WebScraping with JavaScript

Daniel Goldman

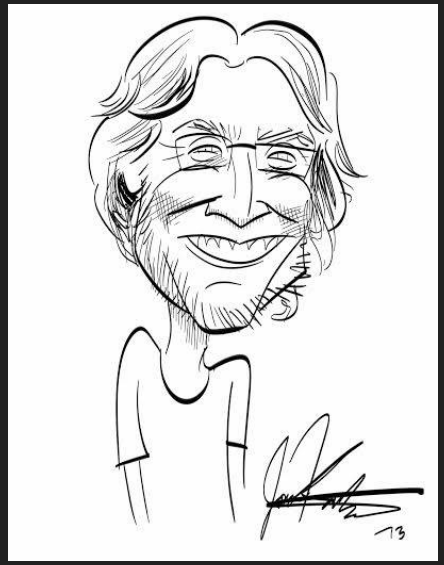
Me (Daniel Goldman)

Studied Mathematics at Wesleyan University

WDI Immersive Grad at General Assembly

Not an Expert

(I might stutter, but in an endearing way)



^Drawn by a New Yorker Cartoonist...

What is Web Scrapping?

Collection Data from the Web:

Plan A: API

Plan B: Web Scraping

(Plan C: Get an intern to do it by hand?)

Some advantages over using APIs:

-Websites tend to be better maintained than APIs (up to date, don't go down, etc)

-No rate limit/ API key nonsense

Warning: Can be legally "iffy"

For more info: http://www.bna.com/legal-issues-raised-by-the-use-of-web-crawling-and-scraping-tools-for-analytics-purposes

Technologies We'll Cover

Cheerio

Request

PhantomJS/Phantom

CasperJS

And some we won't Cover:

Nightmare

Spooky

-X-Ray

Request/Cheerio

Request: Lightweight HTTP requests directly to and from your server

Cheerio: Minimal jQuery functionality for your server

Pros:

-Lightweight, very easy to setup and use

-Node compatible

Cons:

-Very limited use...

PhantomJS/Phantom

Headless browser - simulates full browser functionality without a GUI

Designed primarily for testing

Pros:

-All dynamic, post- HTTP response behavior triggers automatically!

-Node compatible (with Phantom)

Cons:

-Huge, and hard to set up

-Still stuck with asynchronicity limitations...

CasperJS

Library that runs on top of PhantomJS

Designed more for scripting/Scraping

Pros:

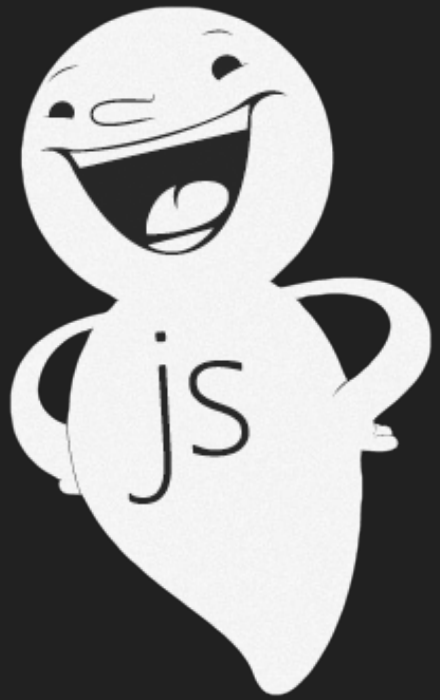
-Built in methods force synchronicity, can hit many URLs the nice way

-Additional methods for Web Crawling

Cons:

-Not Node compatible

-Documentation ain’t great



Other Technologies

-SpookyJS: Casper for node. Note: everyone seems to hate it.

- Electron/Nightmare: Alternate browser simulator (not a full headless browser). Apparently works with Rails?

-X-Ray: Lightweights, designed for scrapping. Can search with CSS selectors

-Worth Noting that many of the most popular Web Scraping tools are in Python

In Sum, How to Get Started

Questions to ask yourself:

About your app:

- Do I need a one-and-done scrape, or do I need to scrape upon user request?

About the site you're scrapping:

- Is the data I need scattered across multiple URLs?
- Is my data rendered upon the page loading?
- Do I need to dynamically interact with the page to make the data available?
- Do I have size limitations for my scrapper?

fin

Code: https://github.com/DZGoldman/Webscraping-Presentation

Slides: available somewhere

me: dzgoldman@wesleyan.edu

http://danielzgoldman.com/

Thanks.