OTS 3
Fachvortrag

3.0-Updates von XSLT und XPath auf einen Blick

Dr. Thomas Meinike, Hochschule Merseburg

Motivation

Die 1999 vom W3C etablierte Transformationssprache XSLT gehört zu den ältesten und produktivsten Technologien im „XML-Universum". Mit der 2007 erschienenen Version 2.0 wurden die Möglichkeiten wesentlich erweitert [1]. Fünf Jahre später stehen die 3.0-Spezifikationen von XSLT und der zum Zugriff auf XML-Strukturen und -Inhalte verwendeten Abfragesprache XPath sowie vom zusätzlich entwickelten XQuery vor der finalen Veröffentlichung. Die zum Entstehungszeitpunkt dieses Beitrages vorliegenden Arbeitsentwürfe [2–4] wurden zuletzt im Juli 2012 bzw. Dezember 2011 aktualisiert.

Schwerpunkte

An dieser Stelle wird nur ein formaler Überblick zu den wesentlichen Erweiterungen gegeben, da es sich insgesamt um eher kleinteilige und in der praktischen Anwendung codelastige Details handelt:

- Streaming soll vor allem die Arbeit mit sehr großen Ausgangsdokumenten verbessern. XML-Daten in der Größenordnung von mehreren hundert Megabytes oder gar Gigabytes bringen jeden noch so performanten Rechner schnell an seine Grenzen. Die Streaming-Technik erlaubt den sequenziellen Zugriff auf Teile des jeweiligen Dokuments. Dazu dienen die neuen XSLT-Elemente xsl:stream und xsl:iterate in Kombination mit xsl:next-iteration, xsl:break und xsl:on-completion sowie xsl:fork und xsl:merge.
- Funktionen höherer Ordnung (Higher-Order Functions, HOF) ermöglichen die Nutzung von Funktionen als Parameter und / oder Rückgabewerte innerhalb eigener Funktionen. Zum Aufbau komplexerer Logik dienen die neuen XPath-Funktionen fn:filter, fn:fold-left, fn:fold-right, fn:map und fn:map-pairs.
- Informationen über (existierende) Funktionen und ihre Argumente lassen sich über fn:function-lookup, fn:function-name und fn:function-arity auswerten.
- Weitere neue Funktionen vereinfachen den selektiven Zugriff auf Knoten und Sequenzen: fn:head, fn:tail sowie fn:innermost, fn:outermost und fn:has-children. Diese lassen sich zwar auch durch andere Abfragen abbilden, sind jedoch bei passender Gelegenheit durchaus nützlich.
- Inline-Funktionen können direkt in Variablen vorgehalten und z. B. innerhalb von xsl:value-of wie vordefinierte oder mittels xsl:function deklarierte Funktionen aufgerufen werden. Die Erweiterungen zur funktionalen Programmierung erlauben ebenfalls das so genannte Currying.
- Im Kontext des Streamings, aber auch darüber hinaus, erscheint die in Aussicht gestellte Nutzung von Akkumulatoren sehr interessant. Mittels xsl:accumulator lassen sich während des Prozessierens Werte auf der Basis vorhergehender Schritte berechnen, ohne dafür separate Rekursionen schreiben zu müssen. Die Spezifikation enthält ein

für die Dokumentationspraxis taugliches Beispiel zur fortlaufenden Nummerierung von Abbildungen innerhalb eines Kapitels.
- Direktes Parsen und Serialisieren von XML- oder JSON-Daten wird durch die neuen Funktionen fn:parse-xml, fn:parse-xml-fragment, fn:parse-json, fn:serialize und fn:serialize-json vereinfacht.
- Maps stellen einen neuen Datentyp zur Verfügung, der die Ablage von Informationen vergleichbar mit assoziativen Arrays in anderen Sprachen erlaubt (Key-/Value-Paare). Neben dem Konstruktor map {…} existieren spezielle Funktionen mit map-Namensraumpräfix wie u. a. map:get und map:remove.
- Mathematische Funktionen: Bisher war Numerik auf die Grundrechenarten beschränkt, die prinzipiell auch zur Formulierung komplexerer Algorithmen ausreichen (siehe die Umsetzung von Sinus und Cosinus unter [1]). Nun sind 14 Funktionen mit math-Präfix zur komfortablen Ermittlung von Exponential-, Logarithmus-, Wurzel- und Winkelfunktionswerten vordefiniert. So lassen sich beispielsweise der Sinus von 1 (Argument im Bogenmaß) über math:sin(1) = 0.479425538604203 und die Potenz 2^5 als math:pow(2,5) = 32 berechnen.
- XPath-Abfragen auf konkrete Knoten können mit xsl:evaluate dynamisch aus Zeichenketten konstruiert werden.
- Zur erweiterten Fehlerbehandlung dienen die neuen XSLT-Elemente xsl:try, xsl:catch und xsl:fallback.
- Den Zugriff auf Umgebungsvariablen des zugrunde liegenden Betriebssystems und darauf laufenden Anwendungen gestatten die Funktionen fn:environment-variable und fn:available-environment-variables (etwa nützlich für die Abfrage temporärer Verzeichnisse wie TEMP oder TMP).
- Mit Paketen und Modulen sollen sich künftig umfangreiche Transformationen besser verwalten lassen. Dazu ergänzen xsl:package und weitere Elemente das bisherige Konzept der Einbindung externer Ressourcen über xsl:include bzw. xsl:import.
- Die bisherige, eher formale Unterscheidung von XSLT- bzw. XPath-Funktionen wird aufgehoben. Funktionen wie format-date, format-dateTime, format-time, format-number, generate-id, unparsed-text können ebenfalls mit dem fn-Präfix aufgerufen werden.
- An einigen Stellen wird Feinschliff sichtbar: xsl:copy erhält ein optionales select-Attribut, fn:unparsed-text-lines erleichtert den Umgang mit unstrukturierten Informationen wie CSV-Dateien (fn:unparsed-text aus XPath 2.0 benötigte noch die Angabe der systemabhängigen Zeilenumbruchnotation), inzeilige Zeichenkettenverknüpfung mit dem „||“-Operator, Ganzzahlformatierung mit fn:format-integer und mehr.
- Die neue Funktion fn:analyze-string gibt eine XML-Struktur zurück und ist eher für den Einsatz unter XQuery 3.0 konzipiert [4].

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="3.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:err="http://www.w3.org/2005/xqt-errors"
  xmlns:fn="http://www.w3.org/2005/xpath-functions"
  xmlns:map="http://www.w3.org/2005/xpath-functions/map"
  xmlns:math="http://www.w3.org/2005/xpath-functions/math"
  exclude-result-prefixes="err fn map math xs">

  <xsl:template match="/">
     <!-- … -->
  </xsl:template>

</xsl:stylesheet>
```

Abb. 1: Prototypisches XSLT-3.0-Grundgerüst

Insgesamt handelt es sich mehr um evolutionäre als revolutionäre Neuerungen. Für ambitionierte Entwickler dürfte aber längerfristig kein Weg an den 3.0-Versionen der genannten Technologien vorbeiführen. Abbildung 1 zeigt ein Grundgerüst mit den relevanten Namensräumen.

Die Unterstützung durch Software ist noch unvollständig und hauptsächlich auf die kommerziellen Ausgaben des Prozessors Saxon 9.3/9.4 beschränkt [5]. Letztere ist bereits in der aktuellen oXygen-Version 14.0 integriert [6]. Im Vortrag werden die beschriebenen Facetten durch Codebeispiele untersetzt und praktisch demonstriert. Zur Vorbereitung wird das Studium der 2.0-Vortragsfolien empfohlen [1].

Literaturangaben und Links

[1] Meinike, T.: XSLT 2.0 und XPath 2.0 für Praktiker – Neuerungen im Überblick. In: tekom, Gesellschaft für technische Kommunikation e. V., Tagungsband zur Jahrestagung 2007 in Wiesbaden, S. 212–215 (Folien unter http://www.tekom.de/upload/2284/INF_114_Meinike_Vortrag.pdf)

[2] W3C: XSL Transformations (XSLT) Version 3.0, http://www.w3.org/TR/xslt-30/

[3] W3C: XML Path Language (XPath) 3.0, http://www.w3.org/TR/xpath-30/

[4] W3C: XPath and XQuery Functions and Operators 3.0, http://www.w3.org/TR/xpath-functions-30/

[5] Saxon Product/Feature Matrix: http://www.saxonica.com/feature-matrix.html

[6] <oXygen/> XML Editor: http://www.oxygenxml.com/

für Rückfragen: thomas.meinike@hs-merseburg.de