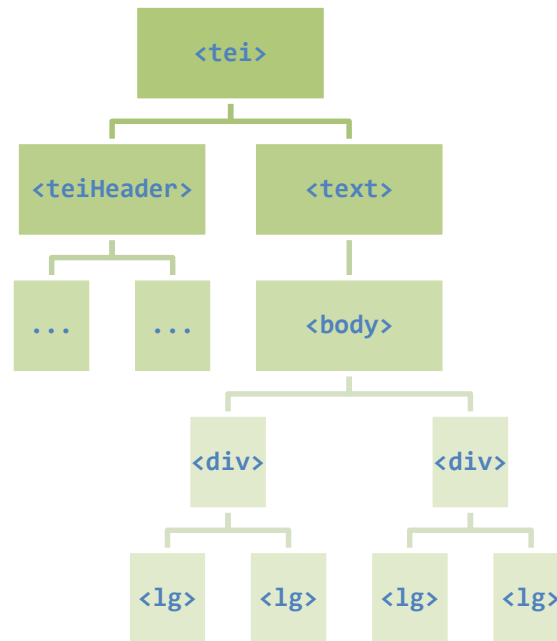


Leopoldina
Nationale Akademie
der Wissenschaften



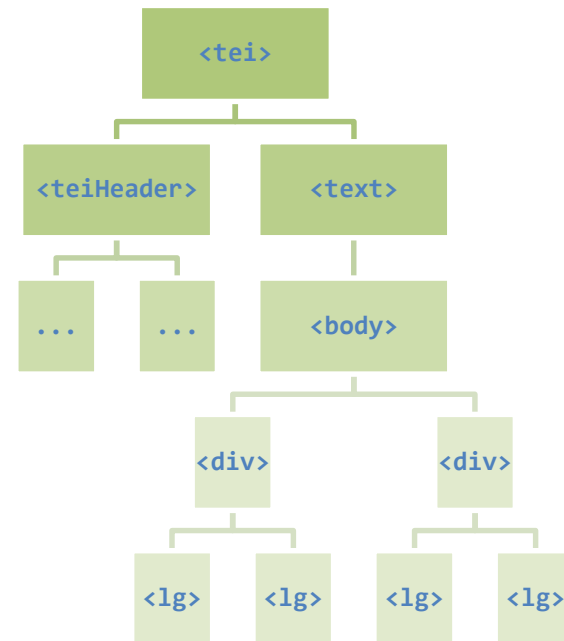
BERGISCHE
UNIVERSITÄT
WUPPERTAL

XML als Baum



XML als Baum

```
<TEI xmlns="http://www.tei-c.org/ns/1.0">
  <teiHeader>...</teiHeader>
  <text>
    <body>
      <div type="sonnet">
        <head>Sonnet 18</head>
        <lg type="quatrain">
          <l n="1">Shall I compare thee to a summer's day?</l>
          <l n="2">Thou art more lovely and more temperate:</l>
          ...
        </lg>
        ...
      </div>
      <div type="couplet">
        <l n="13">So long as men can breathe or eyes can see,</l>
        <l n="14">So long lives this and this gives life to thee.</l>
      </div>
    </body>
  </text>
</TEI>
```



Wichtige Begriffe, kennen Sie schon

- Elemente / Attribute / Knoten
- Eltern – Kinder
- Vorfahren – Nachfahren
- Geschwister
- Wurzelknoten, Dokumentknoten

Knotentypen

- Dokumentknoten
- Wurzelknoten
- Elementknoten
- Attributknoten
- Textknoten

- Kommentarknoten

Bewegung im Baum

Lokalisierungsschritte

„gehe von hier nach da“

Knotentests

„bist Du der, den ich suche?“

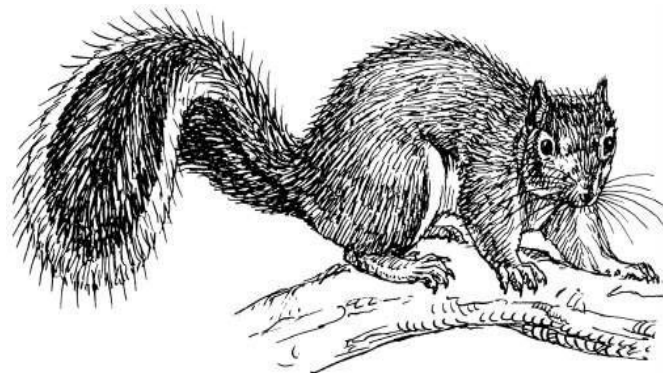
Lokalisierungspfade

[Schritt]/[Schritt]/[Schritt]
/TEI/text/body/div

„Kontext“

absolute Pfade (vom Dokument ausgehend)

vs. relative Pfade (vom aktuellen Kontext ausgehend)



Bewegung im Baum: Achsen

Vertikale Achsen

self::

child::

descendant::

descendant-or-self::

parent::

ancestor::

ancestor-or-self::

Horizontale Achsen

following::

following-or-self::

following-sibling::

preceding::

preceding-or-self::

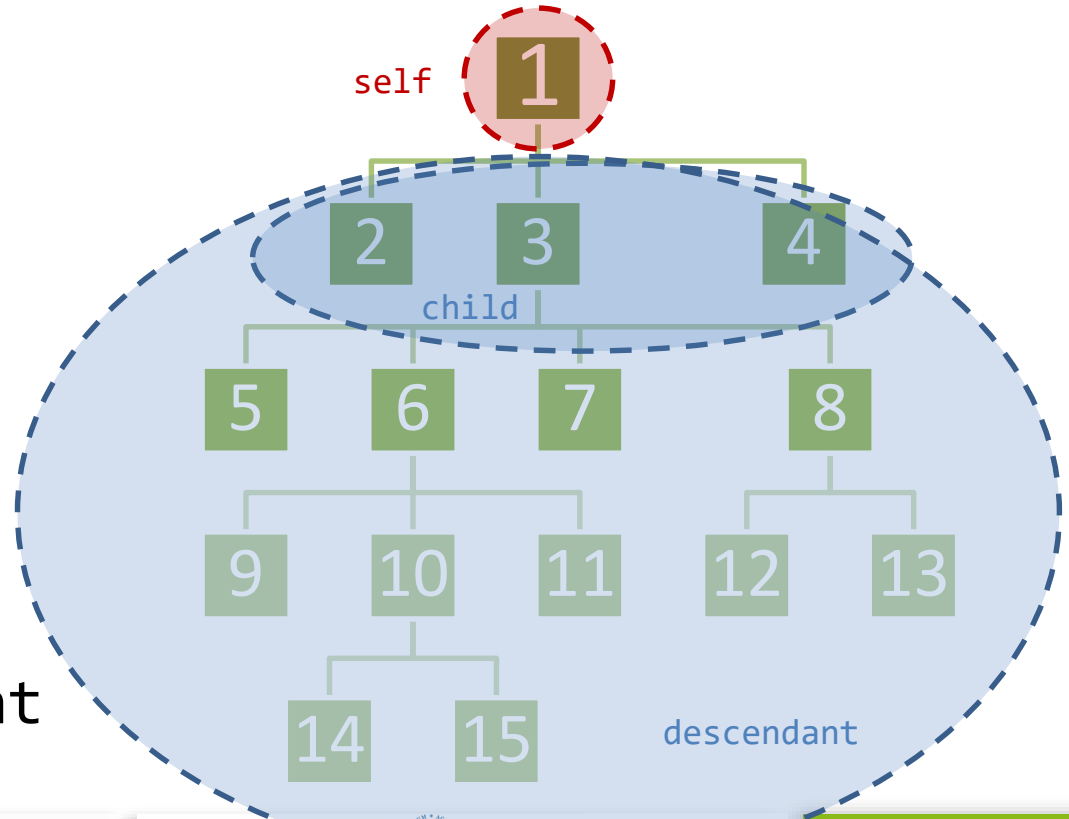
preceding-sibling::

Achsen

Selbst
self

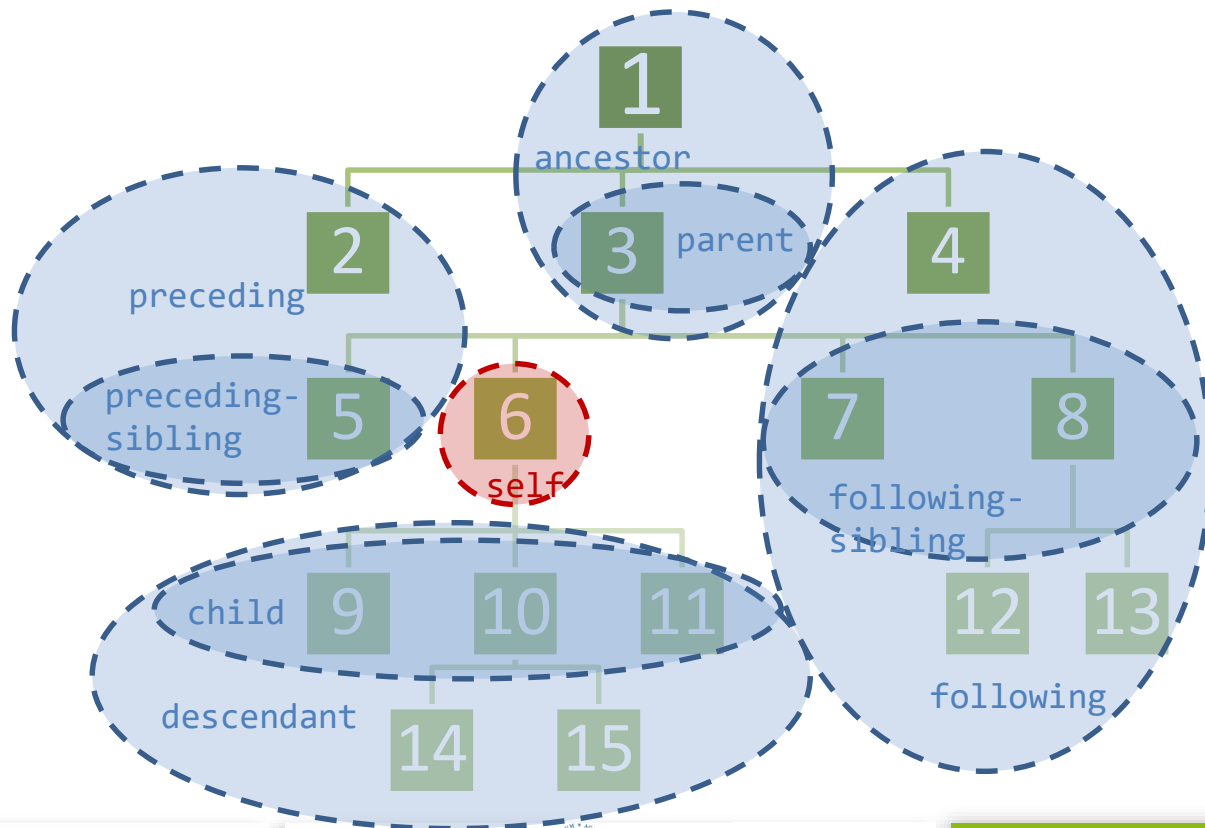
Eltern / Kind
parent / child

Vorfahren / Nachfahren
ancestor / descendant



- Selbst
self
- Eltern / Kind
parent / child
- Vorfahren /
Nachfahren
ancestor
descendant
- Geschwister
preceding-sibling
following-sibling
- Sequenz statt Baum
preceding
following

Achsen



Achsen: abgekürzte Syntax

Wichtig!

self::	.
child::	Elementname
parent::	..
descendant-or-self::	//Elementname
attribute::	@Attributname

Knotentest, beliebiger Elementname *

Beispiel: /TEI/text/body/div//person/@id

Bewegung im Baum

Lokalisierungsschritte

Prinzip: Achse + Knotentest + Prädikat

Syntax: achse::knotentest[Prädikat]

Beispiel

`//body/descendant::persName[@key='A000584']`

Der Grundbaukasten

- Verkettete Lokalisierungsschritte /.../.../
- Bedingungen, Prädikate [.....]
- Klammern, Schachtelung (...(...))
- Operatoren
 and or | = != < > + - * div
- Funktionen
 Funktionsname(Argument, Argument, ...)
- Syntax: 'Strings' in Anführungszeichen! Zahlen nicht!

Die Grund-Funktionsweise

- Der letzte bzw. äußerste Schritt bestimmt, was ein XPath-Ausdruck zurückgibt
- Pfade werden von vorne nach hinten abgearbeitet
- Klammern werden von innen nach außen aufgelöst

Beispiel

`//body/descendant::persName[@key='A000584']`

Rückgaben

XPath-Ausdrücke ergeben Rückgaben verschiedenen Typs:

- Knoten
- Knotenmengen
- Zahlen
- Strings
- Wahrheitswerte / Boolean
- Sequenzen

Funktionen

- Funktionen können mit ihrem Namen aufgerufen werden.
- Funktionen bestehen aus Ihrem Namen und runden Klammern:
funktion()
- Manche Funktionen erwarten in der Klammer die Übergabe von "etwas"
 - Das kann ein Knoten sein, ein Knotensatz, ein String, eine Zahl ...
- Die Übergaben sind durch Kommata getrennt
 - *funktion(parameter,parameter)*
- Funktionen geben dann etwas zurück
 - das kann ein Wahrheitswert sein, eine Zahl, ein String, eine Sequenz ...

Funktionen: Ein Beispiel

`contains(string,string)`

- prüft, ob ein Element oder String (das erste Argument) einen anderen String (das zweite Argument) enthält
 - liefert einen Wahrheitswert zurück
 - Auf Deutsch: Enthält der erste String den zweiten? Wahr oder falsch? True / False?
- `contains('Schnecke','ecke')`
 - Deutsch: Enthält der String Schnecke den String ecke?
 - Rückgabe: true
 - `//forename[contains(.,'Patrick')]`
 - Deutsch: Gibt es in meinem Baum ein Element forename, das den String Patrick enthält?
 - Rückgabe: Alle Elemente forename, für die das Prädikat "true" zurückgibt (Knotenset)

Funktionen

Zum Nachschlagen, zum Lernen, zur Inspiration

- Einfache Liste (bis XPath 2.0):
https://www.w3schools.com/xml/xsl_functions.asp
- Lange Liste (mit XPath 3.1):
<https://www.altova.com/xpath-xquery-reference>
- Die autoritative Referenz: <https://www.w3.org/TR/xpath-functions-31/>

Einige Funktionen I

count(nodeset)

- zählt etwas, erwartet eine Sequenz oder ein Knotenset
- liefert eine Zahl zurück

position()

- gibt die Position eines Knotens an, liefert eine Zahl zurück
- Abgekürzte Syntax: `//person[position()=11] == //person[11]`

string-length(string)

- zählt die Länge eines Strings (in Zeichen)
- liefert eine Zahl zurück

Einige Funktionen II

starts-with(string, string)

- prüft, ob ein String mit einem anderen String beginnt, liefert einen Wahrheitswert zurück
- Starts-with('Patrick','P') → true

not(boolean)

- dreht einen Wahrheitswert um, liefert einen Wahrheitswert
- not(1 > 2) → true

max(sequence_{of numbers}) ähnlich: min(), sum(), avg()

- ermittelt den maximalen Wert aus einer Reihe von Werte, liefert eine Zahl zurück
- max(//preis) → *eine Zahl*

distinct-values(sequence_{of strings})

- gibt eine Sequenz von (unterschiedlichen) Werten zurück
- distinct-values(//vorname) → eine Sequenz unterschiedlicher Strings

Einige Funktionen III

- `substring(string,start,length)`
- `matches(string, pattern)`
→ reguläre Ausdrücke!
- `tokenize(string, pattern)`
- `name()`
- `not(argument)`
- `number(string), string(number)`
- `doc(URI)`

Einige Funktionen IV

Nur der Vollständigkeit halber ...

- `concat(string, string, ...)`
- `translate(string1,string2,string3)`
Converts string1 by replacing the characters in string2 with the characters in string3
- `sum(arg,arg,...)`
- `last()`
- `current-date()`
- `(`

Übungen

1. Legen Sie sich den Foliensatz bereit (zum nachschlagen)
2. persons.xml in oXygen öffnen (<https://tinyurl.com/wsde20>)
3. Ein Gefühl für die Daten entwickeln
4. Den XPath-Evaluator benutzen
5. Üben heißt Übersetzen (Deutsch-XPath / XPath-Deutsch)
Man kann sich schrittweise an das Ergebnis herantasten!

Wie lautet die Überschrift des Dokuments?

`/TEI/text/body/div/head`

Gib mir alle Personen zurück

`/TEI/text/body/div/listPerson/person`

oder `//person (?)` oder `//listPerson/person`

Übungen

Welche ist die dritte Person in der Liste?

`//listPerson/person[position()=3]` *oder* `//listPerson/person[3]`

Gib mir alle Frauen zurück

`//listPerson/person[sex/@value='F']`

Gib mir alle Vornamen aller Frauen zurück

`//listPerson/person[sex/@value='F']/persName/forename`

Gib mir alle Frauen, die mit C anfangen

`//listPerson/person[sex/@value='F']/persName/forename[starts-with(.,'C')]`

Gib mir die unterschiedlichen Vornamen aller Frauen zurück

`distinct-values(//listPerson/person[sex/@value='F']/persName/forename)`

Übungen

Zu welchen Personen liegt kein GND-Link vor?

```
//listPerson/person[not (contains(persName/@ref,'gnd'))]
```

Wieviele sind das?

```
count(//listPerson/person[not (contains(persName/@ref,'gnd'))])
```

Wieviel Prozent der Liste haben keine GND-Information?

```
count(//listPerson/person[not (contains(persName/@ref,'gnd'))])  
div count(//listPerson/person) * 100
```

Das früheste Geburtsjahr in der Liste?

```
min(//listPerson/person/birth/date/@when-iso)
```

Wie alt sind die Leute im Durchschnitt geworden?

Pseudocode! Sterbejahr minus Geburtsjahr. Das einer Funktion übergeben ...

```
avg(//listPerson/person/((death/date/@when-iso) - (birth/date/@when-iso)))
```

Fun: „Hallo Welt“ in XPath ?

XML als Datenstruktur: XPath

Patrick Sahle



Leopoldina
Nationale Akademie
der Wissenschaften



berlin-brandenburgische
AKADEMIE DER WISSENSCHAFTEN



BERGISCHE
UNIVERSITÄT
WUPPERTAL

XPath ist ein großer Spaß !