Data science

Web scraping in Python

2025-2026

Web scraping

- Web scraping is the process of scraping information from websites
- You convert the extracted data into a readable format that a programming language can easily work with.
- This sounds rather silly, as you're creating a database from a website that gets its content from an existing database
  - But if the creator of the database doesn't give you access to the original database, you have to reverse-engineer it
- It's a bit of a cat chasing her tail:
  - a scrape only works as long as the website isn't changed

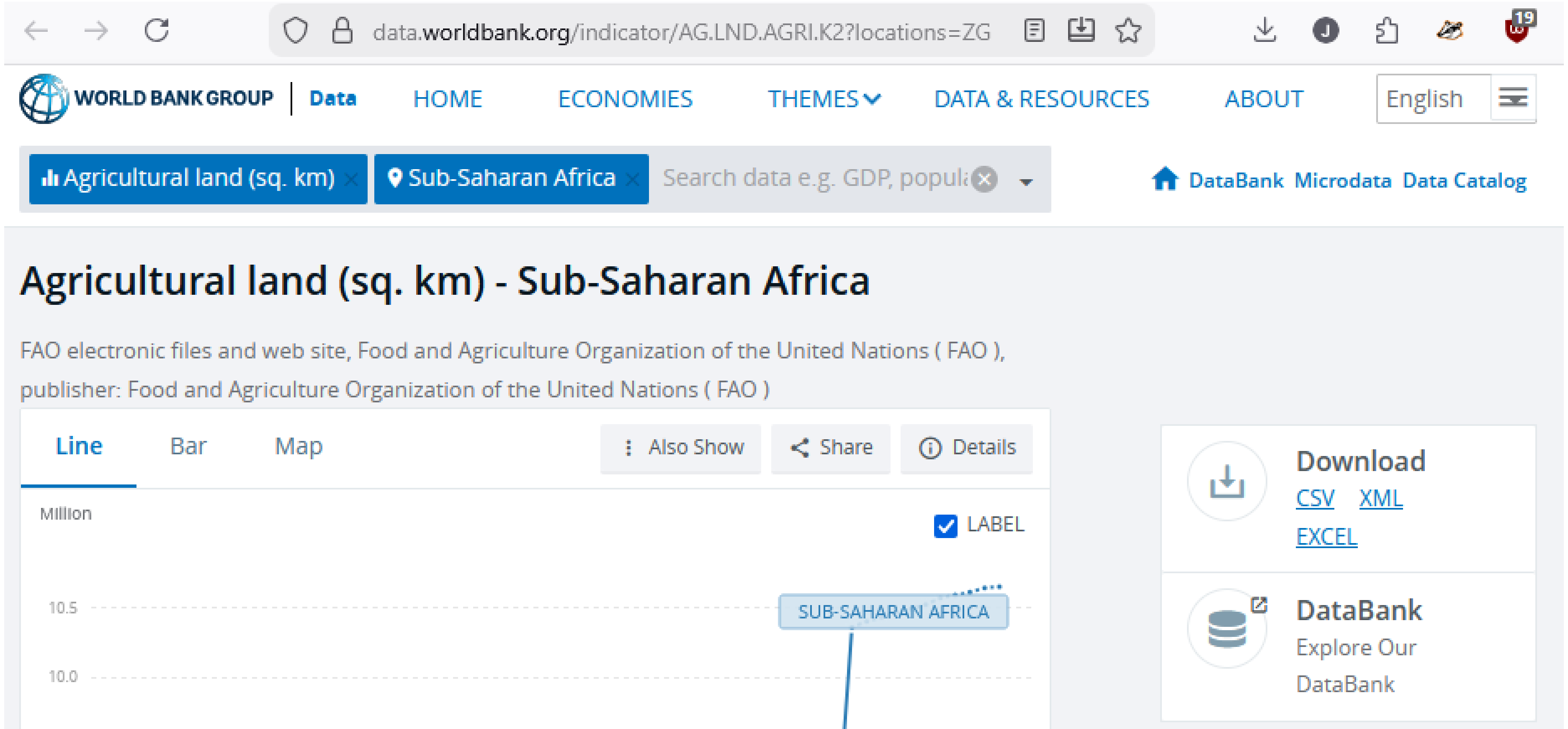
Alternative to scraping: APIs

- Some websites offer **A**pplication **P**rogrammable **I**nterfaces that allow you to access their data using a format like JSON.

- Using an API is more stable and reliable because when the front-end of a website changes, it affects your scraping code while the back-end API structure usually remains unchanged.

Alternative to scraping: Open data

- A lot of data is also made available in CSV-format
- This can be handled by Python very well

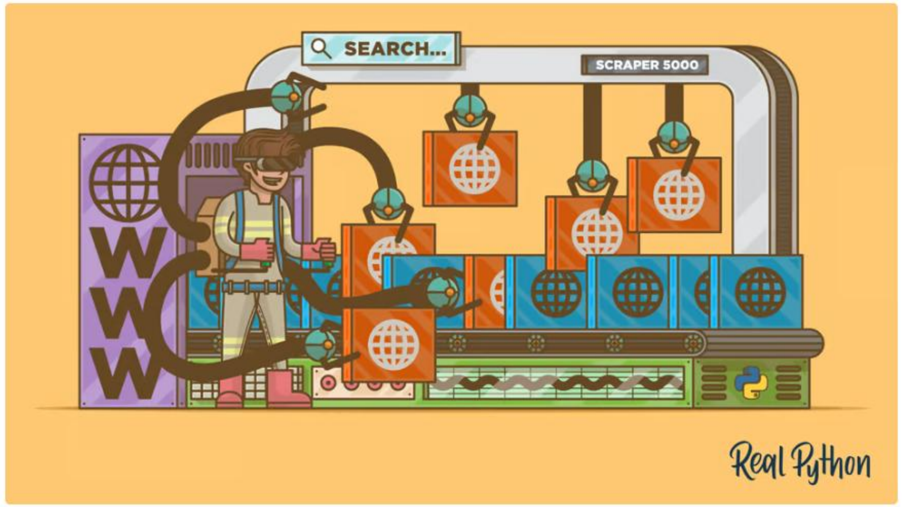


Agricultural land (sq. km) - Sub-Saharan Africa

FAO electronic files and web site, Food and Agriculture Organization of the United Nations (FAO), publisher: Food and Agriculture Organization of the United Nations (FAO)

Web scraping demo from course on Real Python

https://realpython.com/beautiful-soup-web-scraper-python/

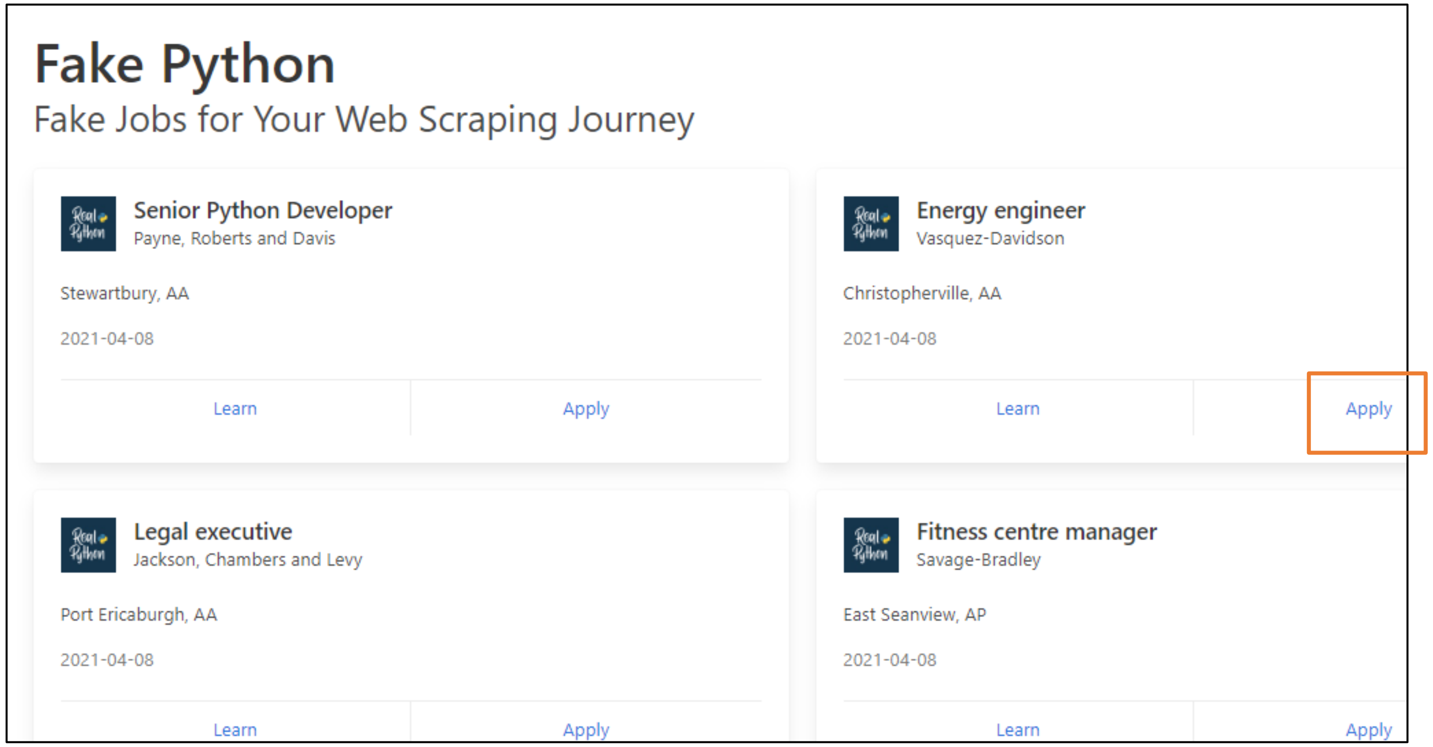


Beautiful Soup: Build a Web Scraper With Python

Web scraping demo from course on Real Python

- Goal: scrape the data from a fake jobs advertising site

  https://realpython.github.io/fake-jobs/



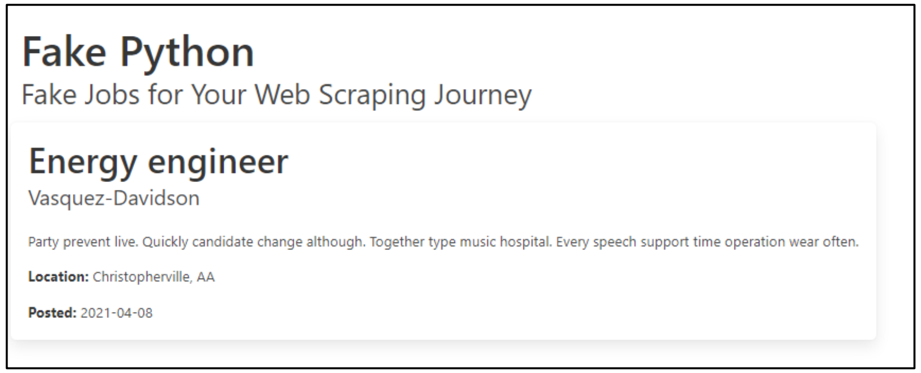
Fake Python
Fake Jobs for Your Web Scraping Journey

Senior Python Developer
Payne, Roberts and Davis
Stewartbury, AA
2021-04-08
Learn | Apply

Energy engineer
Vasquez-Davidson
Christopherville, AA
2021-04-08
Learn | Apply

Legal executive
Jackson, Chambers and Levy
Port Ericaburgh, AA
2021-04-08
Learn | Apply

Fitness centre manager
Savage-Bradley
East Seanview, AP
2021-04-08
Learn | Apply



Fake Python
Fake Jobs for Your Web Scraping Journey

Energy engineer
Vasquez-Davidson

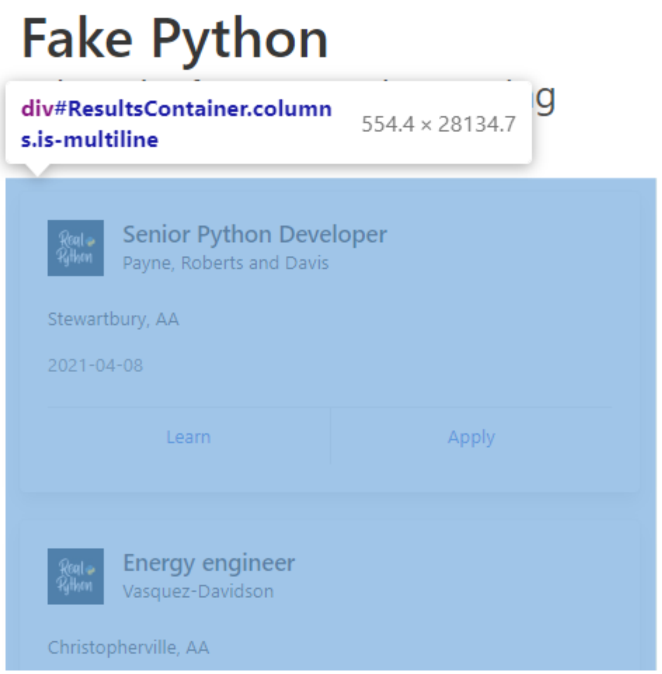
Party prevent live. Quickly candidate change although. Together type music hospital. Every speech support time operation wear often.

Location: Christopherville, AA

Posted: 2021-04-08

Web scraping demo from course on Real Python

- Step 1:
  - Inspect the site’s url(s) and used query parameters
  - Inspect the site’s DOM (document object model)

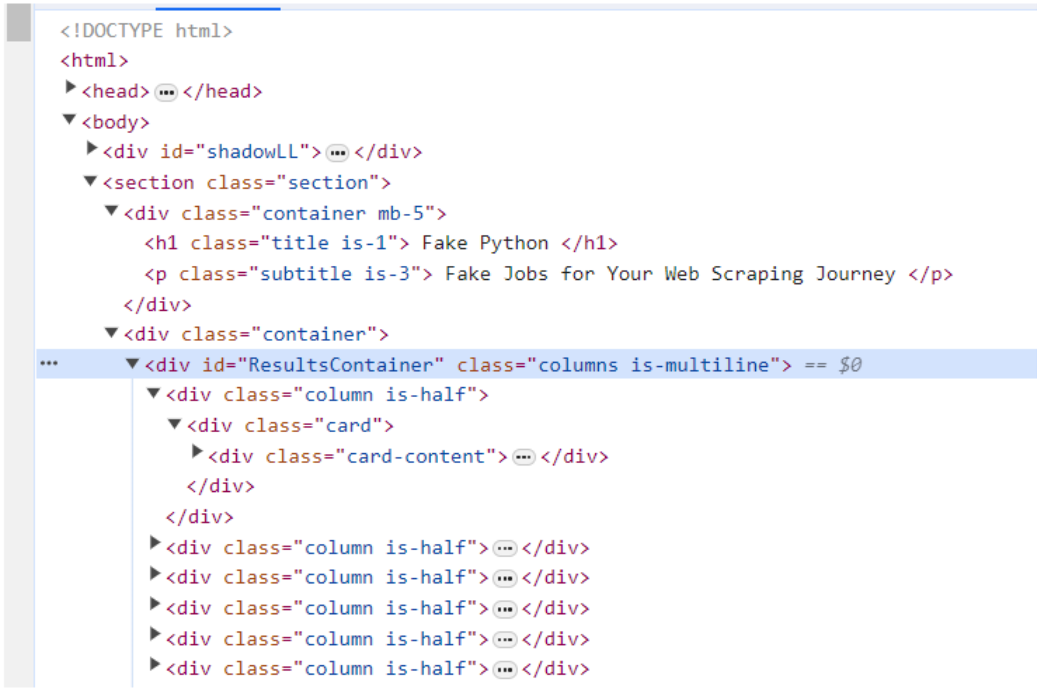


Fake Python

div#ResultsContainer.columns.is-multiline 554.4 × 28134.7

Senior Python Developer
Payne, Roberts and Davis
Stewartbury, AA
2021-04-08
Learn Apply

Energy engineer
Vasquez-Davidson
Christopherville, AA



```
<!DOCTYPE html>
<html>
 ▶<head>…</head>
 ▼<body>
   ▶<div id="shadowLL">…</div>
   ▼<section class="section">
     ▼<div class="container mb-5">
        <h1 class="title is-1"> Fake Python </h1>
        <p class="subtitle is-3"> Fake Jobs for Your Web Scraping Journey </p>
      </div>
     ▼<div class="container">
…      ▼<div id="ResultsContainer" class="columns is-multiline"> == $0
         ▼<div class="column is-half">
           ▼<div class="card">
             ▶<div class="card-content">…</div>
            </div>
          </div>
         ▶<div class="column is-half">…</div>
         ▶<div class="column is-half">…</div>
         ▶<div class="column is-half">…</div>
         ▶<div class="column is-half">…</div>
         ▶<div class="column is-half">…</div>
```

Web scraping demo from course on Real Python

- Step 2:
  - Load the returned HTML code from a specific url into your script using Python’s **requests** library.
  - Install the library in your venv first: `pip install requests`

```
# demo scraping static site
import requests

URL = "https://realpython.github.io/fake-jobs/"

# execute HTTP GET request to retrieve the sent HMTL data
page = requests.get(URL)

print(page.text)
```

Web scraping demo from course on Real Python

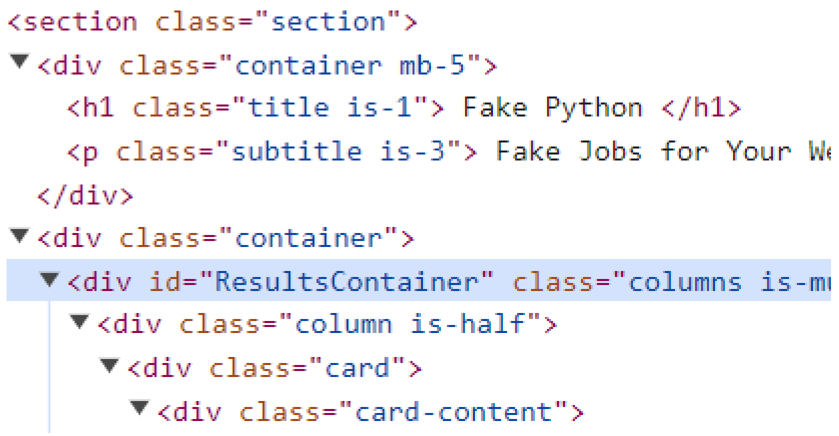
- Step 3:
  - Filter the response from step 2 to the data you need using the Python library Beautiful Soup.
  - Install the library in your venv first: `pip install beautifulsoup4`
  - Parse the response into html-format

```
import requests
from bs4 import BeautifulSoup

URL = "https://realpython.github.io/fake-jobs/"
page = requests.get(URL)

# parse html via bs4
# use page.content instead of page.text to avoid character encoding issues
soup = BeautifulSoup(page.content, "html.parser")
```

Web scraping demo from course on Real Python

- Step 3:
  - **Find** the element you need by using the assigned id attribute
  - Result is a filtered part of the entire html soup

```python
soup = BeautifulSoup(page.content, "html.parser")

# prettify() pretty prints the html
results = soup.find(id="ResultsContainer")
print(results.prettify())
```

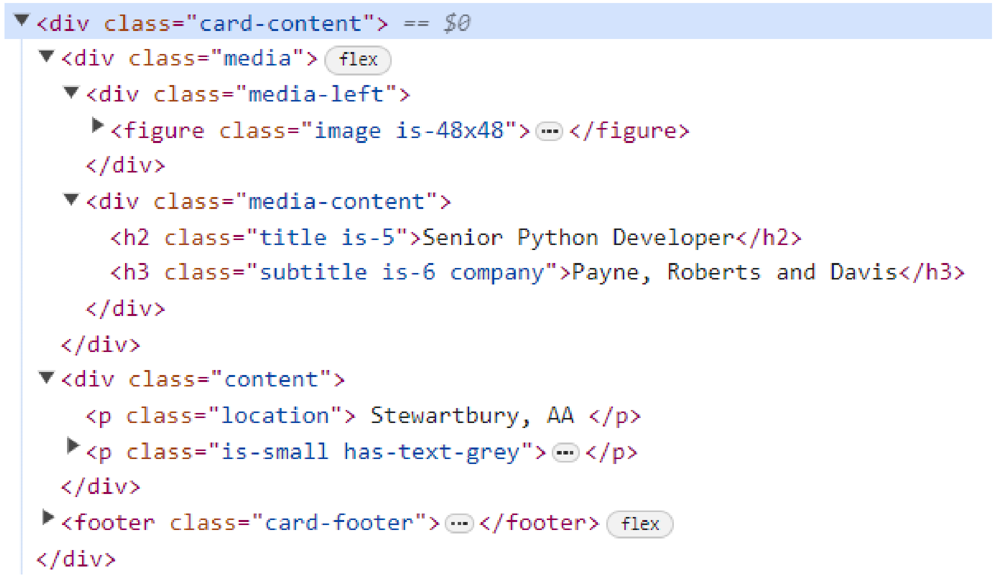


```
<section class="section">
▼<div class="container mb-5">
   <h1 class="title is-1"> Fake Python </h1>
   <p class="subtitle is-3"> Fake Jobs for Your W
 </div>
▼<div class="container">
  ▼<div id="ResultsContainer" class="columns is-m
    ▼<div class="column is-half">
      ▼<div class="card">
        ▼<div class="card-content">
```

Web scraping demo from course on Real Python

- Step 3:
  - Find the elements (**find_all()**) you need by using the HTML class name
  - Result is an iterable part of the entire html soup

```python
soup = BeautifulSoup(page.content, "html.parser")

# more useful result: iterable
cards = soup.find_all("div", class_='card-content')
for card in cards:
    print(card.prettify())
```

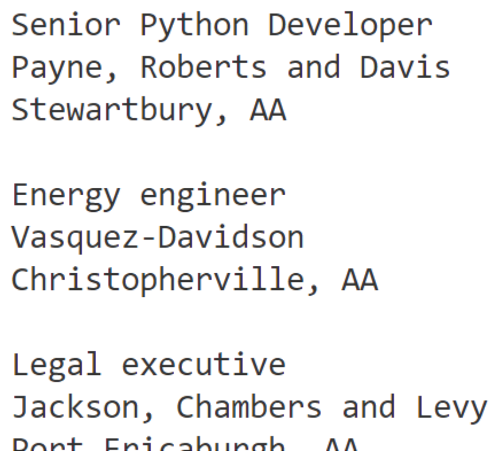


```
▼<div class="card-content"> == $0
  ▼<div class="media"> flex
    ▼<div class="media-left">
      ▶<figure class="image is-48x48">⋯</figure>
     </div>
    ▼<div class="media-content">
       <h2 class="title is-5">Senior Python Developer</h2>
       <h3 class="subtitle is-6 company">Payne, Roberts and Davis</h3>
     </div>
   </div>
  ▼<div class="content">
     <p class="location"> Stewartbury, AA </p>
    ▶<p class="is-small has-text-grey">⋯</p>
   </div>
  ▶<footer class="card-footer">⋯</footer> flex
</div>
```

Web scraping demo from course on Real Python

- Step 4:
  - Filter the information you need out of each card using **find**()

```python
cards = soup.find_all("div", class_='card-content')
for card in cards:
    title_element = card.find("h2", class_="title")
    company_element = card.find("h3", class_="company")
    location_element = card.find("p", class_="location")
    print(title_element.text)
    print(company_element.text)
    print(location_element.text.strip())
    print()
```



```
Senior Python Developer
Payne, Roberts and Davis
Stewartbury, AA

Energy engineer
Vasquez-Davidson
Christopherville, AA

Legal executive
Jackson, Chambers and Levy
Port Ericaburgh, AA
```

Web scraping demo from course on Real Python

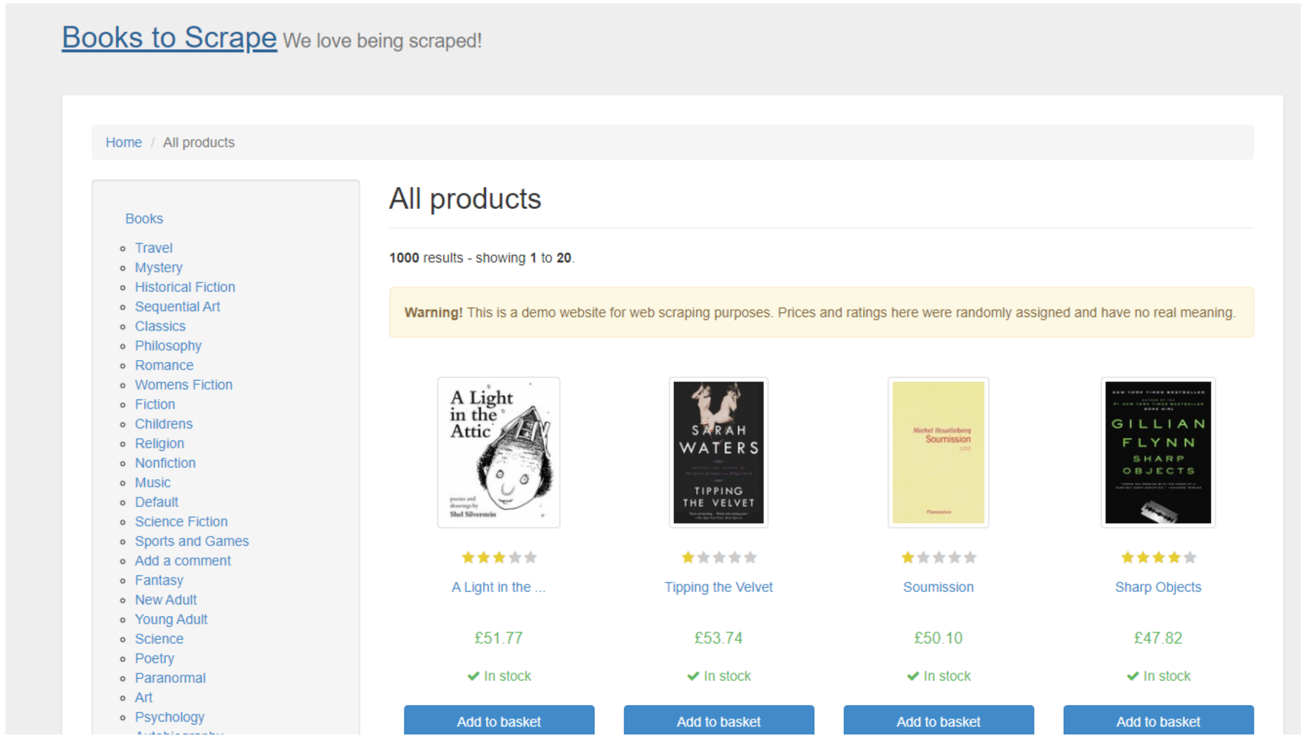
- Step 4 bis:
  - Use **Regex** to filter the information you need out of each card

```python
import re
…
cards = soup.find_all("div", class_='card-content’)

for card in cards:
    job = re.search(r'-5">(.+)</h2>', str(card))
    company = re.search(r'y">(.+)</h3>', str(card))
    location = re.search(r'ion">\s(.+),\s(.+)',str(card))

    print(job.group(1), ' @ ', company.group(1))
    print (location.group(1).strip(),',',location.group(2))
    print()
```

```
Senior Python Developer  @  Payne, Roberts and Davis
Stewartbury , AA

Energy engineer  @  Vasquez-Davidson
Christopherville , AA

Legal executive  @  Jackson, Chambers and Levy
Port Ericaburgh , AA

Fitness centre manager  @  Savage-Bradley
East Seanview , AP
```

Time to practice

- Exercise scraping books from the site books.toscrape.com



Books to Scrape We love being scraped!

Home / All products

Books
- Travel
- Mystery
- Historical Fiction
- Sequential Art
- Classics
- Philosophy
- Romance
- Womens Fiction
- Fiction
- Childrens
- Religion
- Nonfiction
- Music
- Default
- Science Fiction
- Sports and Games
- Add a comment
- Fantasy
- New Adult
- Young Adult
- Science
- Poetry
- Paranormal
- Art
- Psychology

All products

1000 results - showing **1** to **20**.

Warning! This is a demo website for web scraping purposes. Prices and ratings here were randomly assigned and have no real meaning.

★★★☆☆	★☆☆☆☆	★☆☆☆☆	★★★★☆
A Light in the ...	Tipping the Velvet	Soumission	Sharp Objects
£51.77	£53.74	£50.10	£47.82
✓ In stock	✓ In stock	✓ In stock	✓ In stock
Add to basket	Add to basket	Add to basket	Add to basket

Exercise result after part 3:

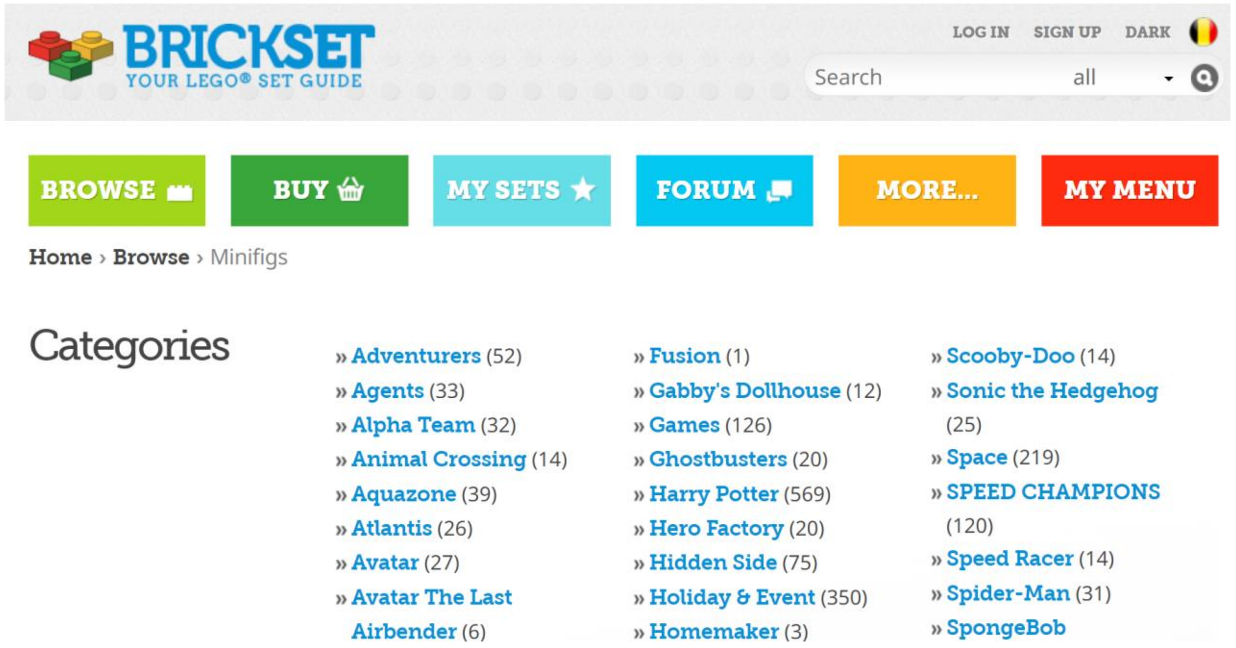
```
A Light in the Attic  :  £51.77
Tipping the Velvet  :  £53.74
Soumission  :  £50.10
Sharp Objects  :  £47.82
Sapiens: A Brief History of Humankind  :  £54.23
The Requiem Red  :  £22.65
The Dirty Little Secrets of Getting Your Dream Job  :  £33.34
The Coming Woman: A Novel Based on the Life of the Infamous Fe
```

Exercise result after part 4:

```
Books : 1000 titles
Travel : 11 titles
Mystery : 32 titles
Historical Fiction : 26 titles
Sequential Art : 75 titles
Classics : 19 titles
Philosophy : 11 titles
Romance : 35 titles
Womens Fiction : 17 titles
```

Time to practice

- Exercise scraping the list of minifigs from the Brickset site



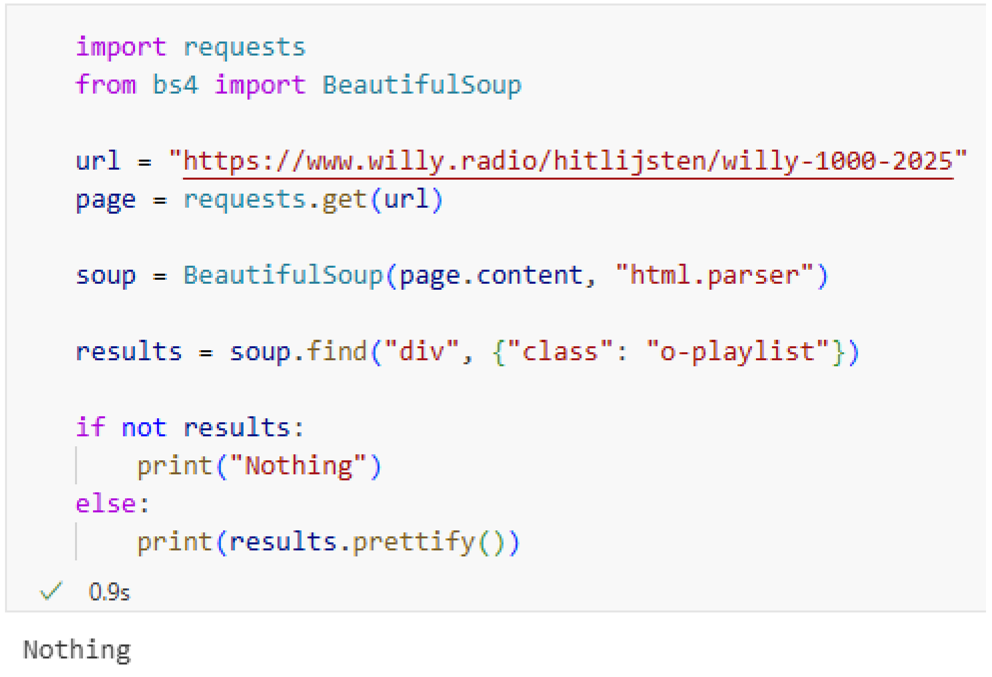
Home › Browse › Minifigs

Categories

» Adventurers (52)
» Agents (33)
» Alpha Team (32)
» Animal Crossing (14)
» Aquazone (39)
» Atlantis (26)
» Avatar (27)
» Avatar The Last Airbender (6)
» Fusion (1)
» Gabby's Dollhouse (12)
» Games (126)
» Ghostbusters (20)
» Harry Potter (569)
» Hero Factory (20)
» Hidden Side (75)
» Holiday & Event (350)
» Homemaker (3)
» Scooby-Doo (14)
» Sonic the Hedgehog (25)
» Space (219)
» SPEED CHAMPIONS (120)
» Speed Racer (14)
» Spider-Man (31)
» SpongeBob

The Willy 1000

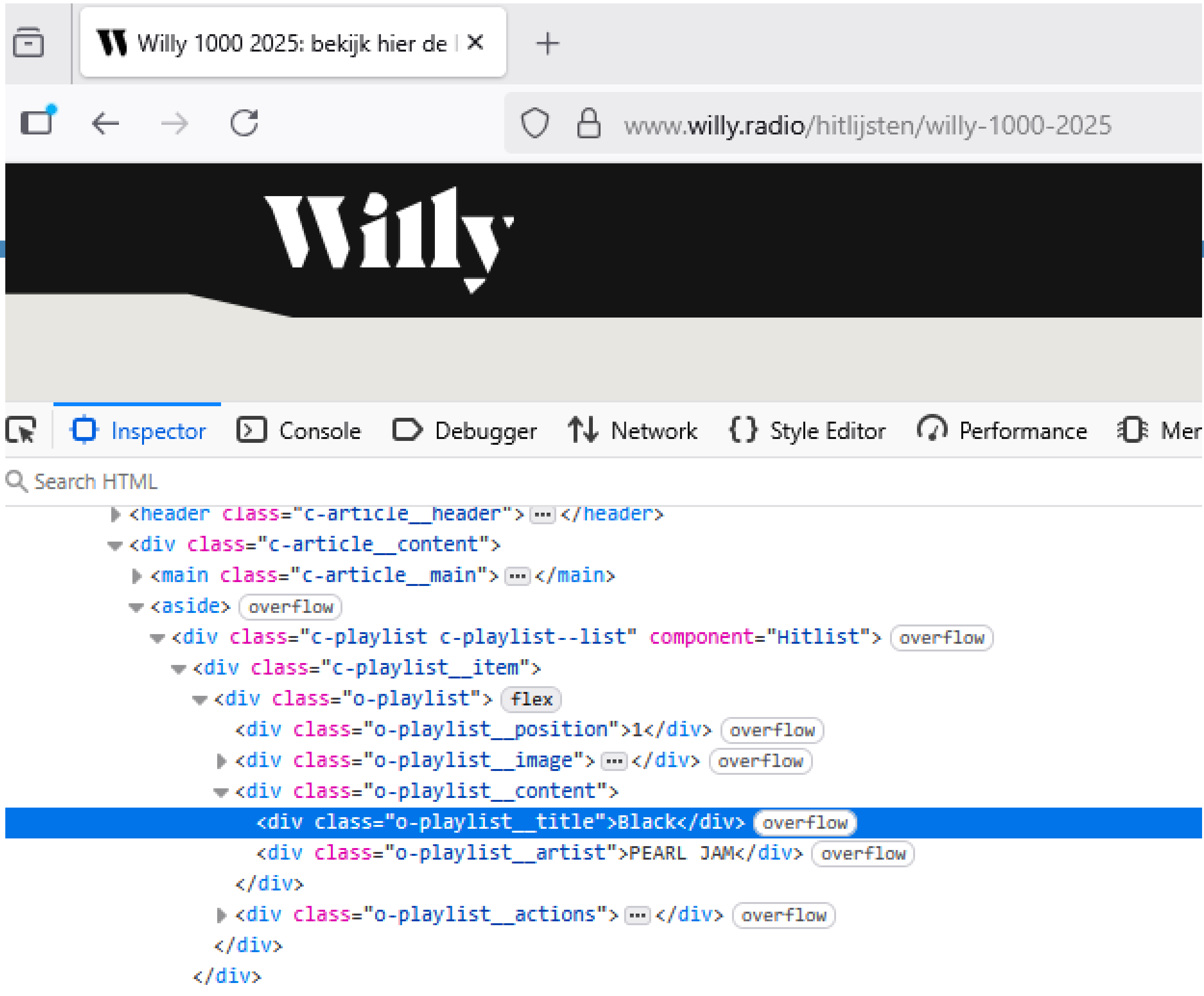
- Is a list of 1000 songs that you might want to scrape.
- The problem:



```python
import requests
from bs4 import BeautifulSoup

url = "https://www.willy.radio/hitlijsten/willy-1000-2025"
page = requests.get(url)

soup = BeautifulSoup(page.content, "html.parser")

results = soup.find("div", {"class": "o-playlist"})

if not results:
    print("Nothing")
else:
    print(results.prettify())
```

✓ 0.9s

```
Nothing
```

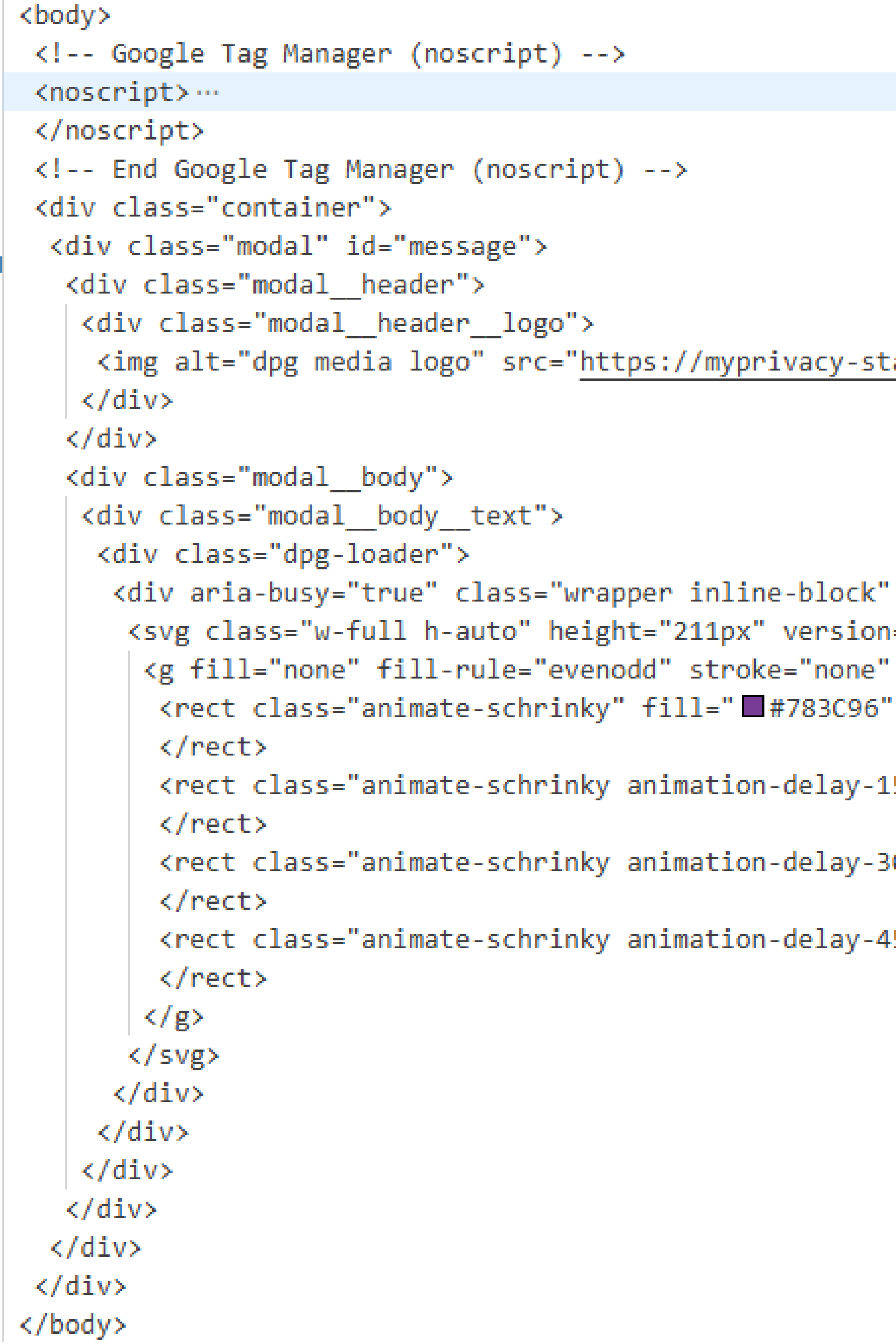


Willy 1000 2025: bekijk hier de
www.willy.radio/hitlijsten/willy-1000-2025
Willy
Inspector
Console
Debugger
Network
Style Editor
Performance
Mer
Search HTML

```html
▸ <header class="c-article__header"> ⋯ </header>
▾ <div class="c-article__content">
  ▸ <main class="c-article__main"> ⋯ </main>
  ▾ <aside> overflow
    ▾ <div class="c-playlist c-playlist--list" component="Hitlist"> overflow
      ▾ <div class="c-playlist__item">
        ▾ <div class="o-playlist"> flex
            <div class="o-playlist__position">1</div> overflow
          ▸ <div class="o-playlist__image"> ⋯ </div> overflow
          ▾ <div class="o-playlist__content">
              <div class="o-playlist__title">Black</div> overflow
              <div class="o-playlist__artist">PEARL JAM</div> overflow
            </div>
          ▸ <div class="o-playlist__actions"> ⋯ </div> overflow
          </div>
        </div>
```

Selenium

- When you look at the source code of the page in Python, there is no list.
- This is because the page you download is a template that is loaded with the correct data by a JavaScript running on the website.
- The solution is using Selenium, which makes Python act like a browser
    - This technology is also used to test applications, as it can automate clicks
- (We won’t be covering it in this course, but remember the name.)



```
<body>
 <!-- Google Tag Manager (noscript) -->
 <noscript>⋯
 </noscript>
 <!-- End Google Tag Manager (noscript) -->
 <div class="container">
  <div class="modal" id="message">
   <div class="modal__header">
    <div class="modal__header__logo">
     <img alt="dpg media logo" src="https://myprivacy-st
    </div>
   </div>
   <div class="modal__body">
    <div class="modal__body__text">
     <div class="dpg-loader">
      <div aria-busy="true" class="wrapper inline-block"
       <svg class="w-full h-auto" height="211px" version
        <g fill="none" fill-rule="evenodd" stroke="none"
         <rect class="animate-schrinky" fill="🟪#783C96"
         </rect>
         <rect class="animate-schrinky animation-delay-1
         </rect>
         <rect class="animate-schrinky animation-delay-3
         </rect>
         <rect class="animate-schrinky animation-delay-4
         </rect>
        </g>
       </svg>
      </div>
     </div>
    </div>
   </div>
  </div>
 </div>
</body>
```