Hochschule für Technik, Wirtschaft und Kultur (HTWK) Leipzig
Fakultät für Informatik, Mathematik und Naturwissenschaften
Fachgebiet Informatik

Projektdokumentation Crossmedia Produktion

Extraktion von Statistikseiten für Eishockey

Von: Artem Pokas
Am: 29. Januar 2016
Studienrichtung: Informatik

Inhaltsverzeichnis

Einleitung

Als Projekt für das Modul Crossmedia Produktion habe ich mir die Extraktion von Statistikdaten aus einer dynamischen Webseite ausgesucht. Diese Daten werden daraufhin in Docbook-XML umgewandelt, um sie dann in alle Formate umwandeln zu können.

Die Extrahierung der Daten erfolgt mit CasperJS, welches auf dem Framework PhantomJS aufbaut. Durch die Kombination dieser Frameworks kann ein Webbrowser simuliert werden, was bei einer Webseite mit JavaScript und dynamischem Inhalt unumgänglich ist.

Im folgenden werde ich den genauen Ablauf beschreiben und auf die Frameworks eingehen.

1 Die Quelle der Daten

Die Daten werden aus einer Webseite [2] extrahiert. Diese Webseite wird kontinuierlich aktualisiert. Dabei ist der Inhalt höchst dynamisch dargestellt. Beispielsweise wird nur der Inhalt einer Tabelle bei einem Klick aktualisiert.
Dadurch kann ein einfaches Parsen der Webseite nicht stattfinden. Hierzu werde ich eine Möglichkeit beschrieben, was den Inhalt trotz des Hindernisses laden kann.
Der Urheber der Webseite ("FlashScore") hat unter anderem viele weitere Seiten eingestellt. Alle haben die selbige Struktur und können so, durch Anpassen der Links auf die neue Seite, ebenfalls bearbeitet werden.

Deshalb muss man sich beim Auswählen der Quelle für die Daten nicht nur auf Eishockey beschränken, sondern kann auch seinen Lieblingssport aussuchen.

2 PhantomJS

PhantomJS [4] ist ein WebKit, welches mittels einer JavaScript API gesteuert wird und so jeden Webbrowser simulieren kann.
Dadurch kann man auf den Quelltext der Webseite zugreifen, wie in einem richtigen Browser. Jedoch ist es nur die reine API für die Simulation des Browsers. Darauf aufbauend existieren viele Frameworks. Unter anderem das Framework bzw. Test-Framework CasperJS. Dieses wird später noch weiter erläutert.

Die wichtigsten Merkmale von PhantomJS:

- Auswahl des zu simulierenden Browsers
- Laden von Bildern der Webseite
- Skripte, die zum Parsen der Seite verwendet werden

3 CasperJS

CasperJS [1] ist ein Framework welches auf PhantomJS [4] aufbaut. Es bietet viele Funktionen um auf der Seite zu navigieren und alle Funktionen zu nutzen, die auch ein üblicher Nutzer einer Webseite nutzen könnte. Hierzu einige wichtige Merkmale:

- Navigieren innerhalb der Webseite
- Folgen von Links
- Erstellen von Screenshots
- Durchlaufen des DOM der Webseite
- Erstellung von eigenen Tests auf der Webseite

Für mein Projekt verwende ich nur einen Bruchteil der Funktionalität von CasperJS. Denn mit Hilfe von CasperJS kann noch viel mehr auf einer Webseite ausgeführt werden. Zum Beispiel kann ein Laufzeittest durchgeführt oder auch ein einfaches Ausfüllen eines Formulars simuliert werden.

Weitere Anwendungsbeispiele sind hierzu auf den jeweiligen Seiten von CasperJS [1] und PhantomJS[4] zu finden.

Beispiel zum Einstieg in ein CasperJS-Skript:

```
1 var casper = require('casper').create({
2   verbose: true,
3   logLevel: 'error',
4   pageSettings: {
5     loadImages: false,
6     loadPlugins: false,
7     userAgent: 'Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36
8           (KHTML, like Gecko) Chrome/29.0.1547.2 Safari/537.36'
9   },
10  clientScripts: ['/Users/Arti/Downloads/jquery-2.1.1.js']
11 });
```

4 Generierung der Daten

Das Script zum Extrahieren der Daten dieser Webseite [2] fängt immer auf der Startseite der jeweiligen Liga an. Daraufhin folgt es allen Links, die noch in dieser Liga spielen, also den Unterligen.

Dadurch das in jedem Land mindestens eine Liga vorhanden ist, gibt es diverse Startpunkte für das Skript. Um dies zu erleichtern, gibt es noch ein weiteres Skript, welches für jede Liga, die auf der Seite existiert, das eigentliche Skript aufruft und die generierten Daten auf eine Datei im selben Verzeichnis umleitet.

```
1 #!/bin/bash
2 casperjs $1 --site=http://www.ergebnisselive.com/eishockey/deutschland/del/
3       > tableDel.xml
4 echo "</table></section></article>" >> tableDel.xml
5 casperjs $1 --site=http://www.ergebnisselive.de/eishockey/finnland/liiga/
6       > tableLiiga.xml
7 echo "</table></section></article>" >> tableLiiga.xml
8 casperjs $1 --site=http://www.ergebnisselive.de/eishockey/osterreich/ebel/
9       > tableEbel.xml
10 echo "</table></section></article>" >> tableEbel.xml
11 casperjs $1 --site=http://www.ergebnisselive.de/eishockey/russland/khl/
12       > tableKhl.xml
13 echo "</table></section></article>" >> tableKhl.xml
14 casperjs $1 --site=http://www.ergebnisselive.de/eishockey/schweden/elitserien/
15       > tableElitserien.xml
16 echo "</table></section></article>" >> tableElitserien.xml
17 casperjs $1 --site=http://www.ergebnisselive.de/eishockey/schweiz/nla/
18       > tableNla.xml
19 echo "</table></section></article>" >> tableNla.xml
```

```
20 casperjs $1 --site=http://www.ergebnisselive.de/eishockey/tschechien/extraliga/
21       > tableExtraliga.xml
22 echo "</table></section></article>" >> tableExtraliga.xml
23 casperjs $1 --site=http://www.ergebnisselive.de/eishockey/usa/nhl/
24       > tableNhl.xml
25 echo "</table></section></article>" >> tableNhl.xml
```

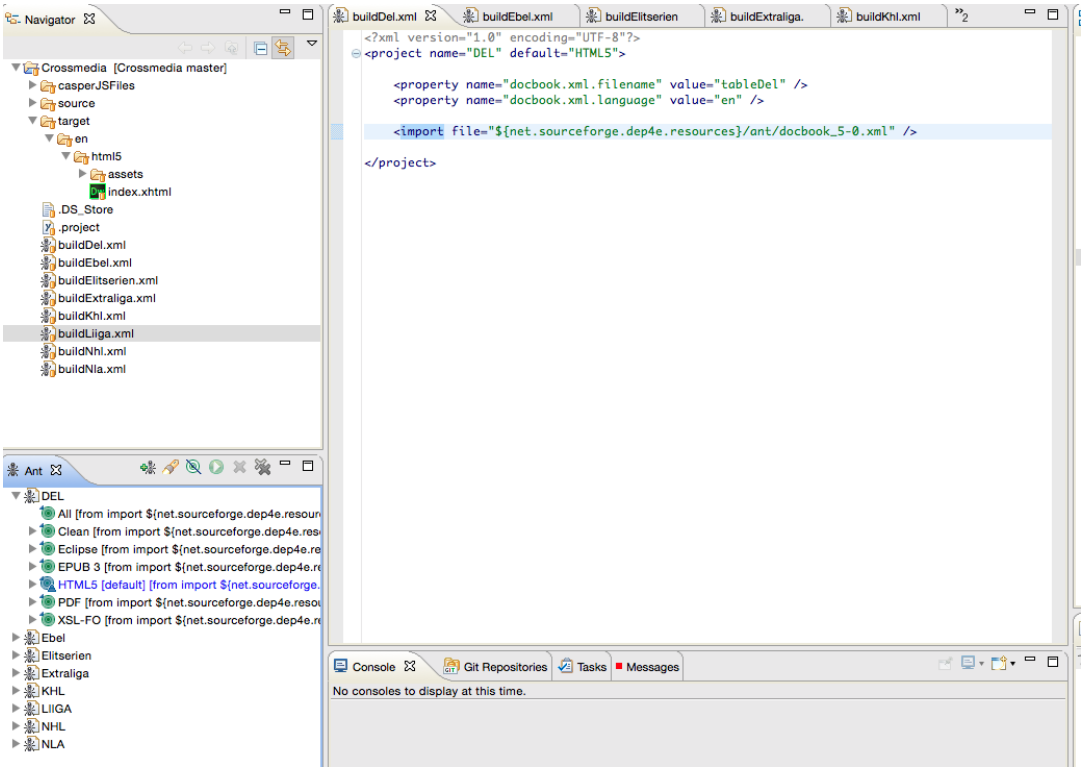
Als Parameter bekommt dieses Skript lediglich den Pfad zum CasperJS Skript, welches im Skript den Parameter $1 im obigen Skript ersetzt.

Da das CasperJS - Skript sehr umfangreich ist, wird dieses im Github [3] abgelegt und liegt auch diesem Dokument bei.

5 Generierung aus Docbook

Die Generierung kann mit jeder beliebigen Software wie z.B. OxyGen o.ä. erfolgen. In meinem Beispiel benutze ich das Dep4E-Plugin für Eclipse. Dadurch kann ein Projekt erzeugt werden, welches dann per XML-File konfiguriert werden kann und mit Hilfe von ANT das Docbook-XML in das gewünschte Format formatiert.
Alle Dateien, die vom CasperJS generiert werden, müssen in das Projekt kopiert werden. Hierzu ist in der Abbildung 'Dep4E Plugin' ein Überblick zu einem solchen Projekt im Eclipse.

Bild 5.1: Dep4E Plugin



6 Zusammenfassung

Insgesamt konnte ich die Extraktion zwar leider nicht komplett generisch gestalten. Jedoch kann das Skript auch auf jeder beliebigen Seite, die von FlashScore eingestellt ist, benutzt werden. Auch kann auf das Archiv zugegriffen werden, da dafür nur Link zur Webseite angepasst werden muss. Aus den extrahierten Daten kann auch im Weiteren ein Diagramm erzeugt werden, da sich doch rausstellte, dass in Tschechien, Russland der den USA doch wesentlich mehr Mannschaften spielen als in Deutschland oder anderen Ländern. Da aber die Daten bereits vorliegen ist das dann keine grosse Hürde mehr. Fast alles konnte automatisiert werden, bis auf das Kopieren der generierten Daten in das Eclipse-Projekt. Durch die Benutzung eines anderen Programmes kann dies aber auch automatisiert werden.

Das Vorgehen kann auch auf beliebige Seiten von FlashScore angewendet werden, da das Theme von deren Seiten ähnlich ist. Somit kann jeder seine Liebllingsseite und auch seinen gewünschten Sport aussuchen.

Da aus der erzeugten Tabelle ebenfalls ein Diagramm erzeugt werden könnte, kann dieses dann z.B. auch in einer eigenen Seite als interessantes Widget oder ähnliches eingebunden werden oder auch direkt in Fachzeitschriften, wenn sich diese z.B. auf die Vorjahre einer Mannschaft beziehen.

Das Verfahren und auch das Skript ist also, mit kleineren Anpassungen, in verschiedensten Richtungen einsetzbar und ebenfalls in verschiedenen Medien.

Literaturverzeichnis

[1] CasperJS. Software zum extrahieren. http://casperjs.org, 2016. [Online; accessed 29-Januar-2016]].

[2] FlashScore. Webseite mit statistikdaten. http://www.ergebnisselive.com/, 2016. [Online; accessed 29-Januar-2016].

[3] Github. Code für projekt. https://github.com/Arti09/Crossmedia, 2016. [Online; accessed 29-Januar-2016]].

[4] PhantomJS. Webkit. http://phantomjs.org, 2016. [Online; accessed 29-Januar-2016]].