Christoph Lüth
Bernd Gersdorf
Christian Maeder
René Wagner
Dennis Walter
Diedrich Wolter
Simon Ulbricht

9. Übungsblatt

Ausgabe: 20.01.11
Abgabe: 31.01.11

9.1 *Persistenz* *8 Punkte*

Schreiben Sie ein Programm, das bei jedem Aufruf seine Kommandozeilenargumente zusammen mit dem Aufrufzeitpunkt in einer Textdatei ablegt. Nur wenn das Argument `show` allein übergeben wird, soll stattdessen die Liste aller bisher gespeicherten Eingaben ausgegeben werden. Das verwendete Datumsformat (hier: RFC 822) kann beliebig gewählt werden, sollte aber mindestens minutengenau sein.

Beispiel:

```
dw$ ./Blatt09a 1   2   3  4
dw$ ./Blatt09a the show must go on
dw$ ./Blatt09a rossbratwurst
dw$ ./Blatt09a show
Wed, 19 Jan 2011 11:24:43 +0100> 1 2 3 4
Wed, 19 Jan 2011 11:26:17 +0100> the show must go on
Wed, 19 Jan 2011 11:29:00 +0100> rossbratwurst
dw$
```

Hinweis: die einfachste Art der Persistierung gelingt über eine Umwandlung der Werte in Strings und entsprechendes Zurücklesen, was durch die Verwendung von `read` und `show` sehr einfach möglich ist. Dies funktioniert (vereinfachend gesagt) für solche Typen, die Instanz der Klassen `Read` und `Show` sind. Üblicherweise gilt für diese nämlich: `read (show x) == x`.

Um erfolgreich per `ghc --make <Datei>` kompiliert und gelinkt zu werden, muss Ihr Modul `Main` heißen (der Dateiname darf hiervon abweichen).

Folgende Funktionen sind hilfreich: `System.getArgs`, `Directory.doesFileExist`, `readFile`, `writeFile`, `read`, `Data.Time.Clock.getCurrentTime`, `Data.Time.Format.formatTime`,

9.2 *Volatilität* *12+2 Punkte*

Interessanter als die eigene Kommandozeilengeschichte ist das wirklich wahre Leben, das sich in den RSS-Newsfeeds der Nachrichtenseiten dieser Welt widerspiegelt. Das Ziel in dieser Aufgabe ist, den (Schlag-)Wörtern in den Titeln von Newseinträgen die Daten zuzuordnen, an denen diese auftraten und somit aktuell waren. Auf Basis dieser Informationen arbeitet dann ein Hype-O-Meter z.B. am Ende des Jahres zahlenmäßig auf, wann uns welche Themen bewegt haben.

1. Implementieren Sie eine Funktion zum Anfordern einer XML-Datei per HTTP-Request. Orientieren Sie sich hierbei an `httpExample` in der Vorlage.

   `getURL :: String → IO (Maybe Element)`

   `Element` ist der Datentyp zur Repräsentation von XML-Dokumenten. Funktionen zum Auffinden von Unterelementen und Attributen finden sich im Modul `Text.XML.Light.Proc`, etwa `strContent`, `filterChildName` oder `filterElementsName`; ein Beispiel ist in `xmlExample` in der Vorlage angegeben.

2. Entwickeln Sie eine Funktion zum Extrahieren der für die Schlagwortzuordnung relevanten Informationen (d.h. Titel und Datum der Veröffentlichung) aus *RSS 2.0* Feeds:

   `rss2ExtractItems :: Element → UTCTime → [(String, UTCTime)]`

   Die Angabe des Veröffentlichungsdatums in Feeds ist optional; verwenden Sie das aktuelle Datum (2. Parameter) für Einträge, die kein eigenes Datum enthalten.

3. Stellen Sie analog zu 9.1 Funktionen zur Persistierung der Schlagwortdatenbank bereit, die folgenden Typ haben könnte:

   ```
   type Keywords = Data.Map.Map String (Data.Set.Set UTCTime)
   ```

4. Implementieren Sie nun den Ablauf der Aktualisierung der Schlagwortdatenbank:

   - An der Kommandozeile werden beliebig viele URLs von Newsfeeds angegeben.
   - Das Program lädt diese Feeds und extrahiert Titel und Daten.
   - Die Schlagwortdatenbank wird aktualisiert, d.h. für bestehende Schlagwörter werden die neuen Daten hinzugefügt und neue Schlagwörter werden eingefügt.
     Wie Sie den Titel in eine Liste von Schlagwörtern zerlegen, bleibt Ihnen überlassen. Für clevere Filter (die mindestens Füllwörter wie *und*, *oder*, etc. entfernen sollten) sind bis zu 2 Bonuspunkte erhältlich. Ein einfacher Ansatz, der schlicht zusammenhängende Buchstaben als Schlagwörter ansieht, reicht aus, etwa

     ```
     split :: String → [String]
     split "Quadratur des Kreises, Teil II" ==
       ["quadratur", "des", "kreises", "teil", "ii"]
     ```

   - Das Programm gibt an der Kommandozeile das aktuelle Datum und die aktualisierte Anzahl der verwalteten Schlagwörter aus.

5. Die Eingabe des Kommandozeilenarguments `top10` soll zur Auswertung der gesammelten Daten führen. Es müssen zwei Datumsangaben (im ISO-Format `%Y-%m-%d`, z.B. 2011-01-01) folgen, die den Starttag (inklusive) und den Endtag (exklusive) des auszuwertenden Intervalls bezeichnen. Es soll die Liste der zehn am häufigsten vorkommenden Schlagwörter in absteigend sortierter Reihenfolge ausgegeben werden.

   Beispiel:

   ```
   dw$ ./Blatt09b top10 2010-01-01 2011-01-01
   1, facebook, 100
   2, youtube, 75
   3, google, 40
   4, ebay, 40
   5, wetter, 35
   6, you, 30
   7, gmx, 25
   8, test, 25
   9, web.de, 25
   10, yahoo, 20
   ```

   (Quelle: Google Insights for Search, Web-Suche Interesse, Deutschland, 2010)

6. Sammeln Sie News der in der Vorlage angegebenen Feeds, werten Sie die gesammelten Daten (per `top10`) aus und dokumentieren Sie das Ergebnis.

Nützliche Funktionen sind:
`Network.HTTP.{simpleHTTP, getRequest, getResponseBody}` sowie
`Text.XML.Light.{Element, filterElementsName, qName, unqual, parseXMLDoc, ...}`.

Die leichtgewichtige XML-Bücherei `Text.XML.Light` kann ggf. per `cabal install xml` nachinstalliert werden.

Gutes Gelingen!