RDFa Crawler
Webcrawler inklusive RDFa Parser

STEFAN ACHMÜLLER, ROLAND GRITZER, MATHIAS GSCHWANDTNER
Universität Innsbruck
February 26, 2017

Zusammenfassung

Ein auf JavaScript basierte Bibliothek zur Durchsuchung von RDFa annotierten Informationen, welche über Internetzugriff einsehbar sind. Ausgehend von einer URL, mit optional möglichem Ver- und Gebietens von URL Domänen, werden zusammenhängende XHTML Dokumente nach RDF Trippel durchsucht.

Schlüsselwörter: webcrawler, parser, rdfa, node, npm

Inhaltsangabe

1 Problem Definition

Mit der historischen Entwicklung des Internets wurden immer wieder Techniken eingeführt, die den Datenaustausch zwischen Rechnern vereinfachen. Während im klassischen Internet, auch Web 1.0 (URL, HTTP, HTML, etc.) genannt, der Fokus auf dem Aufbau und Transport der Daten liegt, betrachtet der Ansatz "Semantic Web" mögliche Interpretationen der Daten. Dies wird auch Web 3.0 genannt und ermöglicht eine vereinfachte Abarbeitung von Aufgaben, basierend auf Internetdaten. So kann zum Beispiel unterschieden werden, ob das Wort "Bremen" auf einer bestimmten Webseite sich entweder auf eine deutsche Stadt, einen Familiennamen oder einem sonstigen Namen bezieht. Die Kerneigenschaft des "Semantic Web" stellt die Universalität der Relationten, sowie die maschinelle Interpretationsömglichkeit der Informationen dar. Dies bedeutet, dass prinzipiell alle Informationsobjekte miteinander verknüpft werden können um Wissen zu repräsentieren und verarbeiten [1].

Um Daten mit diesen Metainformationen anzureichern, wurden diverse Annotationen eingeführt. Neben JSON-LD und Microdata, bietet sich hierfür das "Resource Description Frameworks" (RDF) an. Ziel dieser Arbeit ist die Erstellung einer Programmierbibliothek für den JavaScript basierten "Node Package Manager" (npm) zum automatischen extrahieren von kontextspezifischen Informationen, Schema.org Vokabular annotiert mittels RDFa Syntax (W3C Standard Annotation für RDF), welche in XHTML Webseiten eingebettet sind [2], [3].

1.1 RDFa annotierte Information

RDF definiert eine Schnittstelle für diverse Annotationen zur Darstellung von semantischen Informationen. Zu diesem gehören die schematische Darstellung von Informationen als Graphen. Ein Graph ist hierbei eine Menge von Tripeln, welche einzelne Informationen repräsentieren. Ein Tripel besteht jeweils aus drei verschiedenen Knoten, die in folgender Reihe definiert sind.

- **Subjekt** der Information (z.B. "Hannes ...").
- **Prädikat** der Information (z.B. "... wohnt in ...").
- **Objekt** der Information (z.B. "... Innsbruck").

Jeder Knoten ist entweder ein "International Resource Identifier" (IRI), oder ein "Blank Node", welcher typischerweise als einmalige Zeichenketten (String) dargestellt wird [4]. Teilziel der Arbeit ist die Filtern von RDFa

annotierten Information von Dokumenten im XHTML Format. Der implementierte Prozess der Erstellung eines RDF Graphen basiert auf der "RDFa Core Sequence" [5].

1.2 Webcrawler

Der zweite Teil der Arbeit, bezieht sich auf den Bezug von XHTML formatierten Dokumenten, welche RDFa annotierte Information enthalten. Mittels der Methode eines Webcrawlers wird auf im Internet zugreifbare Daten automatisch zugegriffen. Dabei definiert der Benutzer einen Startpunkt, ein "Uniform Resource Locater" (URL), auf den dieser Zugriff hat. Ausgehend von dieser Ressource, typischerweise ein XHTML Dokument, werden weiterführende URLs durchsucht. Wie diese Durchsuchung im Einzelnen abläuft, zum Beispiel welche Datenformate nicht durchsucht werden sollten, wird vom Benutzer festgelegt [6]. Teilziel der Arbeit ist der Zugriff auf zusammenhängende XHTML formatierte Daten, ausgehend von einer festgelegten URL.

2 Methodologie

Nach der Definition der Aufgabenstellung, wurden die Schritte zur Problemlösung, inklusive einem groben Zeitplan, erstellt. Zum einen dient diese Strukturierung zur leichteren Abgrenzung von Teilproblemen und deren Bearbeitung einzelner Teammitglieder, sowie der Einhaltung strikter Deadlines.

2.1 Recherche und Setup

Die Einarbeitung in die Basisthemen der Arbeit wurde gemeinsam im Team vorgenommen. Folgende Themen und deren Schwerpunkte wurden recherchiert. Im Anschluss wurde mittels Node ein lokaler Server für Test- und Präsentationszwecke erstellt.

- **JavaScript, Node und npm** - verwendete Programmiersprache, serverseitiges Framework und dazugehörirger Paketverwaltung.
- **Schema.org und RDFa** - verwendetes Vokabular und Annotation der Daten.

2.2 Implementation und Tests

Aufgrund der Ergebnisse der Recherche wurde das Erstellen und Testen des Programmcodes unter den Teammitgliedern aufgeteilt. Zeitgleich er-

folgte die Dokumentation der beiden Module, mittels Kommentaren im Programmcode und einer "Readme" Datei.

- **Modul 1: RDFa Parser**
  Mittels dem npm Modul "cheerio" wird auf Daten eines vorliegenden XHTML Dokuments zugegriffen. Im Anschluss werden mittels der RDFa Core Sequenz und dem npm Modul "rdf", RDF Triple erzeugt [7], [8].
- **Modul 2: Webcrawler**
  Mittels dem npm Modul "simplecrawler" wird, ausgehend von einer vorgegebenen URL und den Benutzereinstellungen, eine Liste von zusammenhängenden URLs erstellt. Dabei wird nach jedem Zugriff auf eine URL Modul 1 aufgerufen. Zu den Benutzereinstellungen zählt das "Black-/Whitelisten", daher die Beschränkung der URL Domänen, sowie die Suchtiefe, Länge der Verlinkung ausgehend von der Start URL [9].

2.3 Zusammenführung und Publikation

Im Anschluss wurden die einzelnen Module zusammengeführt und nochmals anhand des "RDFa Test Suite Manifest" getestet, bevor das finale Programm als npm Modul veröffentlicht wurde [10].

3 Ergebnis

Die zusammengeführten Module wurden als einzelnes Paket über npm veröffentlicht [11]. Das Paket bietet zwei Methoden, "parseRDFa" und "crawler" an, um RDFa Inhalte aus dem "World Wide Web" zu extrahieren.

parseRdfa(html, base)
- mit Rückgabe der RDFa Triples in einer Liste.

- *html (String)* - das HTML Dokument, welches RDFa Informationen beinhaltet.
- *base (String)* - die URL, die dem HTML Dokument zugrunde liegt.

crawler(start, depth, callback, whitelist, blacklist)
- ohne Rückgabewert.

- *start (String)* - die URL, von der aus die Funktion startet.
- *depth (int)* - Tiefe der URL Links, die durchsucht werden sollen.

- *callback (Function)* - Funktion, die die gefundenen HTML Dokumente bearbeitet (RDFa Triple extrahieren und weiterverarbeiten).
- *whitelist (String Array)* - Optionale Liste mit Subdomains, die durchsucht werden sollen.
- *blacklist (String Array)* - Optionale Liste mit Subdomains, die niht durchsucht werden sollen.

3.1 Beispielanwendung

Der folgende Code implementiert beispielhaft die Verwendung des erstellten Moduls in Abhängigkeit des "Request" npm Moduls [12].

npm Packet installieren:

```
npm install rdfa-parser
```

Parsen eines einzelnen HTML Dokuments:

```
var rdfaParser = require("rdfa-parser");
var request = require("request");
let base = "http://booking.com";

request(base, function (error, response, html) {
    let triples = rdfaParser.parseRDFa(html, base);
    for (let i = 0; i < triples.length; i++) {
        console.log(triples[i].toString());
    }
});
```

Parsen mehrerer HTML Dokumente:

```
var rdfaParser = require("rdfa-parser");
var request = require("request");
let start = "http://booking.com";
let depth = 2;

rdfaParser.crawler(start, depth, function (base) {
    request(base, function (error, response, html) {
        let triples = rdfaParser.parseRDFa(html, base);
        for (let i = 0; i < triples.length; i++) {
            console.log(triples[i].toString());
        }
    });
});
```

Referenzen

1. Tim Berners-Lee, James Hendler, Ora Lassila, et al. The semantic web. *Scientific american*, 284(5):28–37, 2001.
2. Ramanathan V Guha, Dan Brickley, and Steve Macbeth. Schema. org: Evolution of structured data on the web. *Communications of the ACM*, 59(2):44–51, 2016.
3. Wolfgang Halb, Yves Raimond, and Michael Hausenblas. Building linked data for both humans and machines. In *LDOW*, 2008.
4. Ben Adida, Mark Birbeck, Shane McCarron, and Steven Pemberton. Rdfa in xhtml: Syntax and processing. *Recommendation, W3C*, 7, 2008.
5. Rdfa core sequence. `https://www.w3.org/TR/rdfa-syntax/#s_sequence`. Accessed: 2017-02-20.
6. Brian Pinkerton. *Webcrawler: Finding what people want*. Citeseer, 2000.
7. Html document object manager node module. `https://www.npmjs.com/package/cheerio`. Accessed: 2017-02-20.
8. Rdf interface api node module. `https://www.npmjs.com/package/rdf`. Accessed: 2017-02-20.
9. Webcrawler node module. `https://www.npmjs.com/package/simplecrawler`. Accessed: 2017-02-20.
10. Rdfa test suite. `http://rdfa.info/test-suite/`. Accessed: 2017-02-20.
11. Rdfa webcrawler node module. `https://www.npmjs.com/package/rdfa-parser`. Accessed: 2017-02-20.
12. Http request client node module. `https://www.npmjs.com/package/request`. Accessed: 2017-02-20.