WBA2 SS13 - Phase 1 Ausarbeitung

Ngoc-Anh Dang 11082691

Abgabe am 12.04.2013

1 Wohlgeformtheit, Validität, Namespaces

Erklären Sie kurz die Begriffe Wohlgeformtheit, Validität und Namespaces im Bezug auf XML und XML-Schema.

Wohlgeformtheit Wenn ein XML-Dokument wohlgeformt ist, bedeutet das, dass das XML-Dokument „lesbar“ ist. Das XML-Dokument muss dazu eine korrekte Baumstruktur aufweisen. Konkrete Bedingungen dafür sind beispielweise:

- je XML-Dokument darf nur 1 Wurzelknoten existieren
- ein Element eines XML-Dokuments mit Inhalt muss zwischen einem öffnenden und schließenden Tag sein
- außer das Wurzelelement muss jedes Element von Elementen vollständig umschlossen sein
- ein Attributwert muss zwischen ’ oder “ stehen
- die in dem XML-Dokument verwendeten Zeichen müssen dem festgelegten Zeichensatz entsprechen

Validität Wenn ein XML-Schema valid ist, bedeutet das, dass die Syntax den formulierten Festlegungen gegenüber korrekt ist. Man kann zum einen ein XML-Schema selbst validieren und zum anderen ein XML-Dokument gegen ein XML-Schema validieren. Im letzteren Fall prüft man, ob diese einander entsprechen.

Namespaces Namespaces oder auf deutsch „Namensräume“ dienen zur Festlegung des Vokabulars eines XML-Dokuments. Dies bietet einem die Möglichkeit mehrere Namensräume zu verwenden, ohne, dass es Uneindeutigkeiten z.B. bei gleichen Elementnamen gibt (Verhinderung von sog. „clashing names“, also Unterstützung von sublanguages). Die Namensräume werden eindeutig durch eine URI identifiziert und durch das Attribut „xmls“ beschrieben. I.d.R. werden die Namensräume zu Anfang einmal mit Präfixen (z.B. xmls:xsd) deklariert, welche dann im folgendem Schema die Element-, Typ-Namen, etc. eindeutig bestimmen. Der default namespace lautet xmls=““; der für XML-Schema ist xmlns="http://www.w3.org/2001/XMLSchema".

2 XML, JSON

2.1 XML

Erzeugen Sie ein XML-Dokument, dass die Daten des folgenden Formulars vollständig erfasst: http://www.gm.fh-koeln.de/~vsch/anmeldung/gruppenanmeldung.html . Füllen Sie das Dokument mit einem Beispieldatensatz. Achten Sie darauf, dass über das Formular mehrere Personen gleichzeitig erfasst werden können. Wichtig: Es sollte nicht die HTML-Struktur der Webseite in der XML-Datei abgebildet werden, sondern die zu übertragenden Daten.

Lösung xml/aufgabe2a.xml

In dieser Lösung lautet der Wurzelknoten „Anmeldungen“. Dieser kann mehrere Kindknoten vom Typ „Gruppe“ haben, welche eindeutig anhand der „gruppen_id“ identifizierbar sind. Desweiteren hat jede Gruppe genau einen Gruppenleiter, welcher als Attribut im Gruppenelement angegeben wird. Das Gruppenelement hat wiederum Kindelemente, welche die Daten Mitglieder dieser erfassen. Jedes Mitglied stellt einen Knoten dar. Die persönlichern Daten und Informationen zum Schlagzeug, sowie eine Mitglieds-id werden in dieser Lösung als Attributen gespeichert, da sie eine noch übersichtliche Menge sind und als Eigenschaften eines Mitglieds beschrieben werden, anstatt als „Bestandteile“ (wie es bei weiteren Kindknoten interpretierbar wäre). Die Mitglieds-id des Gruppenleiters wird im Attribut „Gruppenleiter“ des Gruppenelements gespeichert.

Zu beachten ist, dass im Gegensatz du den restlichen Attributen die id-Attribute nicht eingegeben werden, sondern dies automatisch vergeben wird. Dies könnte für Verwirrung sorgen. Um dies zu umgehen könnte man ein Attribut wie „rolle“ einführen, in der entweder „gruppenleiter“ oder „mitglied“ gespeichert wird. Da diese Verwirrung jedoch die Teilnehmer nicht betrifft, sondern eventuell nur Programmierer betreffen, erscheint mir dieses Contra-Argument nicht zureichnend um die Lösung zu ändern.

Desweiteren kann in diesem XML-Dokument nicht sichergestellt werden, dass eine Gruppe mindestens 2 Mitglieder haben muss. Dies könnte man jedoch in einem zugehörigem XML-Schema festlegen.

Eine weitere alternative Lösung ist anstatt der vielen Attribute, diese als Kindelemente darzustellen. Ich habe mich gegen diesen Lösungsweg entschieden, da zum Einen der o.g. Grund ein Argument ist und zum Anderen diese Variante Zeilen spart. Meines Erachtens ist ein solch komplexer Baum, also ein Baum mit vielen Ebenen und Kindknoten, bei dieser nicht-komplexen Aufgabenstellung nicht notwendig.

2.2 JSON

Erzeugen Sie ein JSON-Dokument, dass zu ihrem XML-Dokument äquivalent ist.

Lösung xml/aufgabe2b.json

Diese Lösung ist komplett am XML-Dokument der Teilaufgabe a) orientiert und behinhaltet nur eine kleine Änderung: Unter dem Attribut „schlagzeug“ wird kein String mehr gespeichert („vorhanden“/ „nicht vorhanden“), sondern boolean-Werte (true/ false). In XML gibt es ebenfalls den Datentyp boolean, dies habe mir im Zeitpunkt der Bearbeitung der Aufgabe noch nicht erarbeitet. Diese Variante ist effizienter, da sie „kürzer“ sind als Strings. D.h. z.B. muss auf diese Weise im Schema kein „choice“ programmiert werden und spart somit Komplexität ein. Bis auf diesen Aspekt sind beide Dokumente äquivalent.

3 Rezept XML

3.1 Gegeben ist folgendes Rezept: http://www.chefkoch.de/rezepte/24641006006067/Lench s-Schokoladenkuchen.html Entwickeln Sie ein XML-Dokument, in dem die Daten des Rezeptes abgebildet werden. Achten Sie darauf, dass das Dokument semantisch möglichst reichhaltig ist. Bei dieser und den folgenden Aufgaben lassen sie bitte die Daten in der Marginalspalte auf der rechten Seite weg.

Lösung xml/aufgabe3a.xml

Bei diesem XML Dokument bin ich ähnlich wie beim XML-Dokument der Nr. 2a vorgegangen. Desweiteren habe ich mehr mit Attributen gearbeitet, anstatt mit Kindelementen (id, Arbeitszeiten, Kalorien, Portionen, Schwierigkeit, ...), wieder mit dem Argument, dass dich diese eher als allgemeine Eigenschaften interpretiert habe, antatt als Bestandteile eines Rezeptes. Außerdem lassen sich ihre Daten relativ kurz darstellen (ein Wort oder eine Zahl). Die Elemente Bilder, Zutaten, Schritte und Kommentare dahingegen besitzen, meines Erachtens, eine etwas komplexere Struktur und sind in wiederum, semantisch sinnvolle Unterelemente unterteilbar. Als „Bruch“ dieser Vorgehensweise könnte man meine Umsetzung der Schritte ansehen. Hier habe ich jeden einzelnen Schritt als ein eigenes Element dargestellt, anstatt als einen Fließtext. Diese Modularisierung erschien mir sinnvoll, da ich in vielen Anwendungsfällen bereits erlebt habe, dass ein Prozess - als was man eine Zubereitung ansehen kann - step by step dargestellt wird. Als Beispiel könnte man einen Screenreader nennen, der die Schritte vorliest und zwischen diesen Pausen einlegt, anstatt diesen in einem Zug vorliest.

3.2 Betrachten Sie nun andere Rezepte auf der Webseite http://www.chefkoch.de. Beschreiben Sie welche Gemeinsamkeiten die Rezepte hinsichtlich ihrer Daten haben und worin Sie sich unterscheiden.

Gemeinsamkeiten Daten, welche in jedem Rezept vorkommen sind Titel des Rezepts, Angabe der Zutaten und dessen Menge und die Angabe der Zubereitungsschritte.

Unterschiede Daten, die nicht immer in jedem Rezept auftauchen sind die Zutaten selbst. Desweiteren gibt es je Rezept verschiedene Einheiten (nach Belieben, ein Spritzer, eine Dose) und neue Eigenschaften, wie Ruhezeit, Koch-/Backzeit. Manche Angaben wie der Brennwert oder die Portionenangabe fehlen bei anderen Rezepten. Nicht jedes Rezept hat ein Bild. Außerdem gibt es Unterschiede im Kommentar-Abschnitt: Nicht jedes Rezept hat Kommentare oder manche Kommentare enthalten Antwort-Kommentare.

3.3 Arbeiten Sie die Kriterien heraus, die für die Entwicklung einer XML-Schema-Datei beachtet werden müssen. Die Schema-Datei soll die Struktur für eine XML-Datei definieren, in der mehrere unterschiedliche Rezepte gespeichert werden können. Ziel ist es, dass das XML-Schema möglichst restriktiv (einschränken) ist, so dass in der XML-Datei möglichst semantisch sinnvolle Daten bezüglich der Rezepte gespeichert werden können. Ziehen Sie beim Aufstellen der Kriterien u.A. folgende Fragestellungen in Betracht: Welche Daten müssen in simple und welche in complex-types abgebildet werden? Für welche Daten ist die Abbildung in Attributen sinnvoller? Welche Datentypen müssen für die Elemente definiert werden? Welche Restriktionen müssen definiert werden?

Bei der Entwicklung einer XML-Schema-Datei muss man wissen, was als Element und was als Attribut definiert werden soll. Nach weiteren Recherchen und entgegen meiner bisherigen Vorgehensweise ist es bei einem Schema eher zu vermeiden Attribute zu verwenden. Ein Grund dafür ist, dass die Struktur der Daten verloren geht bzw. stark vereinfacht wird und damit die Lesbarkeit verloren gehen kann. Desweiteren können Attribute keine geordnete Werte enthalten und sind schwer zu Erweitern, im Gegensatz zu Elementen, zu welchen in einer bestimmten Reihenfolge Unterelemente hinzugefügt werden können. Attribute eigenen sich für Werte, die nur einmal im Dokument vorkommen, sowie Identifikatoren. Außerdem die Primärfunktion von XML der Datenaustausch, daher sollten alle Informationen, welche deutlich als Daten wahrgenommen werden, als Elemente definiert werden.

Ein weiteres Kriterium für die Entwicklung einer XML-Schema-Datei ist, dass man sich ein gutes Bild der Struktur machen sollesynon, damit man weiß, welche Elemente welche und wieviele Kindelemente haben. Überdies kann es auch von Bedeutung sein, in welcher Reihenfolge die Elemente und Attribute beschrieben werden. Eine weitere Überlegung ist ob man ein Element als „leer“ also nur mit Attributen definiert, oder mit Text zwischen den Tags definiert. Darüberhinaus sollten die Datentypen (simple-/complex-type) der Elemente und Attribute bekannt sein. Dazu ist es sinnvoll sich sich überlegen, ob es default-Werte geben sollte oder feste Werte oder Wertebereiche.

Aufgrund diesen neuen Erkenntnissen, weichen meine Antworten teilweise von dem XML-Dokument der Teilaugfabe a) ab.

Welche Daten müssen in Simple- und welche in Complex-Types abgebildet werden und welche Datentypen müssen für die Elemente definiert werden?

SimpleType

anyURI bild_url

string user, zutatname, schritt, kommentar

positiveInteger stunde, minuten, brennwert

decimal menge

dateTime datum/uhrzeit

ComplexType chefkochbuch, rezept, bilder, zutaten, zutat, schritte, kommentare, back-/kochzeit

Für welche Daten ist die Abbildung in Attributen sinnvoller?

In diesem Fall sehe ich nur die „id“ als sinnvolles Attribut an, da dies die einzige Information ist, welche nicht als Datum für den Leser relevant ist und sozusagen zu den Metadaten gehört.

Welche Restriktionen müssen definiert werden?

- Bei den Mengenangaben der Zutaten dürfen nur gültige Einheiten verwendet werden.
- Bei den Schwierigkeitsstufen darf nur zwischen leicht, mittel und schwer gewählt werden.

- Die Angabe der Minuten dürfen nur zwischen 0 und 59 sein.
- Manche Angaben sind optional (ruhezeit, bilder, kommentare, brennwert), andere verpflichtend (menge, einheit, portionen, id, rezeptname, zutatname).
- Es sollte mindestens ein Rezept, eine Zutat und ein Schritt geben.

3.4 Erstellen Sie nun ein XML-Schema auf Basis ihrer zuvor definierten Kriterien. Generieren Sie nun auf Basis des Schemas eine XML-Datei und füllen Sie diese mit zwei unterschiedlichen und validen Datensätzen.

\footnote[1]{http://www.w3schools.com/dtd/dtd_el_vs_attr.asp, http://www.teialehrbuch.de/Kostenlose-Kurse/XML/7743-Attribute-oder-Elemente.html (Zugriff: 08.04.2013)}