Scrapy Tutorial #1: Scrapy VS Beautiful Soup

Last updated on Feb 25 2019 by Michael Yin

Introduction:

This is the #1 post of my Scrapy Tutorial Series, in this Scrapy tutorial, I will talk about the features of Scrapy, BeautifulSoup, compare them, and help you decide which one is better for your projects.

Talk About BeautifulSoup

BeautifulSoup is a tool which help programmer quickly extract valid data from web pages, its API is very friendly to newbie developer, and it can also handle malformed markup very well. However, **in most cases, BeautifulSoup alone can not get the job done, you need use another package such as `urlib2` or `requests` to help you download the web page and then you can use BeautifulSoup to parse the HTML source code**. The doc of `BeautifulSoup` is very comprehensive you can get a lot of examples there and quickly learn how to use it.

BeautifulSoup works fine on Python 2 and Python 3, so compatibility will not be a problem, below is a code example of `BeautifulSoup`, as you can see, it is very beginner-friendly.

```python
from bs4 import BeautifulSoup
soup = BeautifulSoup(html_doc, 'html.parser')

for link in soup.find_all('a'):
    print(link.get('href'))
# http://example.com/elsie
# http://example.com/lacie
# http://example.com/tillie
```

Talk About Scrapy

Scrapy is a web crawling framework for developer to write code to create `spider`, which define how a certain site (or a group of sites) will be scraped. The biggest feature is that it is built on Twisted, an asynchronous networking library, so Scrapy is implemented using a non-blocking (aka asynchronous) code for concurrency, which makes the spider performance is very great.

For those who have no idea what is `asynchronous`, here is a simple explanation.

> When you do something synchronously, you wait for it to finish before moving on to another task. When you do something asynchronously, you can move on to another task before it finishes.

Scrapy also works fine on Python 2 and Python 3, so compatibility will not be a problem. It has built-in support for extracting data from HTML sources using XPath expression and CSS expression.

Which one should you choose?

The two Python web scraping tools are created to do different jobs. `BeautifulSoup` is only used to parse HTML and extract data, `Scrapy` is used to download HTML, process data and save it.

When you compare `BeautifulSoup` vs `Scrapy` to figure out what is the best for your project, you should consider many factors.

Learning Curve

`BeautifulSoup` is very easy to learn, you can quickly use it to extract the data you want, in most cases, you will also need a downloader to help you get the HTML source, it is highly recommended to use `Requests` package instead of `urllib2` from built-in python library to implement this function.

Since `Scrapy` does no only deal with content extraction but also many other tasks such as downloading HTML, learning curve of `Scrapy` is much steeper, you need to read some Scrapy Tutorial or Scrapy Doc to understand how it works, and work hard to become a Scrapy expert.

If you are a newbie developer, have not much experience in programming and want to get a small job done, `BeautifulSoup` might be your choice here because it is highly unlikely to let you down.

Ecosystem

Very few people have talked about this before when comparing web scraping tools. Think about why people like to use Wordpress to build CMS instead of other frameworks, the key is `ecosystem`. So many themes, plugins can help people quickly build a CMS which meet the requirement.

Scrapy have so many related projects, plugins on open source websites such as Github, and many discussions on StackOverflow can help you fix the potential issue. For example, if you want to use proxy with your spider project, you can check a project `scrapy-proxies` help you send HTTP requests using random proxy from list. All you need is just changing some settings.

Extensibility

The architecture of `Scrapy` is well designed, you can easily develop custom middleware or pipeline to add custom functionality. Your `Scrapy` project can be both robust and flexible. After you develop several Scrapy projects, you will benefit from the architecture and like its design because it is easy to migrate from existing Scrapy spider project to another one.

So if your project is small, the logic is not very complex and you want job done quickly, you can use `BeautifulSoup` to keep your project simple. If your project needs more customization such as proxy, data pipeline, then the `Scrapy` might be your choice here.

Performance

With `Scrapy`, the spider can send out many requests at the same time, so you need set `download_delay` in most cases to avoid getting banned, the web pages can be crawled quickly. However, `BeautifulSoup` do not have this feature so many people said that `BeautifulSoup` is slow. Actually, this is wrong, you can import `multiprocessing` to speed up the whole progress, but I must say many people using `BeautifulSoup` might do not know how to use `multiprocessing`.

So if you want to develop an efficient spider which can crawl many datasets in a short time, `Scrapy` can save you a lot of time. If you are not experienced Python developer, then `BeautifulSoup` should not be your choice here.

Conclusion

So which one is better? There is no solid answer, the answer depends heavily on the actual situation. Below is a quick reference table.

Framework	BeautifulSoup	Scrapy
Learning Curve	Very easy to learn, beginner-friendly	Learning curve of `Scrapy` is much steeper, you need to read some Scrapy Tutorial or Scrapy Doc to get started, and work hard to become an Scrapy expert.
Ecosystem	Not many related projects or plugins	Many related projects, plugins on open source websites such as Github, and many discussions on StackOverflow can help you fix the potential issue.
Extensibility	Not very easy to extend the project	You can easily develop custom middleware or pipeline to add custom function, easy to maintain.
Performance	You need import `multiprocessing` to make it run quicker	Very efficient, web pages can be crawled in a short time, on the other hand, in many cases you need to set download_delay to avoid getting spider banned.

In short, **If you have not much experience in programming, the job is a very simple project, then `BeautifulSoup` can be your choice. If you want a more powerful and flexible web crawler, or you indeed have some experience in programming, then `Scrapy` is definitely the winner here.**

Resources

- Scrapy Doc
- Beautiful Soup Doc
- Other review articles
    1. BeautifulSoup4 vs Scrapy
    2. When should you use Scrapy over BeautifulSoup?