Intro to Web Scraping with Python

Inland Empire Python Users Group
Oct 3, 2013

John Sheehan

What is Web Scraping?

A brute force way of getting data off of web pages

What is Web Scraping?

A brute force way of getting data off of web pages that:

- You don’t own (if you did you’d already have the data you need)

What is Web Scraping?

A brute force way of getting data off of web pages that:

- You don't own (if you did you'd already have the data you need)
- There is no API for getting the data directly (either as JavaScript or as a web service)

What is Web Scraping?

A brute force way of getting data off of web pages that:

- You don't own (if you did you'd already have the data you need)
- There is no API for getting the data directly (either as JavaScript or as a web service)

Content could be comprised of download links, pricing information, sport scores, election results, geographic coordinates, etc.

Miscellanea:

- If you are doing web scraping of a web site (because there is no API) on a regular basis, be aware of any legal issues.

Miscellanea:

- If you are doing web scraping of a web site (because there is no API) on a regular basis, be aware of any legal issues.

- If you are programmatically grabbing a ridiculous amount of data over a short period of time, be a good netizen and warn the site about the potential high traffic coming their way.

Miscellanea:

- If you are doing web scraping of a web site (because there is no API) on a regular basis, be aware of any legal issues.
- If you are programmatically grabbing a ridiculous amount of data over a short period of time, be a good netizen and warn the site about the potential high traffic coming their way.
- Every site is different, so every Python scraping program will be unique.

Miscellanea:

- If you are doing web scraping of a web site (because there is no API) on a regular basis, be aware of any legal issues.
- If you are programmatically grabbing a ridiculous amount of data over a short period of time, be a good netizen and warn the site about the potential high traffic coming their way.
- Every site is different, so every Python scraping program will be unique.
- Web pages change over time, so your code likely will too.

What this talk will be:

- A case study of a real-world situation

What this talk will be:

- A case study of a real-world situation
- A walk through of the basic process of programmatically pulling data from a web site

What the talk will NOT be:

- An in-depth coverage of any specific Python libraries

What the talk will NOT be:

- An in-depth coverage of any specific Python libraries
- An all-encompassing discussion of anything

The setup:

OMNI Magazine

- Published late 70s to mid 90s
- Founded by Bob Guccione (yes, THAT Bob Guccione)
- Covered science, sci-fi, parapsychology, art, technology, politics, and provocative futurethink in general
- OMNI Reboot: http://omnireboot.com/

The setup:

OMNI Magazine

- Published late 70s to mid 90s
- Founded by Bob Guccione (yes, THAT Bob Guccione)
- Covered science, sci-fi, parapsychology, art, technology, politics, and provocative futurethink in general
- OMNI Reboot: http://omnireboot.com/

Internet Archive

- Home of the Wayback Machine: http://archive.org/web/web.php
- Uploaded scans of every OMNI issue in various formats
- http://archive.org/details/omni-magazine

The Problem:

- I wanted to download every available issue in PDF format for future offline perusal (there was a chance that they could be removed from the site in short order).

The Problem:

- I wanted to download every available issue in PDF format for future offline perusal (there was a chance that they could be removed from the site in short order).

- The Internet Archive site didn't seem to offer any way of doing batch downloads and I didn't want to manually download 200 of them one-at-a-time.

The Solution:

Python of course!

Starting Point:

The web site of interest: http://archive.org/search.php?query=collection%3Aomni-magazine&sort=-publicdate

Starting Point:

The web site of interest: http://archive.org/search.php?query=collection%3Aomni-magazine&sort=-publicdate

Issues to consider:

- Multiple links on each web page
- Multiple web pages to traverse
- Have to drill down to detail page to get actual download link
- Multiple download links

The Process:

In General:

- Use web browser development tools to figure out what we're working with
- Use Python to work the magic
- Utilize the requests library to DL the web pages
- Utilize the BeautifulSoup library to find the links

The Process:

The Specifics:

- Load main page
- Get list of other pages ('page=.\Z')
  - Process main pages
  - For each main page, get list of detail pages ('/details/omni')
    - Process detail pages
    - For each detail page, download PDF file ('/download/omni.*\.pdf\Z')

Future Improvements

Multi-thread to parallelize downloads

Discussion