PyLadies Boston: Web Scraping

Presented by Cherise Bryan

June 27, 2025

Ethics

- Respect robots.txt
  - Sample: https://www.meetup.com/robots.txt
- How does my web scraping affect others?
  - Am I negatively impacting the sales of businesses (especially small businesses)?
  - Am I making data that is meant to be private, public?
  - Am I exposing backdoors to websites that can be taken advantage of by others?

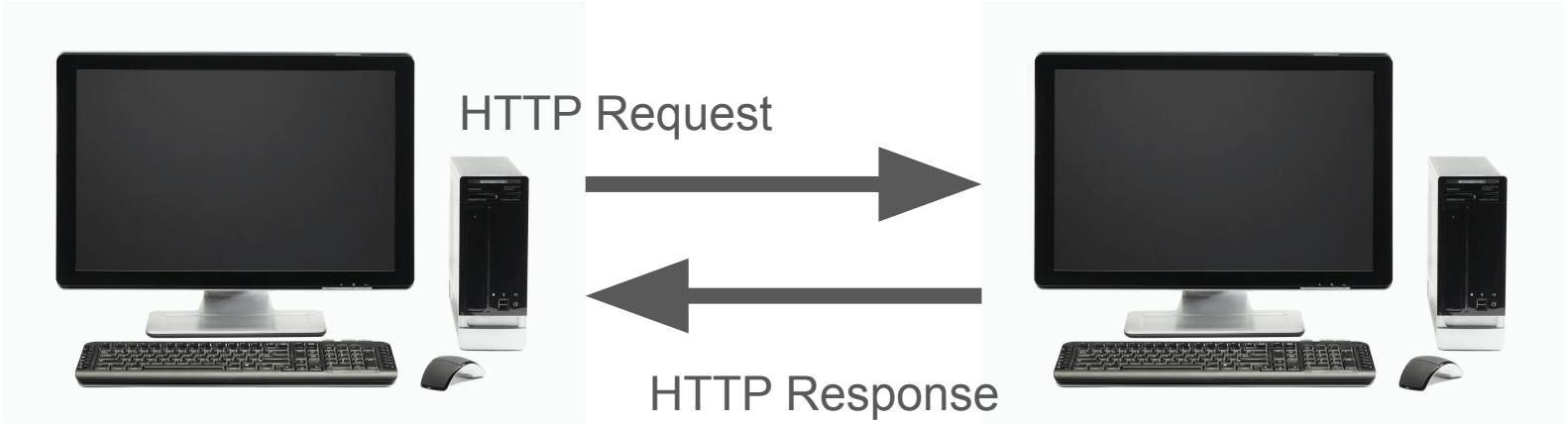
GitHub

https://github.com/CheriseCodes/pyladies-boston-webscraping

What is web scraping?

1. Make an **HTTP request** to a server (website)
    - Usually HTTP **GET** or **POST**
2. Receive an **HTTP response**
3. Parse the response (likely contains **HTML** or **JSON**)
4. Use the data parsed from the response (store it for later or use it immediately)

HTTP



Device A = Client
123.123.123.1

Device B = Server
my-domain.com => 123.123.124.*

HTML

```
<span id="main-menu" class="vector-dropdown-label-text">
    Main menu
</span>
```

- HTML element = element type + attributes
- Select HTML using CSS selectors (element type, id, class, etc.), XPATH, or other methods
- More on CSS selector: https://www.w3schools.com/cssref/css_selectors.php

Scraping with AI

- Least effort required
- Most popular: Crawl4AI

Scraping with Pandas

- Extract tables from HTML easily

Scraping with Beautiful Soup

- Tried and true
- Very flexible HTML parsing
- Beautiful Soup docs: https://beautiful-soup-4.readthedocs.io/en/latest/

Scraping with Selenium

- Was built to test websites
- Best for automating “clicks”
- Also has a great HTML parser (Locator strategies | Selenium)
- Selenium docs: https://selenium-python.readthedocs.io/locating-elements.html

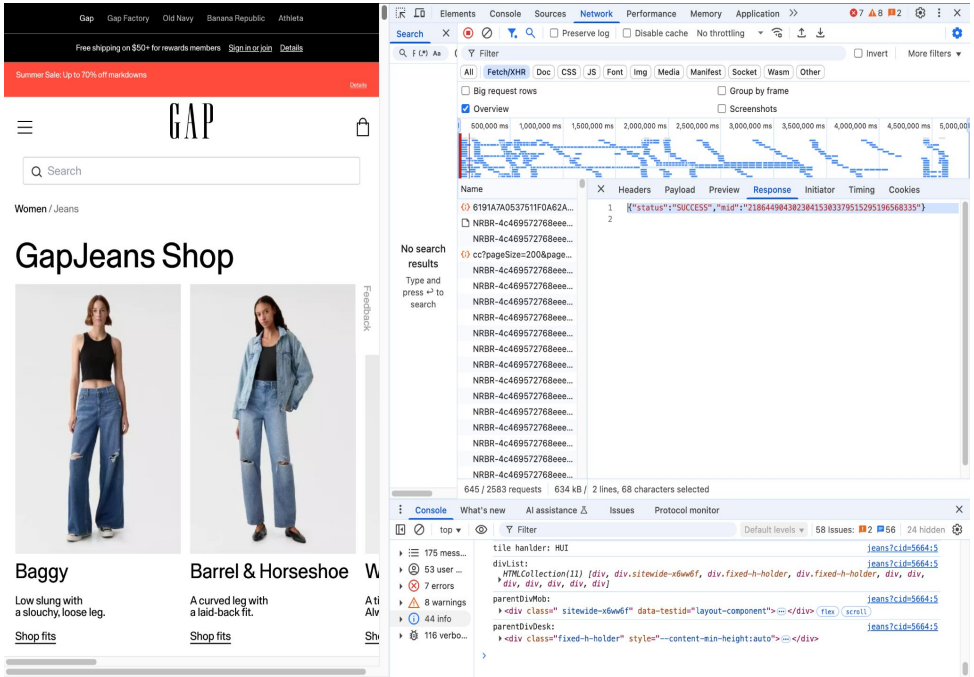
Crawling with Scrapy

- Web Crawling = systematically following links to collect data
- Scrapy is a framework for web crawling
- Scrapy docs: https://docs.scrapy.org/en/latest/index.html

Ditching HTML for JSON

…featuring a detour to Chrome DeveTools (Network)…

https://reddit.com/…anything.json



Chrome DevTools Protocol (cdp)

- Created as a website testing tool (see a pattern?)
- Allows you to interact with Chrome using POST requests
- May be used under the hood
- CDP docs: https://chromedevtools.github.io/devtools-protocol/

More on HTTP requests

- HTTP requests from Python may be blocked by captchas
- Solutions:
  - Set the User-Agent header to a web browser
  - Use a VPN or Proxy (https://proxyscrape.com/free-proxy-list)
  - Use specialized tools (curl_cffi, Selenium Driverless, Nodriver, captcha solvers etc.)
  - Use “fake” / “burner” accounts
  - Mimic human behaviour