Anfragen mit XPath

XPath

- Die Notation für die Adressierung der Knoten einer XML-Baumstruktur ist die XML-Anwendung XPath.
- Zusätzlich bietet XPath logische Ausdrücke und zusätzliche Funktionen wie Bedingungen und Funktionen, die etwas über die Position aussagen.
- Die einfachste Form der Adressierung orientiert sich an der Verzeichnisnotation.
- XPointer basiert auf XPath und bietet Funktionen zum Zugriff auf einzelne Elemente und Bereiche innerhalb des XML-Dokuments.

Knoten in XPath

- XPath kennt sieben verschieden Knoten:
  - Wurzelknoten: Wurzel des XPath-Baums.
  - Elementknoten: Jedes Element ist ein Knoten.
  - Textknoten: Zeichenketten der Elemente.
  - Attributknoten: Attribute der Element.
  - Namensraumknoten: Der Namensraumteil eines Element- oder Attributknotens
  - Verarbeitungsanweisungsknoten: Knoten einer Processing-Instruction.
  - Kommentarknoten

Wurzelknoten

- Der Wurzelknoten ist dem obersten Element übergeordnet. Er entspricht nicht dem Wurzelelement der XML Datei.
- Das Wurzelelement der XML Datei ist der einzige Kind-Knoten des Wurzelknotens.
- Der Wurzelknoten hat kein Namensraum-Bezeichner.

Elementknoten und Textknoten

- Jedes Element eines XML Dokuments wird durch einen Elementknoten repräsentiert.
- Kinder eines Elementknotens können weitere Elemente aber auch Kommentare, Verarbeitungsanweisungen und Text sein.
- Textknoten sind Kinder von Elementknoten.
- Textknoten enthalten den Text, der durch einen Elementknoten eingeschlossen wird.

Attributknoten

- Attributknoten sind die Attribute der Elemente einer XML Datei.
- Sie sind nicht Kinder der Elementknoten.
- Der Zugriff erfolgt über eine spezielle Syntax.
- Der Inhalt der Attributknoten ist der Text, der durch das Attribut festgelegt ist.

Sonstige Knoten

- Namensraumknoten bilden den Namensraum eines Knotens ab.
  - Sie sind nicht Kinder eines Elementknotens!
- Verarbeitungsanweisungsknoten stellen die Verarbeitungsanweisungen (processing instructions) dar.
- Kommentarknoten enthalten den Text des Kommentars.

XPath-Ausdrücke

- Knoten oder Knoten-Mengen des XML-Baums können über folgende generelle Notation erfragt werden:
  - Location Path:

    [/]schritt(/schritt)*

  - wobei jeder Schritt (location step):

    Achse :: typ|name ([prädikat])*

    (statt dem Knotentest „typ|name“ kann auch * als Wildcard benutzt werden)

- Diese Ausdrücke heißen XPath-Ausdrücke.

Übersicht der XPath Achsen

- **child**: Kindelemte.
- **parent**: Direkter Vorfahr.
- **descendant**: Alle Nachfahren.
- **ancestor**: Alle Vorfahren.
- **following**: Alle Nachfolger bzgl. Dokumentenordnung ohne Nachfahren.
- **preceding**: Alle Vorgänger bzgl. Dokumentenordner ohne Vorfahren.
- **following-sibling**: Nachfolgende Geschwister.
- **preceding-sibling**: Vorhergehende Geschwister.

Übersicht der XPath Achsen

- **attribute**: Attribute des Knotens.
- **namespace**: Namensraumkürzel.
- **self**: Der aktuelle Knoten.
- **descendant-or-self**: descendant und der aktuelle Knoten.
- **ancestor-or-self**: ancestor und der aktuelle Knoten.

- Beispiel: achse.xsl

Einschränkung der Knotenmenge

- Die Knoten können über folgende Typprüfungen eingeschränkt werden (Knotentest):
  - * Alle Knoten bzgl. des Achsentyp
  - text() Alle Textknoten
  - comment() Alle Kommentarknoten
  - node() Alle Knoten
  - processing-instruction(name) Alle Verarbeitungsanweisungsknoten mit dem optionalen Namen „name“

Beispiele für XPath-Ausdrücke

`child::text()`

- Dieser Location-Path wählt alle Textelemente aus, die direkte Nachkommen des Kontextknotens sind.

`/child::html`

- Wählt das html-Element aus, das ein Kind des Wurzelknotens sein muß.

`/descendant-or-self::node()/child::table`

- Wählt alle table-Tags im Dokument aus.

`/parent::node()/child::table/attribute::width`

- Wählt die WIDTH-Attribute aller Tabellen aus, die Kinder des übergeordneten Knotens sind.

`following-sibling::*/child::tr/child::td`

- Wählt alle TD-Elemente aus, die innerhalb eines TR-Elements stehen, das in einem Bruderelement des aktuellen Elements steht.

XPath-Ausdrücke (Kurzform)

- Einige oft benutzte Teilausdrücke lassen sich auch in Kurzform schreiben:
  - statt child:: (weglassen)
  - @ statt attribute::
  - . statt self::node()
  - .. statt parent::node()
  - // statt /descendant-or-self::node()/
  - [number] statt [position()=number]

Beispiele für kurze XPath-Ausdrücke

`text()`

- Dieser Location-Path wählt alle Textelemente aus, die direkte Nachkommen des Kontextknotens sind.

`/html`

- Wählt das html-Element aus, das ein Kind des Wurzelknotens sein muß.

`//table`

- Wählt alle table-Tags im Dokument aus.

`../table/@width`

- Wählt die WIDTH-Attribute aller Tabellen aus, die Kinder des übergeordneten Knotens sind.

`following-sibling::*/tr/td`

- Wählt alle TD-Elemente aus, die innerhalb eines TR-Elements stehen, das in einem Bruderelement des aktuellen Elements steht.

Adressierung mit Positionsangabe und Bedingung

- Elemente und Attribute können auch durch die absolute Position oder in Abhängigkeit von einer Bedingung ausgewählt werden.
- Beispiel 1: Zweite Adresse

```
/adressbuch/adresse[2]
```

- Beispiel 2: Adressen mit der Postleitzahl 12345

```
/adressbuch/adresse[./ort/@plz="12345"]
```

- Beispiel 3: Auswahl eines Elements anhand des Zeichenkettenwerts

```
<xsl:apply-templates
select="einkaufspreis[text()='10,00']">
```

Weitere Vergleichsoperatoren

- XPATH bietet folgende Vergleichsoperatoren:
  - **=**: Gleich
  - **!=**: Ungleich
  - **>**: Größer als
  - **>=**: Größer oder gleich
  - **<**: Kleiner als
  - **<=**: Kleiner oder gleich
  - **and**: Und Verknüpfung
  - **or**: Oder Verknüpfung

Numerischer Vergleich

```
<xsl:template match="einkaufspreis[number()=10]">
   <p><xsl:value-of select="."/></p>
</xsl:template>
```

- In diesem Fall wird durch die Funktion **number()** der Textknoten des aktuellen Knotens in einen Zahl umgewandelt.
- Textknoten sind, analog zu den Attributknoten, eigene Knoten im XPath Baum.
- Der Vergleichswert muss natürlich ein numerischer Wert sein.
- Trennzeichen für Nachkommastellen ist der Punkt.

Operationen und Funktionen

- XPATH bietet Operationen für die Verknüpfung von Zahlen und Zeichenketten.
- Neben den Vergleichsoperatoren kann mit Zahlen gerechnet werden und Zeichenketten verändert und zusammengesetzt werden.
- Einige Funktionen arbeiten auch mit Knotenmengen.
- Bei diesen Funktionen wird der aktuelle Knoten verarbeitet oder die Menge als Ganzes betrachtet.

Operationen für Zahlen (rechnen.xsl)

- Vergleichsoperatoren wurden bereits im Zusammenhang mit den Funktionen zur Auswahl von Elementen vorgestellt.
- Zur Berechnung gibt es folgende Operationen:
  - **+**, **-**, *: Plus, Minus, Multiplikation
  - **div**: Division
  - **mod**: ganzzahliger Rest
  - **floor(**number**)**: größte ganze Zahl, die nicht größer als number ist.
  - **ceiling(**number**)**: kleinste ganze Zahl, die nicht kleiner als number ist.
  - **round(**number**)**: gerundete Zahl

Funktionen zur Orientierung

- XSL bietet eine Reihe von Funktionen mit deren Hilfe die Position im XPath Baum bestimmt werden kann.
- Mit diesen Funktionen kann z. B. eine automatische Numerierung der Elemente realisiert werden.
- Ebenso können bestimmte Elemente an bestimmten Positionen ausgewählt werden.
- Es kann durch eine Zahl auch eine absolute Position eines Elements angegeben werden.

Liste der Kernfunktionen

- **last()**: Letzter Knoten einer Knotenliste
- **position()**: Aktuelle Position eines Knotens
- **count(**node-set**)**: Anzahl der Knoten in node-set
- **local-name(**node-set**?)**: Lokaler Name
- **namespache-uri(**node-set**?)**: Namensraumkürzel
- **name(**node-set**?)**: Name mit Namensraumkürzel
- node-set ist eine Knotenmenge.
- Parameter mit **?** müssen nicht unbedingt angeben werden. Wenn keine Knotenmenge angegeben ist wird der aktuelle Knoten verwendet.

Numerierte Liste erzeugen

- Mit der Funktion **position()** kann die aktuelle Position in der Knotenmenge ermittelt werden.
- Die Funktion **number()** bietet Möglichkeiten Zahlen zu formatieren.
- Mit diesen beiden Funktionen kann eine numerierte Ausgabe erzeugt werden.
- Wichtige Attribut von **number()** sind:
  - **value**: Zahlenwert
  - **format**: Formatierung der Ausgabe
- Beispiel: numerieren.xsl

Funktionen für Zeichenketten

- Mit Funktionen für Zeichenketten lassen sich Zeichenketten verändern und deren Inhalte überprüfen.
- Die üblichen Zeichenkettenfunktionen sind:
  - **string(**objekt**?)**: Wandelt objekt in eine Zeichenkette um. Falls kein objekt übergeben wird, wird er aktuelle Knoten oder Unterbaum als objekt genommen.
  - **string-length(**string**?)**: Ermittelt die Länge der aktuellen Zeichenkette. Falls keine Zeichenkette übergeben wird, wird die aktuelle Zeichenkette genommen

Funktionen für Zeichenketten

- Weitere Zeichenkettenfunktionen sind:
  - **concat(**string, string, string***)**: Verbindet die übergebenen Zeichenketten zu einer Zeichenkette. Es müssen mindestens zwei Zeichenketten übergeben werden.
  - **contains(**string, string**)**: Diese Funktion liefert wahr, falls die zweite Zeichenkette in der ersten enthalten ist.
  - **substring(**string, zahl, zahl**?)**: Liefert den Ausschnitt der ersten Zeichenkette von Stelle zahl1 bis zahl2 oder bis zum Ende der Zeichenkette.
  - **starts-with(**string, string**)**: Diese Funktion liefert wahr falls die erste Zeichenkette mit der anderen Zeichenkette beginnt.

Zusammenfassung

- XPath bietet durch die Verzeichnisnotation eine einfache Form, um auf Knoten zuzugreifen.
- Durch die längere Form in XPath ist ein spezielleres Auswählen von Knoten und Knotenmengen möglich.
- XPath-Operationen bieten breitgefächerte Möglichkeiten zum Zugriff auf die Knoten des Dokumentenbaums.
- Durch XPath-Operationen sind Bedingungen und Berechnungen von Werten möglich.
- XPath-Funktionen ermöglichen Zeichenkettenoperationen, wie Teilstring und Zahlenformatierung, sowie das Testen des Knotentyps durch Funktionen wie node() oder comment().