Extract text from a webpage using BeautifulSoup and Python

February 12, 2019

If you're going to spend time crawling the web, one task you might encounter is stripping out visible text content from HTML.

If you're working in Python, we can accomplish this using BeautifulSoup.

Setting up the extraction

To start, we'll need to get some HTML. I'll use Troy Hunt's recent blog post about the "Collection #1" Data Breach.

Here's how you might download the HTML:

```
import requests

url = 'https://www.troyhunt.com/the-773-million-record-collection-1-data-reach/'
res = requests.get(url)
html_page = res.content
```

Now, we have the HTML.. but there will be a lot of clutter in there. How can we extract the information we want?

Creating the "beautiful soup"

We'll use Beautiful Soup to parse the HTML as follows:

```
from bs4 import BeautifulSoup

soup = BeautifulSoup(html_page, 'html.parser')
```

Finding the text

BeautifulSoup provides a simple way to find text content (i.e. non-HTML) from the HTML:

```
text = soup.find_all(text=True)
```

However, this is going to give us some information we don't want.

Look at the output of the following statement:

```
set([t.parent.name for t in text])

# {'label', 'h4', 'ol', '[document]', 'a', 'h1',
'noscript', 'span', 'header', 'ul', 'html', 'section',
'article', 'em', 'meta', 'title', 'body', 'aside',
'footer', 'div', 'form', 'nav', 'p', 'head', 'link',
'strong', 'h6', 'br', 'li', 'h3',
'h5', 'input', 'blockquote', 'main', 'script', 'figure'}
```

There are a few items in here that we likely do not want:

- `[document]`
- `noscript`
- `header`
- `html`
- `meta`
- `head`
- `input`
- `script`

For the others, you should check to see which you want.

Extracting the valuable text

Now that we can see our valuable elements, we can build our output:

```
output = ''
blacklist = [
    '[document]',
   'noscript',
    'header',
    'html',
    'meta',
    'head',
    'input',
    'script',
    # there may be more elements you don't want, such as "style", etc.
]

for t in text:
    if t.parent.name not in blacklist:
        output += '{} '.format(t)
```

The full script

Finally, here's the full Python script to get text from a webpage:

```
import requests
from bs4 import BeautifulSoup

url = 'https://www.troyhunt.com/the-773-million-record-collection-1-data-reach/'
res = requests.get(url)
html_page = res.content
soup = BeautifulSoup(html_page, 'html.parser')
text = soup.find_all(text=True)

output = ''
blacklist = [
    '[document]',
    'noscript',
    'header',
    'html',
    'meta',
    'head',
    'input',
    'script',
    # there may be more elements you don't want, such as "style", etc.
]

for t in text:
    if t.parent.name not in blacklist:
        output += '{} '.format(t)
```

```
print(output)
```

Improvements

If you look at `output` now, you'll see that we have some things we don't want.

There's some text from the header:

```
Home \n \n \n Workshops \n \n \n Speaking \n \n \n Media
\n \n
\n About \n \n \n Contact \n \n \n Sponsor \n \n \n \n \n
\n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n
\n    \n \n \n \n Sponsored by:
```

And there's also some text from the footer:

```
\n \n \n \n \n \n Weekly Update 122 \n \n \n \n \n Weekly
Update 121 \n \n \n \n \n \n \n \n Subscribe  \n \n \n \n
\n \n \n \n \n \n Subscribe Now! \n \n \n \n \r\n
Send new blog posts: \n    daily \n
 weekly \n \n \n \n Hey, just quickly confirm you\'re not
a robot: \n  Submitting... \n Got it! Check your email,
click the confirmation
link I just sent you and we\'re done. \n \n \n \n \n \n
\n \n Copyright 2019, Troy Hunt \n This work is licensed
under a  Creative Commons Attribution 4.0 International
License . In other words, share generously but provide
attribution. \n \n \n Disclaimer \n Opinions expressed
here are my own and may not reflect those of people I
work with, my mates, my wife, the kids etc. Unless I\'m
quoting someone, they\'re just my own views. \n \n \n
Published with Ghost \n This site runs entirely on  Ghost
and is made possible thanks to their kind support. Read
more about  why I chose to use Ghost . \n \n \n \n \n \n
\n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n \n
\n
 \n \n \n \n \n '
```

If you're just extracting text from a single site, you can probably look at the HTML and find a way to parse out only the valuable content from the page.

Unfortunately, the internet is a messy place and you'll have a tough time finding consensus on HTML semantics.

Good luck!