Hochschule Hof

Wintersemester 2017/2018

Studienarbeit

in RESTful Webservices

Search++

von

Christian Deinlein, Thomas Herpich, Sebastian Pahl, Christian Pöhlmann, Tobias Wirth

Inhaltsverzeichnis

1 Architektur **4**
1.1 Package-Struktur 4
1.2 Weitere Verzeichnisse 5

2 Fremde Webservices **6**
2.1 Amazon 6
2.1.1 Verbindung 6
2.1.2 Erfahrungen und Probleme 7
2.2 Ebay 8
2.2.1 Verbindung 8
2.2.2 Erfahrungen und Probleme 9
2.3 Google 9

3 Bewertung **10**

4 Client **11**

1 Architektur

1.1 Package-Struktur

- searchpp
  Klasse Main - Start des Servers
- /database/
  Datenbankverbindung, Speichern und Laden der Produkte, User
- /localservice/
  PriceHistoryService, der im Hintergrund läuft und Preise für beobachtete Produkte lädt.
- /model
  Repräsentation der Produkte und User aus Datenbank und Webservices.
  - /config/
    Laden der Konfigurationseinstellungen
  - /json/
    Interfaces für die Konvertierung zu JSON
  - /products/
    Klassen für Produkte der verschiedenen Webservices und Bewertungen von Amazon.
  - /user/
    Klassen für die User
- /services/
  Anbindung der Webservices
- /sites
  REST-Schnittstellen des Servers
  - /products/
    Anfragen zum Suchen von Produkten.
  - /usr/
    Anfragen zum Anmelden von Benutzern und zum Anlegen bzw. Abfragen von beobachteten Produkten.

- /utils/
  Hilfsmethoden

1.2 Weitere Verzeichnisse

- /Client/
  Quellcode des Clients

- Hauptverzeichnis des Servers
  Konfigurationsdatei *searchpp.conf* mit Zugangsdaten der APIs.

2 Fremde Webservices

In diesem Kapitel wird die Verbindung zu fremden Webservices, Erfahrungen damit und eventuelle Probleme beschrieben.

2.1 Amazon

Von Amazon wurde die Amazon Product Advertising API verwendet. Diese API wurde ausgesucht, um Produktpreise, Produktinformationen und Bewertungen abzurufen. Leider mussten wir feststellen, dass Amazon die Produktbewertungen über die API nicht weitergibt. Genaueres dazu bei den Erfahrungen.

2.1.1 Verbindung

Um die API zu verwenden, wird ein Amazon Partnernet Account benötigt. Hat man dort ein Konto, kann man sich einen Access Key generieren lassen. Dieser wird für Anfragen an die API benötigt. Um diese erfolgreich zu stellen, muss man die Anfrage mit folgenden Schritten signieren:

- Zeitstempel an die URL anhängen
- Kodierung von ',' und ':' in RFC 3986 Spezifikation
- Parameter/Wert Paare am '&'-Zeichen Teilen
- Parameter/Wert Paare nach Byte-Wert sortieren
- Paare wieder mit '&' zusammenfügen
- 3 weitere Zeilen vor dem String anhängen
- Die Anfrage mit SHA256 über den AWS Secret Key verschlüsseln
- '+' und '=' Kodieren
- Berechneten Schlüssel als Signatur an Anfrage anhängen

Genaueres ist hier beschrieben.
Um einzelne Produkte über die ASIN (eindeutige ID für jedes Produkt auf Amazon) zu suchen, wurde in der Anfrage für den Parameter "Operation" der Wert "ItemLookup" und

für “IdType“ der Wert “ASIN“ angegeben. Für mehrere Produkte wurde für “Operation“ der Wert “ItemSearch“ verwendet. Der “SearchIndex“ legt fest, in welcher Kategorie die Produkte gesucht werden sollen. Mit der “ResponseGroup“ kann die Antwort von Amazon eingeschränkt werden, um uninteressante Werte auszulassen.
Für das Projekt waren folgende Werte von Amazon interessant:

- ASIN - Die ID des Produkts auf Amazon
- Title - Der Produkttitel
- Condition - Artikelzustand
- LowestNewPrice - Der günstigste Preis für ein neues Produkt
- SalesRank - Ähnlich wie Bestseller, Produkte mit geringen SalesRank wurden häufiger verkauft
- Manufacturer - Hersteller des Produkts
- Model - Modellbezeichnung des Produkts
- LargeImage - URL zu einem Produktfoto
- DetailPageURL - URL zum Produkt auf Amazon
- HasReviews - Gibt an, ob das Produkt bewertet wurde

2.1.2 Erfahrungen und Probleme

Wie bereits erwähnt, gibt Amazon keine Bewertung über seine API weiter. Dies war jedoch der Hauptgrund für uns, Amazon zu verwenden. Da es auch keine andere API gibt, die kostenlos Bewertungen oder Testberichte von Produkten liefert, haben wir beschlossen, die Bewertungen von der Website direkt auszulesen. Beim Abruf von Amazon Produkt Bewertungen kann es passieren, dass Amazon das System als Roboter erkennt und ein Captcha anzeigt. Mit vielen verschieden User-Agents wird versucht, diese zu umgehen und trotzdem an Daten zu kommen. Sollte dennoch nichts zurückkommen, wird das Produkt ignoriert, weshalb sich die Produktliste manchmal unterscheidet.

Ein weiteres Problem ist, dass Amazon bei mehreren, aufeinanderfolgenden Anfragen manchmal den Fehler 503 mit der Information “You are submitting requests too quickly. Please retry your requests at a slower rate.“ zurückliefert. Diese tritt vor allem bei der Suche nach mehreren Produkten auf. Mit einer Anfrage bekommt man nur 10 Ergebnisse geliefert, man kann aber pro Anfrage eine Seitenzahl angeben. Diese kann bei der Suche in allen Kategorien maximal 5 sein. Um also 50 Ergebnisse zu erhalten, müssen 5 Abfragen nacheinander erfolgen. Dabei bremst Amazon, durch den Fehler, die Anfragen teilweise stark aus.

Die Dokumentation auf der Website könnte von der Navigation und Gliederung auch besser strukturiert sein. An manchen Stellen werden Beispiele beschrieben, die entweder nicht mehr funktionieren oder deren Funktion sonst nicht beschrieben wird.

2.2 Ebay

Von Ebay wurde die Finding API verwendet. Die API wurde ausgesucht, um weitere Preise und Angebote zu Produkten zu finden, die nach dem Algorithmus als "empfohlen" eingestuft wurden. Dazu zählen auch B-Ware und gebrauchte Produkte (sofern der Benutzer das möchte), die entweder als Auktion oder Sofortkauf angeboten werden.

2.2.1 Verbindung

Zur Verwendung der API wird ebenfalls ein Key benötigt. Dafür kann man sich einfach ein neues Ebay Konto anlegen, oder ein bestehendes verwenden. Mit einem Konto kann man sich dann Keys für den Sandbox Modus oder den Production Modus generieren lassen. Der Sandbox Modus ermöglicht es dem Benutzer, mit einem virtuellen Kapital virtuelle Produkte zu kaufen. Dieser wurde von uns aber nicht benutzt. Für eine erfolgreiche Anfrage muss lediglich die App-ID, die mit dem Key generiert wurde, angegeben werden.

Leider konnten die Anfragen für eine Liste von Produkten und die Anfrage für ein einzelnes Produkt nicht über die selbe API erreicht werden. Beide API liefern jedoch ähnliche Antworten, die sich eigentlich nur durch Groß-/Kleinschreibung der Attribute unterscheiden.
Folgende Informationen waren für das Projekt von Interesse:

- ItemId - Die ID des Produkts auf Ebay
- Title - Der Titel des Angebots
- Condition - Der Artikelzustand wird als ID geliefert, die viele Werte annehmen kann (siehe Enumeration Condition in searchpp.model.products)
- CurrentPrice - Der aktuelle Preis des Artikels
- ListingTyp - Art des Angbots (siehe Enumeration ListingType in searchpp.model.products)
- GalleryURL - URL zu einem Produktfoto
- ViewItemURL - URL zur Produktseite auf Ebay

Wichtig ist es bei den Anfragen noch die Global-ID bzw. Site-ID festzulegen. Letztere gibt an, ob die Produkte z.B. auf ebay.de oder auf ebay.com gesucht werden.

2.2.2 Erfahrungen und Probleme

Die Verwendung von Ebay ist sehr einfach, vor allem im Vergleich mit Amazon. Man kann direkt ein Konto anlegen, ohne spezielle Anforderungen zu erfüllen oder sich zunächst zu bewerben. Die Anfragen müssen ebenfalls nicht bearbeitet werden und können so direkt von Browser aus gestellt und getestet werden. Die Anzahl der Anfragen ist unbegrenzt.
Bei der Dokumentation sind keine Fehler aufgefallen und es wurden viele Beispiele eingearbeitet.

2.3 Google

Bei Google wurde die OAuth2- (Login) und die Calendar-API verwendet. Es gibt für die Schnittstellen Implementierungen in verschiedenen Programmiersprachen, u. a. in Java. Allerdings sind die Java-Implementierung umständlich zu verwenden, deshalb wurden die APIs direkt per REST-Schnittstelle angesprochen. Dies ist im Rahmen einer Studienarbeit vertretbar, sollte aber für ein ëchtesPProjekt vermieten werden, wegen Fehler und Sicherheit.
Die REST-Schnittstelle ist einfach zu verstehen und anzuwenden.
Vorgehen Allgemein:

- Client-ID und Secret-Key erzeugen über Google-Konto
- RedirectUrls eintragen

Vorgehen Login:

- Client-ID und Scope (hier z.B. calendar und openid) mitgeben
- Benutzer wird aufgefordert, sich einzuloggen und den Zugriff, beschränkt auf Scope, zuzulassen.
- Danach bekommt man einen Code, mit dem man einen Access Token beantragen kann.
- Hat man einen Account Token, kann man die Informationen des Benutzer abfragen.
- Danach ist der Benutzer angemeldet.

3 Bewertung

Für den einfacheren Vergleich von Produkten wird die Bewertung, durchschnittliche Bewertung und Anzahl an Bewertungen in einen einzigen Wert umgerechnet. Dazu wurde folgende Funktion entworfen:

$$y = z + z * (z * a * (x - b)^3)$$

- y - Produktbewertung
- z - Anzahl aller Bewertungen
- x - Durchschnittliche Bewertung
- a - Faktor <= 1, um den Bereich mit wenigen Bewertungen abzuflachen. Wird benötigt, wenn das Ergebnis gerundet wird. (Empfehlung: 0.25)
- b - Beschreibt die Durchschnittsbewertung, welche neutral bewertet werden soll (Empfehlung: 3)

a und b sind Erfahrungswerte und müssen gegebenenfalls angepasst werden.

4 Client

Für das Design des Clients wurde Bootstrap, für die Webanwendung das Framework AngularJS benutzt. Der Client wurde mithilfe von AngularJS bzw. ngRoute als Singlepage-Application aufgebaut. Dabei besitzt jede View einen eigenen Controller, der bei der Navigation aktiviert wird. Im Controller werden je nach Seite Daten geladen oder nach Benutzereingabe abgerufen. Seitenübergreifende Daten werden nur im $rootScope gespeichert, weshalb ein neu Laden der Seite alle Informationen löscht. Wenn sich ein Benutzer über Google anmeldet, wird er wieder auf die Startseite weitergeleitet und der Token in der URL übergeben. Dieser wird dann im $rootScope gespeichert und an die Anfragen angehängt.