Prof. Dr. Stefan Brass
Institut für Informatik
MLU Halle-Wittenberg

13. Dezember 2018

XML und Datenbanken
— 8. Übungsblatt: XDM, DOM, SAX —

Bei Teil a) bis d) handelt es sich um Präsenzübungen, die nicht abgegeben werden müssen. Sie sollten über die Wiederholungsfragen a) allerdings vor der nächsten Übung selbst nachdenken. Teil e) ist eine Hausaufgabe. Bitte schicken Sie eine Lösung per EMail an den Dozenten (mit "`xml18`" in der Betreff-Zeile, bis zum 19.12.2018). Schicken Sie bitte keine leeren EMails nur mit Anhang, diese landen im Spam-Ordner.

a) Wie würden Sie in einer mündlichen Prüfung auf folgende Fragen zum XPath/XQuery Data Model (XDM) antworten?

- Die wesentliche Datenstruktur, mit der XPath und XQuery arbeiten, ist eine Sequenz. Wie ist das definiert?
- Was sind die sieben Arten von Knoten? Welche Knoten können Kind-Knoten sein, welche Eltern-Knoten?
- Warum braucht man einen extra Dokument-Knoten und verwendet nicht einfach einen Wurzel-Element-Knoten?
- Nennen Sie einige syntaktische Details, die beim Einlesen/Parsen einer XML Datei in die interne XDM-Repräsentation verloren gehen.
- Warum gibt es noch den XML InfoSet Standard, und nicht nur den XDM Standard?
- Welche Unterschiede bestehen zwischen einer XDM-Baumstruktur, die ohne Validierung aufgebaut wird, und einer, die aus einem "Post-Validation InfoSet" aufgebaut wird?
- Beschreiben Sie die Behandlung von Leerplatz zwischen Elementknoten.
- Warum sollen Namenspace-Knoten nicht mehr verwendet werden? Was ist das Problem?
- Was ist die "Document Order"?
- Welche Beziehung besteht zwischen
  - "`anyType`",
  - "`anySimpleType`",
  - "`anyAtomicType`" und
  - "`untypedAtomic`"?

  Erklären Sie die Typ-Hierarchie und nennen Sie jeweils ein Beispiel für einen Typ, der noch zum jeweiligen Obertyp gehört, aber nicht zum Untertyp.

- Was ist der String-Value von Element-Knoten? Und der typed value, wenn das Dokument validiert wurde? Hier müssen Sie Elemente mit einfachem Inhalt, Elemente mit reinem Element-Inhalt und Elemente mit "mixed content model" unterscheiden.

Präsenzaufgaben

b) Laden Sie die folgenden Dateien herunter:

- [http://users.informatik.uni-halle.de/˜brass/xml18/xsl/ex1_query.xml]
- [http://users.informatik.uni-halle.de/˜brass/xml18/xsl/ex2_query.xml]
- [http://users.informatik.uni-halle.de/˜brass/xml18/xsl/query.xsl]

Sie können auch das Verzeichnis im Browser öffnen, um auf die Dateien zuzugreifen:

[http://users.informatik.uni-halle.de/˜brass/xml18/xsl/]

Das Stylesheet zeigt informationen über die XDM Knoten an, die beim Parsen der XML Dateien erzeugt werden. Wenn Sie die XML-Dateien in einem Browser öffnen, sollte Ihnen eine HTML Liste der Knoten angezeigt werden, und zu jedem Knoten der Typ, der Name und der "String Value". Vergleichen Sie dies mit dem XML Code in den Dateien. (Sie können den Quellcode auch im Browser mit "View Source" anzeigen, `Ctrl+U` im Firefox.)

c) Experimentieren Sie mit der XPath-Anfrage in der Datei `query.xsl`. Eine ganz kleine XPath-Teilmenge wurde im Kapitel über XML Schema vorgestellt. Sie könnten z.B. XPath-Anfragen schreiben, um die Nachnamen aller Studenten zu selektieren.

d) In der Vorlesung wurde ein Beispiel-Programm für die SAX-Schnittstelle besprochen. Sie können eine leicht erweiterte Variante davon hier herunterladen:

[http://users.informatik.uni-halle.de/˜brass/xml18/SaxExample.java]

Compilieren Sie das Programm mit

```
javac SaxExample.java
```

Probieren Sie es dann mit verschiedenen Eingabedateien aus:

```
java SaxExample ⟨Datei⟩.xml
```

Sie können das Programm auch modifizieren.

Hausaufgabe

e) Falls noch nicht geschehen, informieren Sie sich bitte über die DOM-Schnittstelle, um XML-Daten in Java-Programmen verarbeiten zu können. Z.B. könnten Sie das folgende Tutorial lesen:

[https://docs.oracle.com/javase/tutorial/jaxp/dom/index.html]

Weitere Tutorials sind:

- [https://www.tutorialspoint.com/java_xml/java_dom_parser.htm]
- [https://www.mkyong.com/java/how-to-read-xml-file-in-java-dom-parser/]

Die DOM-Spezifikation wurde vom W3C entwickelt:

- [https://www.w3.org/TR/DOM-Level-3-Core/]
- [https://www.w3.org/DOM/]

DOM wird auch benutzt, um HTML in JavaScript-Programmen zu manipulieren. Für XML ist aber nur der "DOM Core" relevant.

Die Hausaufgabe ist nun, ein kleines Java-Programm zu schreiben, das die SIDs aller Studenten druckt, die 10 Punkte in Hausaufgabe 1 haben. Diese Aufgabe könnte natürlich auch mit einer ganz kurzen XPath-Anfrage gelöst werden, aber es ist nur ein Beispiel für die Verarbeitung von XML-formatierten Daten in Java.

Sie dürfen wählen, ob Sie die DOM- oder die SAX-Schnittstelle verwenden. Die SAX-Schnittstelle wurde in der Vorlesung besprochen, Sie könnten das Beispielprogramm passend modifizieren.

Sie dürfen auch wählen, welche Codierung der Daten in XML Sie bevorzugen:

- Die XML-Datei mit den Daten in Attributen:

  [http://users.informatik.uni-halle.de/~brass/xml18/examples/ex1.xml]

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<GRADES-DB>
<STUDENT SID='101' FIRST='Ann' LAST="Smith"
        EMAIL='smith@acm.org'/>
    <STUDENT SID='102' FIRST='Michael' LAST='Jones'/>
    ...
    <EXERCISE CAT='H' ENO='1' TOPIC='Relational Algebra' MAXPT='10'/>
    ...
    <RESULT SID='101' CAT='H' ENO='1' POINTS='10'/>
    <RESULT SID='101' CAT='H' ENO='2' POINTS= '8'/>
    ...
</GRADES-DB>
```


- Oder die Version mit den Daten in geschachtelten Elementen:

  [http://users.informatik.uni-halle.de/~brass/xml18/examples/ex2.xml]

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<GRADES-DB>
    <STUDENTS>
        <STUDENT>
            <SID>101</SID>
            <FIRST>Ann</FIRST>
            <LAST>Smith</LAST>
            <EMAIL>smith@acm.org</EMAIL>
        </STUDENT>
        ...
    </STUDENTS>
    <EXERCISES>
        <EXERCISE>
            <CAT>H</CAT>
            ...
        </EXERCISE>
        ...
    </EXERCISES>
    <RESULTS>
        <RESULT>
            <SID>101</SID>
            <CAT>H</CAT>
            <ENO>1</ENO>
            <POINTS>10</POINTS>
        </RESULT>
        ...
    </RESULTS>
</GRADES-DB>
```