Web Scraping W/ Python

BS4, Selenium, XPath & Friends

Brian Nguyen 2021

About Me

1. Background:
   - Neuroscience 🧠
   - Computational Modelling 💻
   - Intelligent Systems 🤖
   - Start-up Founder 💼
2. Currently:
   - MSBA Student 📊
   - Can’t wait to get scraping! 🤗

linkedin.com/in/brian-nguyen0305

B.S. Neuroscience (UTD 2020)
M.S. Applied Cog-Neuro (UTD 2020)

Agenda

1. HTML Basics
2. Chrome DevTools
3. Compare Web-Scraping Packages
4. BeautifulSoup
   - Simple Exercise
5. Selenium
   - A-tad-harder Exercise
6. Advanced Scraping and Crawling Demo
7. Ethical & Efficiency Discussion

Goals

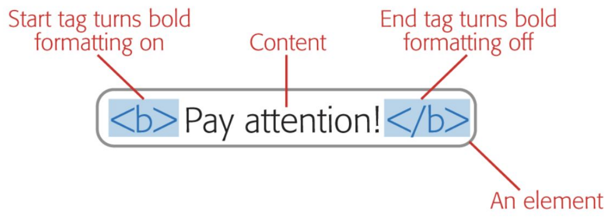
1. **Inspect** an HTML page & Identify what to scrape.
2. **Scrape** with requests and BeautifulSoup.
3. **Drive** web crawling with Selenium.
4. How to be a **responsible** Scraper.

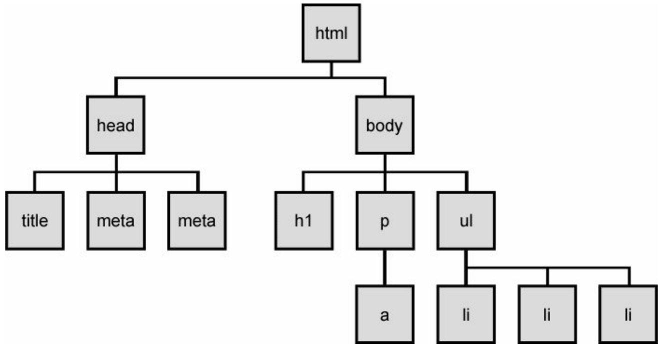
Why Do We Scrape?

1. Build **Datasets:**
    - Texts
    - Numbers
    - Images
2. For **Analysis** 📈
    - Sales
    - Marketing
3. For **Machine Learning** 🤖
4. End-to-end **testing**

HTML Basics

1. **Hypertext Markup Language (HTML)**
2. Standard markup language for documents.
3. Instruct web browser how to display content.
    - Provide structure.
    - + Cascading Style Sheets (**CSS**) = **Style.**
    - + **JavaScript** (or any script) = **Interactive.**
4. **Tags < >** are the Elements.
    - Paired
    - Start: `<head>`
    - End: `</head>`


Start tag turns bold formatting on
Content
End tag turns bold formatting off
<b>Pay attention!</b>
An element



🌲 **Tree Structure** 🌲

HTML Basics: Common Tags

1. <!DOCTYPE html> declaration defines this document to be HTML5.
2. <html> element is the root element of an HTML page.
3. <div> tag defines a division or a section in an HTML document. It's usually a container for other elements.
4. <head> element contains meta information about the document.
5. <title> element specifies a title for the document.
6. <body> element contains the visible page content.
7. <h1> element defines a large heading.
8. <p> element defines a paragraph.
9. <a> element defines a hyperlink.
10. And **Many More!**

Make A Simple HTML Page

1. Please head to this **Google Colab Notebook:**

   https://tinyurl.com/web-scrape-utd-main

2. Yayyy! No installations headaches. 🤞
3. We will use this throughout our Workshop today.
4. Let us know if you have any problem! 🤔 → 🙋‍♂️

Chrome DevTools

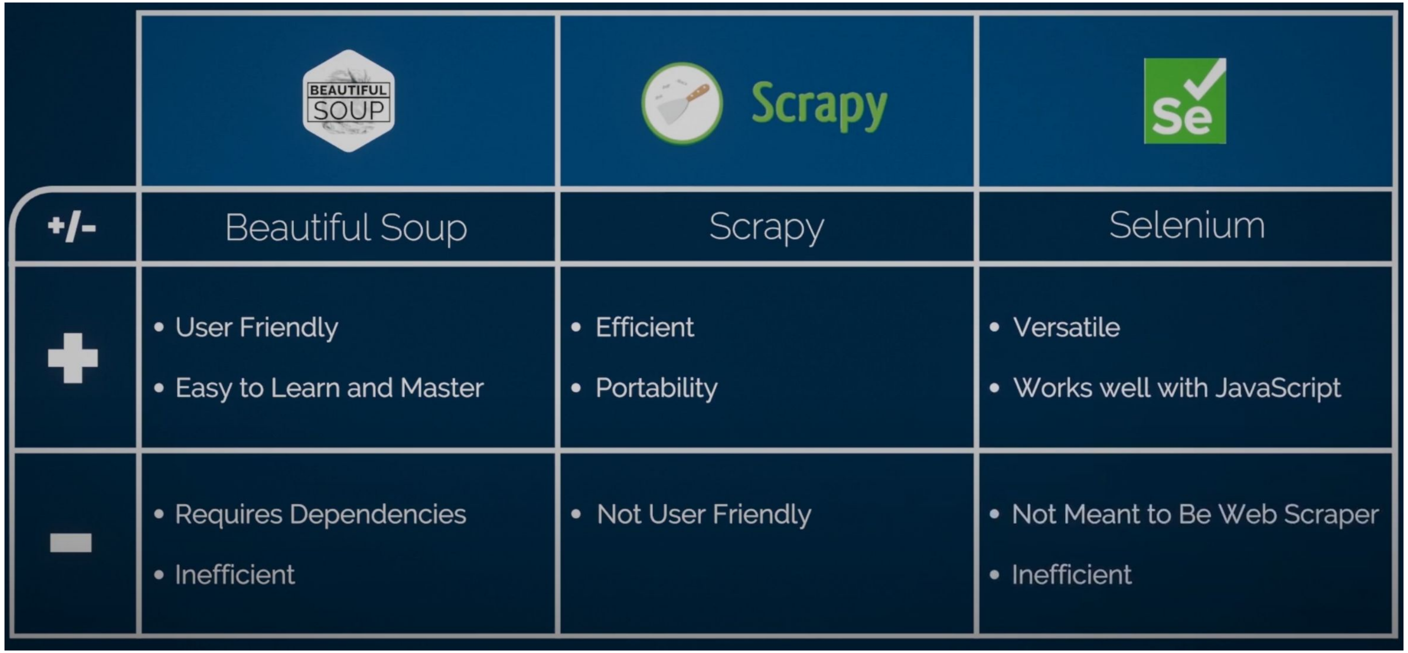
1. Built in to Chrome.
2. Super useful tool:
   a. **View** Source
   b. **Inspect** Elements
   c. **Edit** Webpage
3. Equivalence available for other browsers.

Quick Exercise

1. **What is your favourite Website?**
   - IMDB
   - Associated Press
   - Reddit
   - LinkedIn
2. Tasks:
   - Find Logo
   - Find Text
   - Find a Button
3. **Command + Option + C (Mac)** or **Control + Shift + C (Windows)**

When you accidently hit F12 instead of F11 on Google Chrome
Alright. I've hacked into the mainframe and disabled their algorithms.

Web-Scraping Packages Versus



+/-	Beautiful Soup	Scrapy	Selenium
+	• User Friendly • Easy to Learn and Master	• Efficient • Portability	• Versatile • Works well with JavaScript
-	• Requires Dependencies • Inefficient	• Not User Friendly	• Not Meant to Be Web Scraper • Inefficient

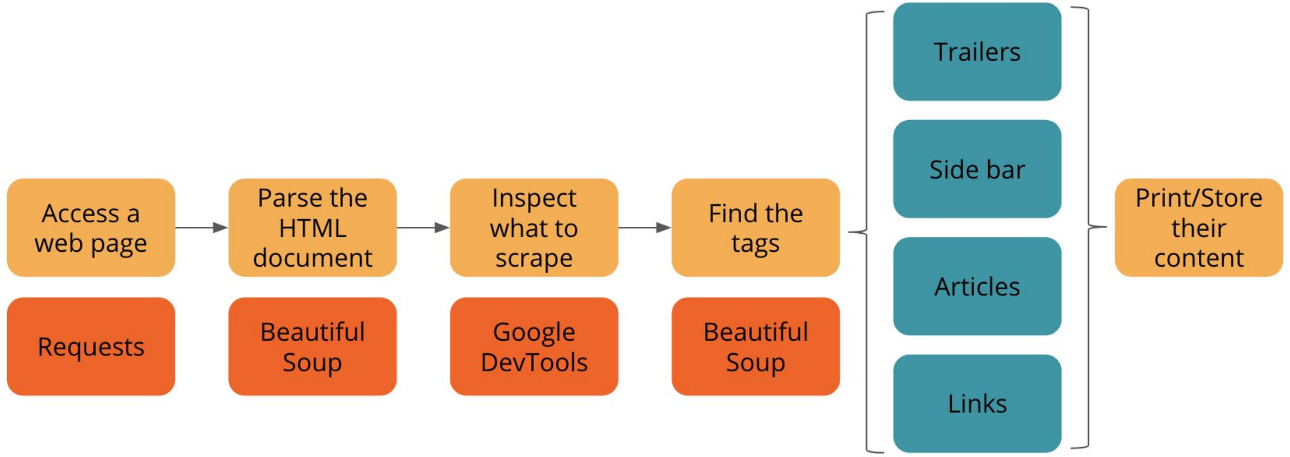
Kite Youtube Channel, 2020
https://www.youtube.com/watch?v=zucvHSQsKHA

Web-Scraping w/ BeautifulSoup 🍲

1. **Requests** access, collect page source (all code).
2. **BeautifulSoup** Is:
   - Python Library.
   - Extract HTML, XML files.
   - **Navigate** and **Scrape** Webpage's Tree structure.
3. Please head to the provided **Google Colab Notebook.**

Simple Exercise w/ BeautifulSoup 🍲

1. Please head to the provided **Google Colab Notebook.**
2. **Pipeline:**



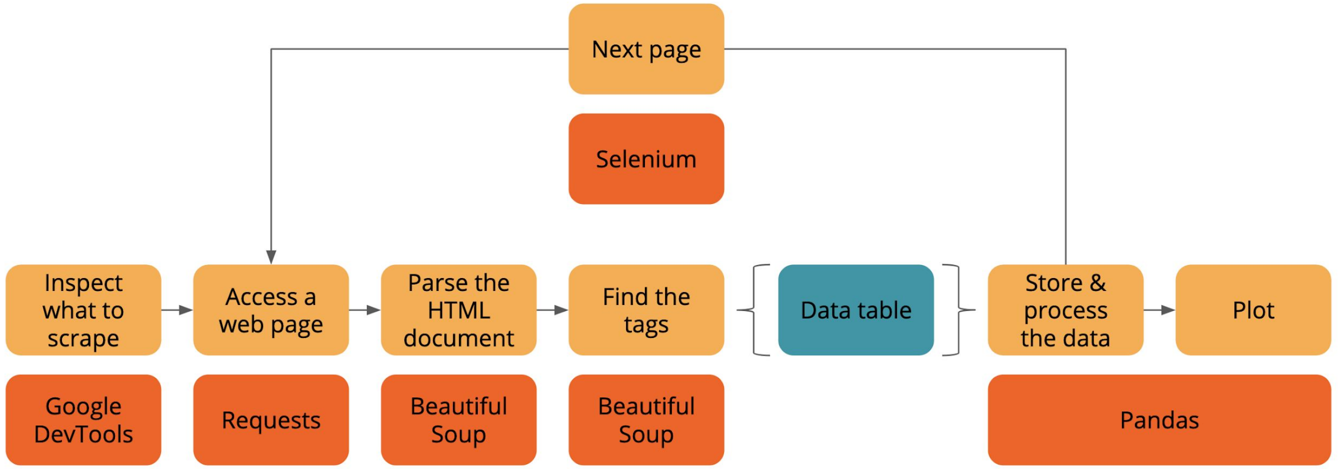
Web-Scraping w/ Selenium

1. The **most versatile** of all web-scraper.
2. In the right hand, it can become a **Powerful Web Automator** (Driver)
3. Only one can read **JavaScript** easily.
4. Can be very efficient when combined w/ Scrapy.
5. **IMO, Best Combo Right Now:**

Se + **XPath**

“A-tad-harder” Exercise w/ Selenium

1. Please head to the provided **Google Colab Notebook.**
2. **Pipeline:**



Don’t Give Up!

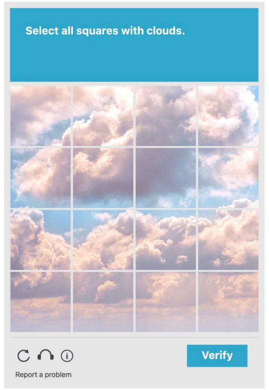
Advanced Scraping and Crawling Demo

1. **Crawling** is when you automate web interaction
   - Imitate human behavior.
   - Interact with website / search engine.
   - Useful when solving complex problems where you don't know the URL.
2. Scenario:
   - Give: List of Graduates (just names).
   - Find: first job & time from LinkedIn.
3. **Be Creative!**

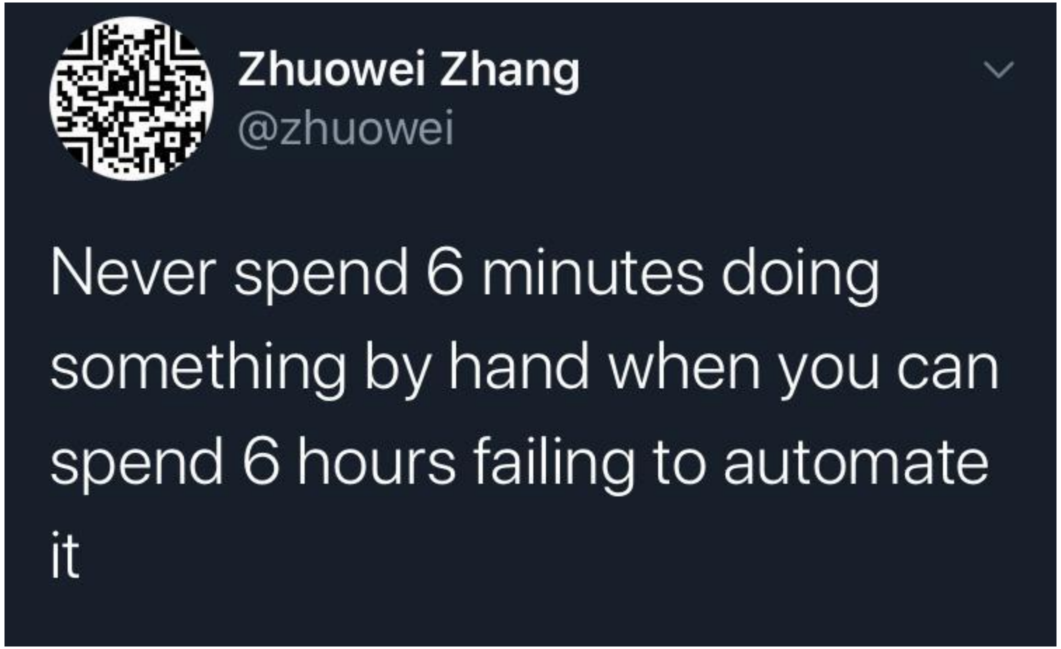
https://tinyurl.com/web-scrape-utd-demo **(Local Machine Only)**

Ethical & Efficiency Discussion

1. Now that you can scrape - **Should you tho?**
2. **Ethical:**
   - Read the Terms of Service and Privacy Policies first.
   - The Robots Exclusion Protocol (Captcha) → User-Agent (Login info, name, email).
   - Respect - No Ownership. Return Values - No Duplication.
3. **Efficiency:**
   - Use API if available.
   - Only extract, save what you need.
   - Use the right tools (Scraping Images is a whole diff. story).
   - Run off-peak hours & Space out requests.



A Wise Engineer Once Said...


Zhuowei Zhang
@zhuowei

Never spend 6 minutes doing something by hand when you can spend 6 hours failing to automate it

Happy Scrapin’