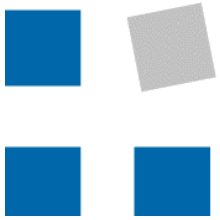


HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

XML

Teil 6: XLink, XPointer, XPath

Abteilung Informatik – WS 02/03

XLink und XPointer

- Verknüpfungen in XML:
  - XLink (XML Linking Language) definiert, wie ein Dokument zu anderen Dokumenten verknüpft wird
  - XPointer (XML Pointer Language) definiert, wie einzelne Teile eines Dokuments adressiert werden

Einfacher XLink (1/2)

- Ein einfacher XLink definiert eine „one-way“ Verbindung zwischen zwei Ressourcen:

  Start-Ressource → End-Ressource

- Start-Ressource (engl. *starting ressource*): XML-Element
- End-Ressource (engl. *ending ressource*):
  - XML-Dokument
  - Element oder Gruppe von Elementen in einem XML-Dokument
  - Nicht-XML-Entität (z.B. Bild)

Beispiel

```
<?xml version="1.0" encoding="UTF-8"?>
<FACHHOCHSCHULEN xmlns:xlink="http://www.w3.org/1999/xlink">
   <HOCHSCHULE xlink:type="simple"
      xlink:href="http://www.hsr.ch">Rapperswil</HOCHSCHULE>
   <INFO xlink:type="simple"
      xlink:href="studiengaenge.xml">Studiengänge</INFO>
   <LOGO xlink:type="simple" xlink:href="logo.gif"/>
</FACHHOCHSCHULEN>
```

Einfacher XLink (2/2)

- XLink-Namensraum: `http://www.w3.org/1999/xlink`, meistens mit dem Präfix `xlink`
- XLink-Attribute:
  - `xlink:type` mit Wert `simple`. Andere möglichen Werte für einen nicht-einfachen XLink: `extented`, `locator`, `arc`, `resource`, `title`, `none`
  - `xlink:href` : URI der End-Ressource, z.B. absolute oder relative URL

Link-Verhalten

- Ein XLink zeigt eine Verbindung zwischen Ressourcen
- Die Applikation (z.B. der Browser) entscheidet, was diese Verbindung bedeutet
- Das XML-Dokument kann der Applikation (z.B. dem Browser) eine Bedeutung der Verbindung vorschlagen
  - Attribut `xlink:show` schlägt vor, wie die Ressource angezeigt werden soll. Werte: `new`, `replace`, `embed`, `other`, `none`
  - Attribut `xlink:actuate` schlägt vor, ob und wie die Verbindung befolgt werden soll. Werte: `onLoad`, `onRequest`, `other`, `none`

Linkverhalten: Beispiel

```
<?xml version="1.0" encoding="UTF-8"?>
<FACHHOCHSCHULEN xmlns:xlink="http://www.w3.org/1999/xlink">
   <HOCHSCHULE xlink:type="simple"
      xlink:href="http://www.hsr.ch"
      xlink:actuate="onRequest" xlink:show="new">
   Rapperswil
   </HOCHSCHULE>
</FACHHOCHSCHULEN>
```

Link-Semantik

- Zwei Attribute für die Semantik:
  - `xlink:title` mit einem String-Wert
  - `xlink:role="URI"`
- Die Applikation entscheidet, wie der Wert dieser Attribute verwendet wird

Link-Semantik: Beispiel

```
<?xml version="1.0" encoding="UTF-8"?>
<FACHHOCHSCHULEN
   xmlns:xlink="http://www.w3.org/1999/xlink">
   <HOCHSCHULE xlink:type="simple"
      xlink:href="http://www.hsr.ch"
      xlink:actuate="onRequest" xlink:show="new"
      xlink:title="Homepage der HSR">
   Rapperswil
   </HOCHSCHULE>
</FACHHOCHSCHULEN>
```

Erweiterter Link: Beispiel

```
<?xml version="1.0" encoding="UTF-8"?>
<FACHHOCHSCHULE xmlns:xlink="http://www.w3.org/1999/xlink"
        xlink:type="extended">
   <HOCHSCHULE xlink:type="locator"
               xlink:href="http://www.hsr.ch"
               xlink:label="HSR" >Rapperswil
   </HOCHSCHULE>
   <HOCHSCHULE xlink:type="locator"
               xlink:href="http://www.hsr.ch/studiengaenge.xml"
               xlink:label="Studien" >
   </HOCHSCHULE>
   <HOCHSCHULE xlink:type="locator"
               xlink:href=" http://www.hsr.ch/adresse.xml"
               xlink:label="Adresse" >
   </HOCHSCHULE>
   <MORE xlink:type="arc" xlink:from="HSR" xlink:to=" Studien "/>
   <MORE xlink:type="arc" xlink:from="HSR" xlink:to="Adresse"/>
   <BACK xlink:type="arc" xlink:from="Studien" xlink:to="HSR"/>
</FACHHOCHSCHULE>
```

Erweiterter Link

- Erweiterter Link = Gerichteter Graph mit Labels
  - Knoten ↔ Ressourcen
  - Pfad ↔ Verbindung
  - Label ↔ URI
- xlink-Typen:
  - xlink:type = "extended"
  - xlink:type = "arc"
  - xlink:type = "ressource"
  - xlink:type = "locator "

DTD un XLink

- XLink-Attribute müssen deklariert werden, wie alle anderen Attribute

DTD und XLink: Beispiel (1/2)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE WEBSITE SYSTEM "ex84.dtd">
<WEBSITE xmlns:xlink="http://www.w3.org/1999/xlink"
       xlink:type="extended" xlink:title="HSR Homepage">
   <NAME xlink:type="resource">Welcome to the HSR</NAME>
   <HOMESITE xlink:type="locator"
       xlink:href="http://www.hsr.ch"/>HSR-Site</>
   <MIRROR xlink:type="locator" xlink:title="HSR Homepage
       (ITA Mirror)"
       xlink:href="http://www.ita.hsr.ch/hsr/"/>Mirror</>
</WEBSITE>
```

DTD und XLink: Beispiel (2/2)

```
<!ELEMENT WEBSITE (NAME, HOMESITE, MIRROR*) >
<!ATTLIST WEBSITE
  xmlns:xlink  CDATA      #FIXED
   "http://www.w3.org/1999/xlink"
  xlink:type  (extended) #FIXED  "extended"
  xlink:title  CDATA      #IMPLIED>
<!ELEMENT NAME (#PCDATA)>
<!ATTLIST NAME
   xlink:type  (resource) #FIXED    "resource">
<!ELEMENT HOMESITE (#PCDATA)>
<!ATTLIST HOMESITE
   xlink:type  (locator)  #FIXED    "locator"
   xlink:href   CDATA      #REQUIRED
  xlink:title  CDATA      #IMPLIED>
<!ELEMENT MIRROR (#PCDATA)>
<!ATTLIST MIRROR
   xlink:type  (locator)  #FIXED  "locator"
   xlink:href   CDATA      #REQUIRED
   xlink:title  CDATA      #IMPLIED>
```

XPointer: Motivation

- Deklaration einer URL mit Anker in einem HTML-Dokument:

```
<A HREF="http://www.ita.hsr.ch/ebiz.htm#Inhalt>
```

- Ziel-Dokument muss modifiziert werden, damit sich das Quelldokument mit ihm verbinden kann

`xpointer()`-Ausdruck

- XPointer können mit einem `xpointer()`-Ausdruck definiert werden, welche am Ende einer URI angehängt werden kann
- Syntax: `xpointer(xpath)`
- Beispiele:
  - `xpointer(id("PERSON"))`
  - `xpointer(/child::PERSON/child::PROFESSION[position()=2]`

XPointer mit XLink

- Beispiel einer URL mit XPointer:

```
http://www.hsr.ch/modulliste.xml#xpointer(//informatik)
```

- Beispiel eines XPointers mit XLink:

```
<first_image xlink:type="simple"
    xlink:href="#xpointer(//slide[position()=1]">
```

- Bemerkungen:
  - Der Kontextknoten ist der Root-Knoten der Entität, welche der XPointer enthält, wenn der Xpath-Ausdruck relativ ist
  - Alternative `xpointer()`–Ausdrücke werden der Reihe nach berücksichtigt, z.B.

```
xpointer(//first_name)xpointer(//last_name)
```

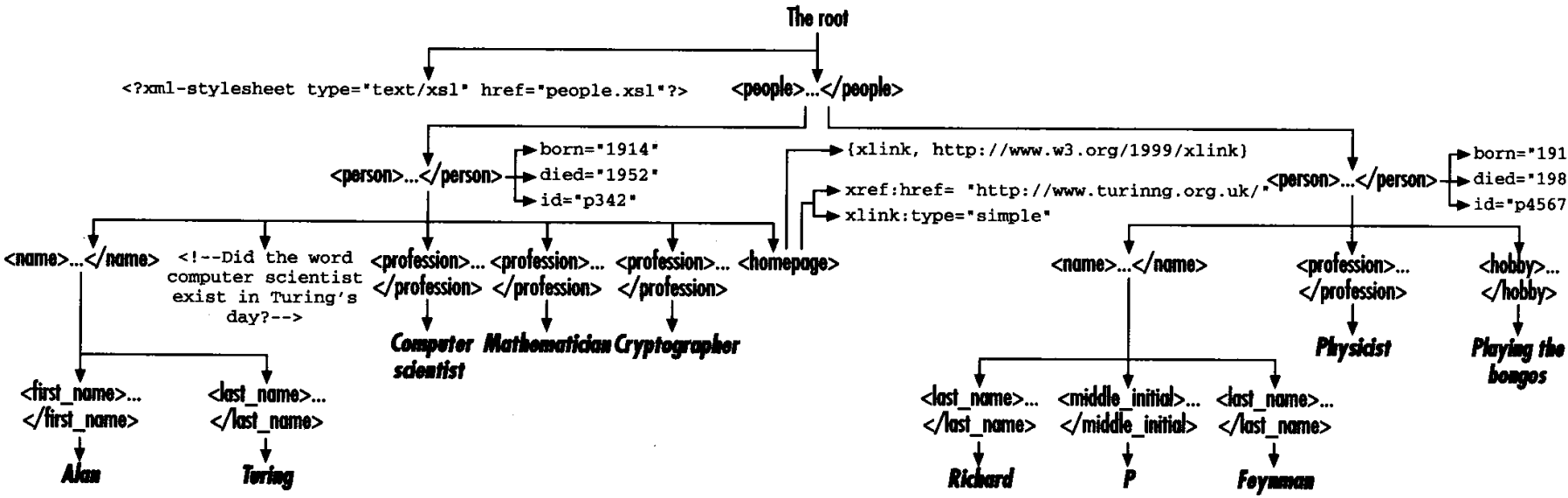
X-Path

- X-path = Nicht-XML-Sprache, um Teile eines Dokuments zu identifizieren
- X-Path Datenmodell: ein XML-Dokument hat eine Baumstruktur
- Knoten:
  - Wurzel (engl. root)
  - Elemente
  - Texte
  - Attribute
  - Namensräume
  - Anweisungen (engl. processing instructions)
  - Kommentare

X-Path Datenmodell: Beispiel (1/2)

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="people.xsl"?>
<!DOCTYPE people [
    <!ATTLIST homepage
    xlink:type CDATA #FIXED "simple"
          xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
>
    <!ATTLIST person
    id ID #IMPLIED
>
]>
<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>
```

X-Path Datenmodell: Beispiel (2/2)



Quelle: E. Harold & al. XML in a Nutshell, s. 156

X-Path-Datenmodell

- Root-Knoten ≠ Root-Element!
- Folgende Dokument-Teile sind nicht Teil des X-path-Datenmodell (d.h. sind nicht im XML-Dokument-Baum enthalten):
  - XML-Deklaration
  - DOCTYPE-Deklaration
  - DTD
  - `xmlns`- und `xmlns:`*`prefix`*-Attribute
  - Entitätsreferenzen (z.B.: `&`), Charakterreferenzen (z.B.: `&#x3C3;` für α), CDATA-Abschnitte

Location Path (Bezeichnungen)

- Ein **Lokalisierungspfad** (engl.: *location path*) identifiziert einen Knoten oder eine Menge von Knoten in einem Dokument
- Ein Lokalisierungspfad besteht aus **Lokalisierungsschritten** (engl. *location steps*)
- Jeder Lokalisierungsschritt bezieht sich auf einen **Kontextknoten** (engl. *context node*)

Root Location Path

- Der Root-Lokalisierungspfad wählt den Root-Knoten aus
- Bezeichnung: /

Lokalisierungsschritt

- Syntax: Ein Lokalisierungsschritt besteht aus einer **Achse**, einem **Knotentest** und einem (optionalen) **Predikat**:

```
axis::node-test[predicate]
```

- Beispiel: Der Lokalisierungsschritt

```
child::PERSON[position()=2]
```

wählt das zweite PERSON-Element, das ein Child-Element des Kontextknotens ist

Achsen (1/2)

- Die Achse gibt an, in welcher „Richtung“ man ab dem Kontextknoten suchen muss
- Liste der Xpath-Achsen:
  - `child`: Alle Knoten, die im Kontextknoten enthalten sind, aber nicht in einem anderen Knoten aus dem Kontextknoten
  - `parent`: Der einmalige Knoten, der den Kontextknoten, aber keinen anderen Knoten enthält
  - `self`: Der Kontextknoten
  - `ancestor`: Die Vorfahren des Kontextknotens und den Kontextknoten selbst
  - `attribute`: Die Attribute des Kontextknotens
  - `descendant`: Die Ableitungen (Nachfolger) des Kontextknotens, die Ableitungen von den Ableitungen des Kontextknotens, etc.

Achsen (2/2)

Liste der Xpath-Achsen (Fortsetzung):

- `descendant-or-self`: Der Kontextknoten selbst und seine Ableitungen
- `following`: Alle Knoten, die nach dem Ende des Kontextknotens beginnen, ausser Attributs- und Namensraumknoten
- `following-sibling`: Alle Knoten, die nach dem Ende des Kontextknotens beginnen und den gleichen Stamm wie der Kontextknoten haben (Beachte: Attributs- und Namensraumknoten haben keine Siblings)
- `namespace`: Alle Namensräume, die für den Kontextknoten definiert wurden
- `preceding`: Alle Knoten, die enden, bevor der Kontextknoten anfängt, ausser den Attributs- und Namensraumknoten
- `preceding-sibling`: Alle Knoten, die vor Begin des Kontextknotens anfangen und den gleichen Stamm wie der Kontextknoten haben

Knotenteste (1/2)

- Meistens: Element- oder Attributsname, z.B.

  `/descendant::last_name, /descendant::@died`

- Weitere Knotenteste:
  - `*` steht für ein belibieges Element
  - `prefix:*` wählt alle Elemente im Namensraum aus, die mit dem Präfix `prefix` versehen sind
  - `@prefix:*` wählt alle Attribute im `prefix`-Namensraum aus
  - `text()` wählt den Inhalt eines Elements, der aus geparstem Text besteht, z.B.

    `/descendant::text()[position()=2]`

    verweist auf Blablabla... im XML-Dokument:

```
<brief>Zürich<datum>12.11.02</datum>Blablabla...<brief>
```

Knotenteste (2/2)

- Weitere Knotenteste (Fortsetzung)
  - `comment()` wählt ein Kommentar aus

    `/descendant::comment()[position()=3]`
  - `processing-instruction()` wählt alle Anweisungen, die entlang der Achse stehen. Mit oder ohne Argumente:
    - `processing-instruction()` : alle Anweisungen werden ausgewählt
    - `processing-instruction('target')`: nur die target-Answeisung wird ausgewählt, z.B.

    `/descendant::processing-instruction('style-sheet')`

Prädikate

- Jeder Lokalisierungsschritt kann ein oder mehrere Prädikate enthalten, die weiter den Lokalisierungspfad einschränken
- Beispiele:
  - `. = "physicist"` (der aktuelle Knoten hat den Wert `physicist`)
  - `name = "Turing"` (das `name`-Element hat den Wert `Turing`)
  - `@id = "p33-222"` (das `id`-Attribut hat den Wert `p33-222`)

Location Path

- Lokalisierungspfad: Folge von Lokalisierungs-schritten, welche durch das Zeichen `/` getrennt werden
- Beispiel: Der Lokalisierungspfad

```
/child::FAMILYTREE/child::PERSON
```

besteht aus zwei Lokalisierungsschritten:

- `/child::FAMILYTREE`: Absoluter Schritt, der alle Child-Elemente des Root-Knotens mit Namen `FAMILYTREE` sucht
- `child::PERSON`: Schritt der alle Child-Elemente des Kontextknotens (hier: `FAMILYTREE`-Element-Knoten) mit Namen Person auswählt

Abgekürzte Lokalisierungsschritte

`child::name`	`name`
`parent::node()`	`..`
`self::node()`	`.`
`descendant-or-self::node()`	`//`
`attribute::name`	`@name`

Beispiel:

Nicht abgekürzt:

```
/child::list/child::person[position()=3]/attribute::born
```

Abgekürzt:

```
/list/person[position()=3]/@born
```

Wildcards

- Ersatzzeichen (engl. Wildcard):
  - `*`: jeder Element-Knoten
  - `@*`: jeder Attribut-Knoten
  - `node()`: jeder Knoten
- Beispiel:

```
/list/person[position()=3]/@*
```

XPath-Funktionen (1/2)

- Es gibt 27 vordefinierte Funktionen, die in XPath-Ausdrücke verwendet werden sollen:
- Die Ein- und Ausgabe dieser Funktionen haben einen der folgenden XPath-Datentypen:
  - `boolean` (Werte: `true`, `false`)
  - `number` (entspricht dem `double`-Datentyp in JAVA)
  - `string` (Folge aus Unicode-Zeichen)
  - `node-set`: Menge aus n≥0 Knoten eines XML-Dokuments

XPath-Funktionen (2/2)

- Beispiel von XPath-Funktionen:
  - `node-set id(string IDs)`
  - `number position()`
  - `string substring(string s, number n, number length)`
- Liste aller Xpath-Funktionen: siehe z.B. E. Harold, W. Means, *XML in a Nutshell*, S. 423-429.

Referenzen

- E. Harold, W. Means: *XML in a Nutshell*
  - XLink: Kapitel 10
  - XPointer: Kapitel 11, s. 190 - 193
  - XPath: Kapitel 9