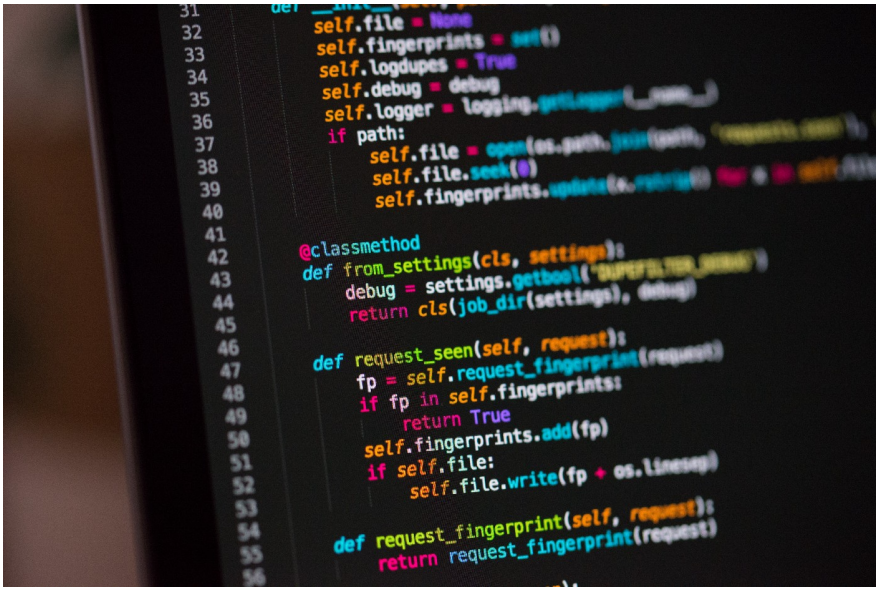
A Beginner’s Guide for Webscraping in Python



Web Scraping

Web scraping is a technique to automatically access and extract large amounts of information from a website, which can save a huge amount of time and effort. In this article, we will go through an easy example of how to automate downloading hundreds of files from the New York MTA. This is a great exercise for web scraping beginners who are looking to understand how to web scrape. Web scraping can be slightly intimidating, so this tutorial will break down the process of how to go about the process.

New York MTA Data

We will be downloading turnstile data from this site:

```
http://web.mta.info/developers/turnstile.html
```

Turnstile data is compiled every week from May 2010 to present, so hundreds of .txt files exist on the site. Below is a snippet of what some of the data looks like. Each date is a link to the .txt file that you can download.

Data Files

Saturday, September 22, 2018
Saturday, September 15, 2018
Saturday, September 08, 2018
Saturday, September 01, 2018
Saturday, August 25, 2018
Saturday, August 18, 2018
Saturday, August 11, 2018
Saturday, August 04, 2018
Saturday, July 28, 2018
Saturday, July 21, 2018
Saturday, July 14, 2018
Saturday, July 07, 2018
Saturday, June 30, 2018
Saturday, June 23, 2018
Saturday, June 16, 2018
Saturday, June 09, 2018
Saturday, June 02, 2018
Saturday, May 26, 2018
Saturday, May 19, 2018
Saturday, May 12, 2018
Saturday, May 05, 2018
Saturday, April 28, 2018
Saturday, April 21, 2018

It would be torturous to manually right click on each link and save to your desktop. Luckily, there's web-scraping!

Important notes about web scraping:

1. Read through the website's Terms and Conditions to understand how you can legally use the data. Most sites prohibit you from using the data for commercial purposes.

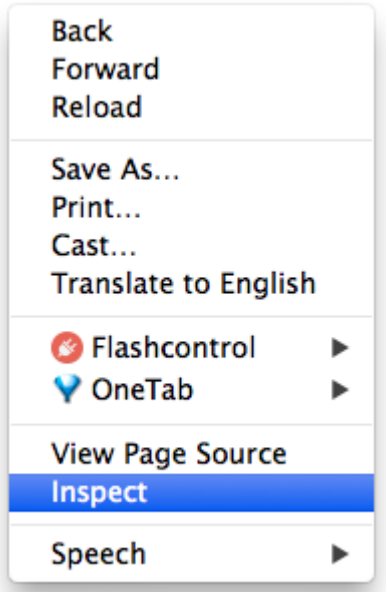
2. Make sure you are not downloading data at too rapid a rate because this may break the website. You may potentially be blocked from the site as well.

Inspecting the Website

The first thing that we need to do is to figure out where we can locate the links to the files we want to download inside the multiple levels of HTML tags. Simply put, there is a lot of code on a website page and we want to find the relevant

pieces of code that contains our data. If you are not familiar with HTML tags, refer to W3Schools Tutorials. It is important to understand the basics of HTML in order to successfully web scrape.

On the website, right click and click on “Inspect”. This allows you to see the raw code behind the site.



Once you’ve clicked on “Inspect”, you should see this console pop up.

Console

Notice that on the top left of the console, there is an arrow symbol.

If you click on this arrow and then click on an area of the site itself, the code for that particular item will be highlighted in the console. I've clicked on the very first data file, Saturday, September 22, 2018 and the console has highlighted in blue the link to that particular file.

```
<a
href="data/nyct/turnstile/turnstile_180922.txt">S
 September 22, 2018</a>
```

Notice that all the .txt files are inside the `<a>` tag following the line above. As you do more web scraping, you will find that the `<a>` is used for hyperlinks.

Now that we've identified the location of the links, let's get started on coding!

Python Code

We start by importing the following libraries.

```
import requests
import urllib.request
import time
from bs4 import BeautifulSoup
```

Next, we set the url to the website and access the site with our requests library.

```
url =
'http://web.mta.info/developers/turnstile.html'
response = requests.get(url)
```

If the access was successful, you should see the following output:

```
In [9]: response #200 means it went through
Out[9]: <Response [200]>
```

Next we parse the html with BeautifulSoup so that we can work with a nicer, nested BeautifulSoup data structure. If you are interested in learning more about this library, check out the BeatifulSoup documentation.

```
soup = BeautifulSoup(response.text,
"html.parser")
```

We use the method .findAll to locate all of our `<a>` tags.

```
soup.findAll('a')
```

This code gives us every line of code that has an `<a>` tag. The information that we are interested in starts on line 38 as seen below. That is, the very first text file is located in line 38, so we want to grab the rest of the text files located below.

```
 <a href="http://web.mta.info/accountability">Main Page</a>,
 <a href="http://web.mta.info/mta/boardmaterials.html">Board Materials</a>,
 <a href="http://web.mta.info/mta/budget/">Budget Info</a>,
 <a href="http://web.mta.info/capital">Capital Program Info</a>,
 <a href="http://web.mta.info/capitaldashboard/CPDHome.html">Capital Program Dashboard</
a>,
 <a href="http://web.mta.info/mta/investor/">Investor Information</a>,
 <a href="http://web.mta.info/mta/leadership/">MTA Leadership</a>,
 <a href="http://web.mta.info/persdashboard/performance14.html">Performance Indicators</
a>,
 <a href="http://www.mta.info/mta-news">Press Releases and News</a>,
 <a href="http://web.mta.info/mta/news/hearings">Public Hearings</a>,
 <a class="last" href="http://web.mta.info/mta/news/hearings/index-reinvention.html">Tra
nsportation Reinvention Commission</a>,
 <a name="main-content"> </a>,
 <a href="resources/nyct/turnstile/ts_Field_Description_pre-10-18-2014.txt">Prior to 10/
18/14</a>,
```

```
 <a href="resources/nyct/turnstile/ts_Field_Description.txt">Current</a>,
 <a href="resources/nyct/turnstile/Remote-Booth-Station.xls">Remote Unit/Control Area/St
ation Name Key</a>,
 <a href="data/nyct/turnstile/turnstile_180922.txt">Saturday, September 22, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180915.txt">Saturday, September 15, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180908.txt">Saturday, September 08, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180901.txt">Saturday, September 01, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180825.txt">Saturday, August 25, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180818.txt">Saturday, August 18, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180811.txt">Saturday, August 11, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180804.txt">Saturday, August 04, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180728.txt">Saturday, July 28, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180721.txt">Saturday, July 21, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180714.txt">Saturday, July 14, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180707.txt">Saturday, July 07, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180630.txt">Saturday, June 30, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180623.txt">Saturday, June 23, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180616.txt">Saturday, June 16, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180609.txt">Saturday, June 09, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180602.txt">Saturday, June 02, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180526.txt">Saturday, May 26, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180519.txt">Saturday, May 19, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180512.txt">Saturday, May 12, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180505.txt">Saturday, May 05, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180428.txt">Saturday, April 28, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180421.txt">Saturday, April 21, 2018</a>,
 <a href="data/nyct/turnstile/turnstile_180414.txt">Saturday, April 14, 2018</a>,
```

subset of all <a> tags

Next, let's extract the actual link that we want. Let's test out the first link.

```
one_a_tag = soup.findAll('a')[38]
link = one_a_tag['href']
```

This code saves the first text file, 'data/nyct/turnstile/turnstile_180922.txt' to our variable link. The full url to download the data is actually 'http://web.mta.info/developers/data/nyct/turnstile/turnstile_180922.txt' which I discovered by clicking on the first data file on the website as a test. We can use our urllib.request library to download this file path to our computer. We provide request.urlretrieve with two

parameters: file url and the filename. For my files, I named them “turnstile_180922.txt”, “turnstile_180901”, etc.

```
download_url =
'http://web.mta.info/developers/'+ link
urllib.request.urlretrieve(download_url,'./'+link
```

Last but not least, we should include this line of code so that we can pause our code for a second so that we are not spamming the website with requests. This helps us avoid getting flagged as a spammer.

```
time.sleep(1)
```

Now that we understand how to download a file, let’s try downloading the entire set of data files with a for loop. The code below contains the entire set of code for web scraping the NY MTA turnstile data.

Note that line 19 should be 38 and not 36 due to an updated on the website.

You can find my Jupyter Notebook for this on my Github.