Dokumentation Html Parser
Autor: Moritz Engelmann
Projekt: swt13-wb

Als Grundlage wird die URL Liste der Personenseiten genommen, die im Moment im HTML-Format gespeichert sind und die letztendlich anhand eines Python-Skripts in .pickle Dateien umgewandelt werden.
Die gewonnen Daten werden dann speziell formatiert und in ein Dictionary umgewandelt, um sie später leichter weiterverarbeiten zu können.

PersonDictionaryParser.py

Dieses Python-Skript kümmert sich um das Parsen, Formatieren und Abspeichern der html-Daten. Dabei ist es in mehrere Klassen unterteilt.

PersonDataParser: Die ist der eigentlich html-Parser, welcher den in Python standardmäßig eingebauten Html-Parser benutzt. Er findet haupsächlich über die Tags `<td>` und `</td>` die Stellen, an denen Daten stehen die für unser Projekt wichtig sind. So findet er in einem ersten Schritt die Bezeichnung der einzelnen Abschnitte (z.B A, B, E, M, GPV), sofern eine Bezeichnung oder gar der Abschnitt selber existiert. Die Bezeichnung eines solchen Abschnittes befindet sich innerhalb von `<td>` und `</td>`, die Daten, welche zu diesem Abschnitt gehören wiederum auch. Um einzelne Abschnitte leichter zu finden wird auch nach dem Attribut `class` gesucht, welches dann je nach Abschnitt verschiedene Werte haben kann, wie z.B. `t5s`, `t100g`, `t100s` oder `t100n`.
Einzelne Abschnitte haben aber gleiche Werte, so haben z.B. Geburtsdaten und Sterbedaten beide `class=t100n`. Daher wird hier auf die ersten Zeichen der Daten geachtet, um die richtige Zuordnung der Daten zu den Abschnitten zu gewährleisten. So beginnen die Geburtsdaten immer mit einem `*` und die Sterbedaten immer mit einem `†`.
Aus den gesammelten Daten entsteht während das Parsens ein String, wobei einzelne Abschnitte mit einer nicht im Text vorkommenden Zeichenkette getrennt werden, in diesem Fall ist `.:._.:.` verwenden worden.

DataFormater: Diese Klasse ist für die Formatierung der Daten verantwortlich. Sie zerteilt zum einen den geparsten String und erstellt daraus ein Dictionary in der Form:

```
{Abschnitt 1: Daten von Abschnitt 1, Abschnitt 2: Daten von Abschnitt 2, …}
```

Oder an einem Beispiel:

```
{'Name': 'Mueller, Peter', 'Geburtsdaten': '*12.6.1966, Dresden', …}
```

In weiteren Methoden werden nun einzelne Abschnitte genauer formatiert. Dies sei hier am Beispiel der Sterbedaten gezeigt. Zuerst wird geschaut, ob in den Sterbedaten der Begriff Grabstätte auftaucht. Ist dies der Fall, wird der String an Grabstätte in zwei Elemente eines Feldes geteilt. Dabei wird der Begriff Grabstätte aus dem String gelöscht. Danache wird ein neues Dictionary erstellt, das die Form

```
{'Grabstaette': Daten zur Grabstaette}
```

hat. Dieses wird dann dem Dictionary für die Sterbedaten hinzugefügt, so das man dann ein Dictionary in der Form

```
{'Sterbedaten': {'Grabstaette': Daten zur Grabstaette}}
```

erhält. Das nun verbleibende erste Listenelement kann nun noch das Sterbedatum und den Sterbeort

enthalten. Zuerst wird das * am Anfang entfernt, da es für die Daten nicht mehr wichtig ist. Falls dieses ein oder mehrere Kommas enthält, wird es dort in einen neue Liste aufgespalten. Die neue Liste ist nun ein, zwei oder mehr Elemente lang. Dabei ist das erste Element das Todesdatum. Sofern ein zweites Element vorhanden ist, so ist dies der Todesort. Bei mehr als zwei Elementen werden alle zusätzlichen Elemente dem Todesort hinzugefügt.
Am Ende ensteht folgendes Dictionary:

```
{'Sterbedaten': {'Grabstaette': Daten zur Grabstaette (falls vorhanden), 'Sterbedatum': Daten zum Sterbedatum, 'Sterbeort': Daten zum Sterbeort (falls vorhanden)}}
```

Andere Abschnitte werden ebenfalls gesondert formatiert. So enthält Q z.B. Überschriften und Fußnoten, welche gesondert behandelt werden müssen. Auch beginnen die Daten bei einzelnen Abschnitten mit • und da muss dieser entfernt werden.
Für ein Beispiel des fertigen Dictionarys, das Python-Skript showPickle.py benutzen.

MakePickleFromData: Diese Klasse liest die Liste der Html Dateien aus einer .pickle Datei, lässt den Parser über alle laufen und speichert die gewonnen Dictionarys in einem Unterordner .pickle. Dabei können die Daten direkt aus dem Web abgefragt werden oder wenn sie sich in einem Unterordner html befinden, von der lokalen Festplatte gelesen werden.
Außerdem enthält dies Klasse die Möglichkeit die Portrait Bilder mit herunterzuladen. Sie werden dabei ein den Unterordner Portraits gespeichert.
In der Main Methode kann über zwei boolean Parameter beim Aufruf der Methode getData() bestimmt werden, ob lokale Dateien oder Daten aus dem Web verwendet werden sollen und ob die Portrait Bilde heruntergeladen werden sollen.

dictHelper.py

Dieses Python-Skript enthält einige Hilfsmethoden zum öffnen von und zum speichern von Dictionarys in .pickle-Dateien. Eine Methode zum speichern der Portrait Bilder ist ebenso enthalten , wie eine Methode um html Zeichen und Unicode umzuwandeln.

downloadHTML.py

Dieses Python-Skrip dient dazu, die Html-Daten aus dem Web auf den lokalen Rechner zu laden, um die Daten schneller zu verarbeiten, wenn man z.B. die anderen Klassen testen möchte. Es erstellt auch eine .pickle Datei, welche die lokalen Html-Daten als Liste enthält, um sie leichter im Parser einzulesen.

showPickle.py

Mit diesem Python-Skript kann man sich einzelne oder anhand der ID bestimmte fertig .pickle Dateien mit Dictionarys zu den einzelnen Personen anzeigen lassen. Es dient hauptsächlich zum Testen und Finden von Fehlern.

listOfAllHTMLSites.pickle

Enthält eine Liste mit den Urls aller Html-Dateien im Web.