DataCraft

Datenschnittstellen

3 Webscraping

DataCraft

Webscraping bezeichnet das automatisierte Auslesen von Informationen aus Webseiten. Das ist z.B. dann nötig, wenn keine API zur Verfügung steht.

Webscraping besteht aus 3 Stufen:

- Aufruf der Webseite
- Extraktion der Daten
- Speichern der Daten

Webcrawler verwenden die auf der Seite gefundenen Links, um zur nächsten Seite zu gelangen. Webcrawler werden meist von Suchmaschinen genutzt.

- Viele größere Webseiten versuchen durch Verschleierung und Verschlüsselung das Webscraping zu erschweren. So können z.B. Email-Adressen nicht im Klartext, sondern über ein Javascript-Codeabschnitt eingefügt werden, um den Zugriff zu erschweren.

- Auch muss die Legalität (Urheberrecht) beachtet werden. Einige Anbieter verbieten das automatische Auslesen von Daten in ihren AGBs.

- Auf der anderen Seite kann es aber auch Vorteile haben (Auflistung in Suchmaschinen, Preisvergleichen, …).

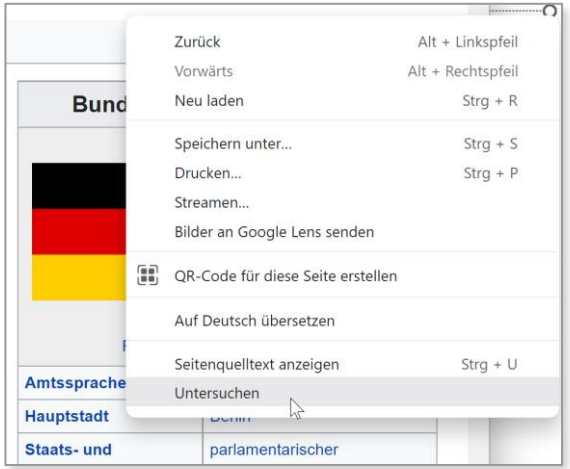
- Solange die Anfragen selten sind und wenig Traffic generieren, ist es normalerweise kein Problem. Bots werden meist anhand ihrer Geschwindigkeit erkannt

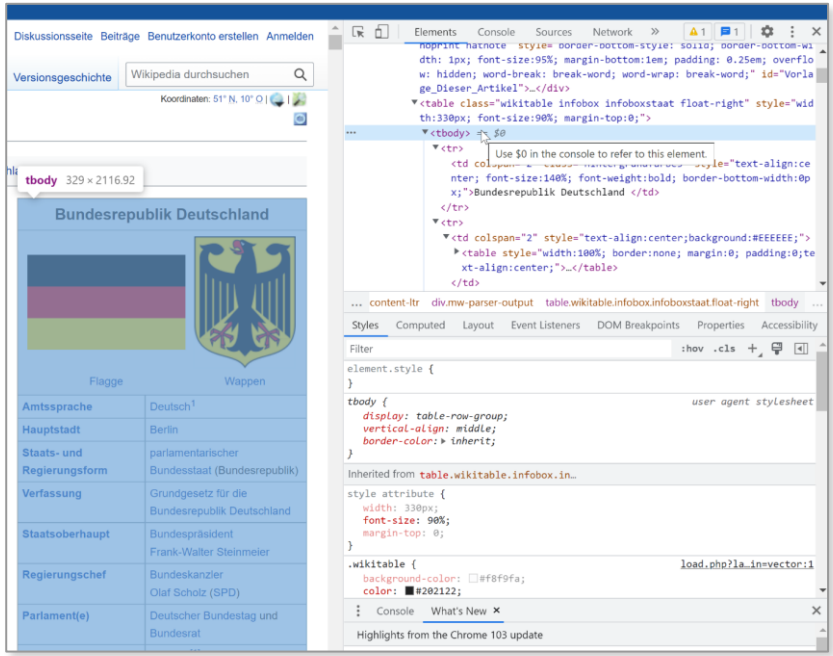
- **Requests**, um Daten per HTTP-Verben (GET, POST) abzufragen.
- **Beautiful Soup** ist die am häufigsten verwendete Bibliothek, um Daten aus XML und HTML zu extrahieren
- **Scrapy** ist ein komplettes Framework, bestehend aus mehreren Bibliotheken, um Webscraper zu bauen. Damit sind die Einstiegshürden etwas höher.
- **Selenium** bietet die Möglichkeit der Browser-Fernsteuerung aus Python heraus

Untersuchung der Webseite mittels Browser

DataCraft

Chrome: Bewege die Maus auf ein Element, rechte Maustaste, Untersuchen





Installation von Beautiful Soup mittels conda install beautifulsoup4

Das Auslesen von (statischen) Webseiten ist recht einfach:

```
import requests
from bs4 import BeautifulSoup

r = requests.get("https://de.wikipedia.org/wiki/Deutschland")

soup = BeautifulSoup(r.content, 'html.parser')
print(soup.prettify)
```

Beautiful Soup: Suche

Beautiful Soup bietet viele Möglichkeiten, Elemente in einer Seite zu finden. Die wichtigsten Funktionen sind **find()** und **find_all**, um nach HTML-Elementen zu suchen

```
print(soup.title)
print(soup.title.string)

print(soup.find("h1").string)

ueberschriften = soup.find_all("h2")
print([u.string for u in ueberschriften])

links = soup.find_all("a")
print([l.get("href") for l in links])
```

Beautiful Soup: fortgeschrittene Suche

DataCraft

- Suche nach mehreren Elementen

```
soup.find_all(name=["h1","h2"])
```

- Es kann auch nach regulären Ausdrücken gesucht werden

```
import re
regex = re.compile("^h[1-6]")
soup.find_all(name=regex)
```

- Filterung mit eigener Funktion

```
def has_class_no_id(tag):
        return tag.has_attr("class") and not tag.has_attr("id")

soup.find_all(has_class_no_id)
```

Beautiful Soup: fortgeschrittene Suche

DataCraft

- Suche nach Keywords

```
soup.find_all(name="table", class_="infotable")
```

- auch über dictionaries

```
name_soup = BeautifulSoup('<input name="email"/>')
name_soup.find_all(attrs={"name": "email"})
```

- Suche nach Inhalten. Es können Strings, reguläre Ausdrücke, Liste oder Funktionen übergeben werden

```
# exakte Übereinstimmung
soup.find_all(string="Hauptstadt")
# enthält den Teilstring
soup.find_all(string=re.compile("Hauptstadt"))
```

Beautiful Soup: Navigation

HTML-Seiten sind hierarchisch aufgebaut.

- Navigation über Verkettung

  ```
  soup.body.a
  ```

- und/oder über children und parent

  ```
  soup.find(name="table").children
  soup.find(name="td").parent.parent
  ```

mehr in der offiziellen Dokumentation von Beautiful Soup