IN22 Fachvortrag

Verarbeitung von JSON-Daten mit aktuellen XSLT-Techniken

Dr. Thomas Meinike, Hochschule Merseburg

Motivation

Von Organisationen und Verwaltungen werden zunehmend Datensätze als Open Data publiziert [1]. Neben klassischen Ansätzen wie CSV und XML kommt dabei das JSON-Format zum Einsatz, wobei die Unabhängigkeit von bestimmten Programmiersprachen eine flexible Verarbeitung ermöglicht. Auch die aktuellen Spezifikationen von XSLT 3.0 und XPath 3.1 stellen Techniken für den Zugriff auf JSON-kodierte Daten zur Verfügung. Dieser Beitrag widmet sich den wesentlichen Verarbeitungsansätzen und demonstriert dynamische Abfragen ausgewählter Daten mittels XSLT unter Verwendung des im Browser lauffähigen Prozessors Saxon-JS.

Einstieg in JSON

JSON ist das Kürzel der JavaScript Object Notation, die sich seit Jahren großer Beliebtheit in der JavaScript-Szene und darüber hinaus erfreut. Ursprünglich wurde die Sprache 2001 von Douglas Crockford – u. a. Buchautor von „JavaScript: The Good Parts“ – entworfen und ist mittlerweile im ECMA-Standard 404 aufgegangen [2].

Die Kodierung von Daten erfolgt mit den aus der JavaScript-Syntax bekannten grundlegenden Konstrukten Object {…}, Array […], Zeichenketten in Anführungszeichen (String), Zahlenwerten als ganze oder Fließkommazahlen (Number), den Wahrheitswerten true bzw. false (Boolean) sowie dem Wert null. Einzelne Datenobjekte werden in Form von Key-Value-Paaren abgebildet. Dabei steht links von einem Doppelpunkt der Bezeichner und rechts davon der jeweilige Wert. Die Abgrenzung von Blöcken erfolgt durch Kommasetzung. Eine verschachtelte Kombination dieser Möglichkeiten führt zu konkreten Strukturen [3]. Abbildung 1 zeigt einen kompakten Ansatz zur Ablage von Informationen für Konferenzen.

```
{
  "konferenzen": [
    {
      "kid": "k01",
      "name": "tekom-Jahrestagung 2018",
      "ort": "Messe Stuttgart",
      "tage": 3,
      "datum": ["2018-11-13", "2018-11-14", "2018-11-15"],
      "infos": "https://tagungen.tekom.de/",
      "topevent": true
    },
    { weitere Einträge … }
  ]
}
```

Abb. 1: JSON-Beispiel für Konferenzdaten

Innerhalb der äußeren Objekthülle ist ein Array mit dem Namen „konferenzen“ deklariert. Darin sind die eigentlichen Daten, wiederum in Objektnotation, hinterlegt: eine ID, der Name der Konferenz

und ihr Austragungsort als Zeichenketten, die Anzahl der Tage als Ganzzahlwert, ein weiteres Array „datum“ mit den Tageszuweisungen, die zugehörige Webadresse unter „infos“ und eine Bewertung als „topevent“ mit den Möglichkeiten true oder false. Weitere Einträge lassen sich in der angedeuteten Weise vornehmen.

Zur praktischen Modellierung und Validierung kann JSON Schema verwendet werden. Dabei handelt es sich selbst um JSON-Strukturen [4]. Ein zur gezeigten Konferenzstruktur passendes Schema wird im Vortragsmaterial enthalten sein.

Praktische Nutzung von JSON

Abgrenzung

Dieser Beitrag beschäftigt sich im Wesentlichen mit der Nutzung vorhandener oder selbst gefertigter JSON-Ressourcen, insbesondere mit dem eingangs genannten Fokus auf XSLT-Techniken. Betont werden soll ausdrücklich, dass JSON einen überwiegend datenorientierten Ansatz liefert, während sich XML für stärker dokumentgeprägte Anwendungen eignet. Insofern haben also beide Technologien je nach Anwendungszweck ihre Vorteile und sollen hier nicht gegeneinander antreten.

XSLT und XPath

Die Transformationssprache XSLT existiert in Version 1.0 seit 1999, wurde 2007 als Version 2.0 vielfältig erweitert und schließlich 2017 mit Version 3.0 nochmals funktional aufgewertet [5]. Auch die zugehörige Abfragesprache XPath für Knoten und Inhalte hat diese Evolution durchlebt und steht seit 2017 in Version 3.1 zur Nutzung bereit [6]. Wesentliche „Goodies“ der aktuellen Spezifikationen beschreiben Birnbaum [7] und Cagle [8]. Das kürzlich erschienene XML-Fachbuch [9] und die zugehörige Online-Präsenz seien zur umfassenden Erschließung der Möglichkeiten empfohlen.

JSON-Techniken

Zentrale Techniken basieren auf den neuen Datentypen Array und Map [10], die direkt zur Nutzung von JSON einladen. Auf externe Quellen kann über die XPath-Funktion fn:json-doc() zugegriffen werden. Das Ergebnis lässt sich durch Abfragen in der (erweiterbaren) Form $jsonobject?name auswerten. Unmittelbar innerhalb einer Transformation als Zeichenketten formulierte JSON-Daten sind über fn:parse-json() zugänglich. Umwandlungen zwischen JSON und XML bewerkstelligt man mit fn:json-to-xml() sowie fn:xml-to-json().

Für die Nutzung der Datentypen Array und Map stehen weitere spezifische Funktionen zur Verfügung, die von den Konzepten anderer Sprachen adaptiert wurden, z. B. array:get(), array:append(), array:join() bzw. map:find(), map:put(), map:for-each().

Beispielanwendung

Zur Demonstration grundlegender Methoden wurde eine Beispielanwendung entwickelt, die auf einen öffentlichen JSON-Datensatz zugreift und abgefragte Inhalte auf einer Website darstellt. Es handelt sich um die Liste der in München zugänglichen Parkhäuser, vgl. [11] und Abbildung 2.

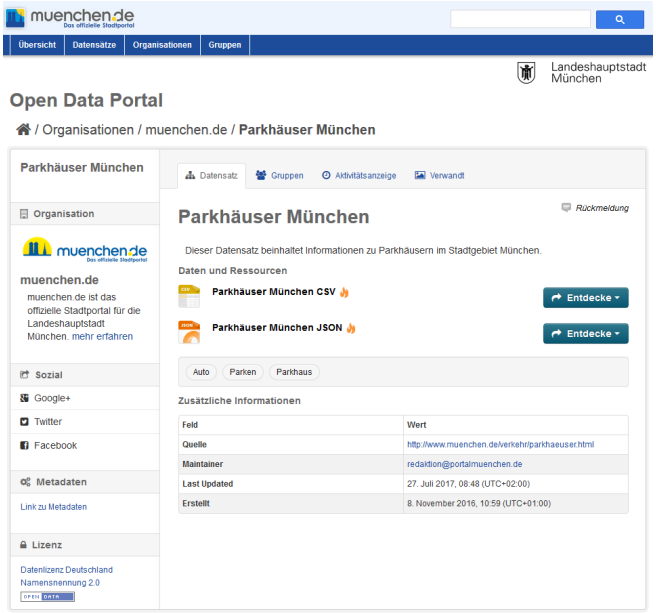

Abb. 2: Datenquelle Parkhäuser München [11]

Die JSON-Daten wurden mit dem JavaScript-basierten XSLT-Prozessor Saxon-JS [12] im Browser verarbeitet. Unter [13] hat der Autor den Einsatz von Saxon-JS bereits beschrieben und praktisch gezeigt. Das Ergebnis zu den Parkhäusern kann unter [14] abgerufen werden. Abbildung 3 zeigt einen konkreten Datensatz von gegenwärtig insgesamt 72, die in einer Auswahlliste stehen. Es erscheint jeweils ein Auszug relevanter Daten hinsichtlich ID, genauer Verortung der Straße, ggf. vorliegendem Foto, einem Nutzer-Rating und dem Link zu weiteren Details. Da auch die zugehörigen Geokoordinaten (Latitude / Longitude) enthalten sind, werden diese auf eine OpenStreetMap-Karte [15] projiziert, die als IFrame im HTML-Dokument eingebunden wird.

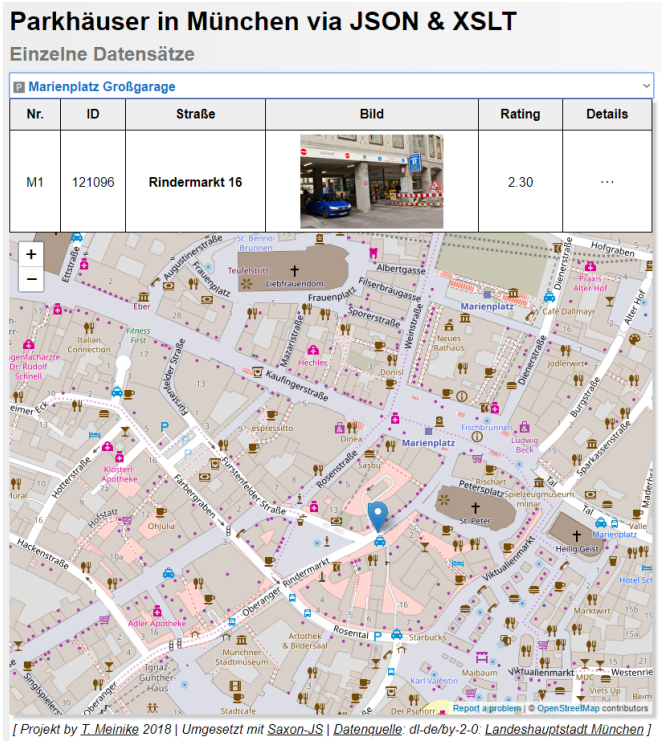

Abb. 3: Beispielausgabe zu einem Parkhaus-Datensatz [14]

Fazit und Ausblick

Die im Werkzeugkasten von XSLT und XPath hinzugekommenen JSON-Techniken erweitern das Methodenspektrum zur Verarbeitung strukturierter Daten. Das Beispielprojekt wird im Vortrag anhand von Codeauszügen detaillierter untersetzt. Interessant ist dabei auch die Realisierung der Karte, speziell die Ermittlung der so genannten Bounding Box für die darzustellenden Kacheln. Nötige Berechnungen werden ebenfalls mit neueren mathematischen, u. a. trigonometrischen, Funktionen durchgeführt. Weitere Aspekte bilden Sortierung und Gruppierung der Datensätze.

Literaturangaben und Links

[1] GovData: Das Datenportal für Deutschland. https://www.govdata.de/

[2] ECMA: Standard ECMA-404 – The JSON Data Interchange Syntax. https://www.ecma-international.org/publications/standards/Ecma-404.htm

[3] Introducing JSON. https://json.org/

[4] JSON Schema. https://json-schema.org/

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

[6] W3C: XML Path Language (XPath) Version 3.1. https://www.w3.org/TR/xpath-31/

[7] Birnbaum, D. J.: What's new in XSLT 3.0 and XPath 3.1? http://dh.obdurodon.org/xslt3.xhtml

[8] Cagle, K.: Why You Should Be Using XSLT 3.0. https://www.xml.com/articles/2017/02/14/why-you-should-be-using-xslt-30/

[9] Grupe, W.: XML – Technologien | Grundlagen | Validierung | Auswertung. mitp-Verlag 2018. Online-Material: http://www.wilfried-grupe.de/

[10] W3C: XQuery and XPath Data Model 3.1. http://www.w3.org/TR/xpath-datamodel-31/

[11] OpenData-Portal der Landeshauptstadt München: Parkhäuser München. https://www.opengov-muenchen.de/dataset/parkhaeuser-munchen

[12] Saxonica: Saxon-JS. https://saxonica.com/html/saxon-js/

[13] Meinike, T.: Interaktive XML-Verarbeitung im Browser mit Saxon-JS und XSLT 3.0. In: tekom, Gesellschaft für technische Kommunikation e. V., Tagungsband zur Jahrestagung 2017, S. 177–180. https://datenverdrahten.de/PDF/tekom2017_IN29_Meinike.pdf

[14] Meinike, T.: Parkhäuser in München via JSON und XSLT. https://datenverdrahten.de/xslt3/saxon-js/parkhaeuser/

[15] OpenStreetMap Foundation: OpenStreetMap. https://www.openstreetmap.org/about

Kontakt:
thomas.meinike@hs-merseburg.de