

Forum:[Hack Forums](#)

Thread: [BeautifulSoup4 Python Guide! \(Web Scraping\) \(Python 3\)](#)

AeroBlue - March 28, 2019, 6:22 pm

Introduction

Aero Wrote: This guide took me way longer than I anticipated so I will probably be adding more info later on. All feedback is welcome and appreciated, I am also willing to help anyone with questions they have regarding this post or bs4 / requests in general. I still don't really like the format of this post but I can't really think of a better way lol. Definitely my most HQ post I've ever made on HF, although that's not really saying much... Hope I can help a few people learn this amazing combination for webscraping in python!

What is Bs4?

Quote: Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching and modifying the parse tree. It commonly saves programmers hours or days of work.

Modules Used: Requests

Code pip install requests Bs4

Code pip install bs4 Bs4 and requests both require python 3 (3.6 recommended)

Step 1: Getting the page

```
Code url = "https://example.com/"
r = requests.get(url)
page = r.text
soup = BeautifulSoup(page, "html.parser")
print(soup.prettify())
```

Code Result: "<html>...</html>" If you need more help on using requests, please see [Nachzehrer's Awesome Guide](#)

Step 2: Parsing the page

Method 1: Find using soup.find() or soup.find_all()

Finding all links on a page using soup.find_all()

Code Example HTML: Somelink, Otherlink

```
Code links = [link.get("href") for link in soup.find_all("a")]
print(links)
```

Code Result: "example.com", "otherlink.net", etc.

Finding text in first <p> tag using soup.find()

Code Example HTML: <p>This is not the text you are looking for!</p>

```
Code paragraph = soup.find("p").text
print(paragraph)
```

Code Result: "This is not the text you are looking for!"

Method 2: Find using soup.select()

Finding by class

Code Example HTML: <p class="comment">Text to scrape</p>

```
Code
print(soup.select(".comment")) #make sure you are finding the right information
comments = [comment.text for comment in soup.select(".comment")] #Get text for all comments on
```

page
print(comments) Code Result: "Text to scrape","other comment","more comments","etc"
Finding by tags
Code Example HTML: <p>Random text to find</p> Code print(soup.select("p > span"))
blocks = [block.text for block in soup.select("p > span")]
print(blocks) Code Result: "Random text to find", "another block of text"
Finding by id
Code Example HTML: <p>Comments, yay!</p> Code
soup.select("#comments") OR soup.select("p > #comments")
comment_list = [comment.text for comment in soup.select("#comments")]
print(comment_list) Code Result: "comments, yay!", "comments are dumb", "more comments"

Useful Links[Latest Version of Python](#)
[BeautifulSoup4 Documentation](#)
[Requests Documentation](#)
[Lengthy Guide on Web Scraping](#)
[Nachzehrer's Guide on Requests](#)

Quevoy43 - April 6, 2019, 4:52 pm

This helped me tremendously, thank you!

AeroBlue - April 6, 2019, 6:51 pm

(Apr 6, 2019 04:52 PM)Quevoy43 Wrote: [\[CITE\]](#) This helped me tremendously, thank you!
Wow, I'm so glad I actually helped someone! If you have any suggestions or need any help just let me know! ?

friday305 - June 3, 2019, 2:16 pm

This helped me tremendously, thank you!

Deathbox - June 6, 2019, 7:10 am

Cheers buddy, had a go at Selenium last night but haven't looked at beautiful soup yet.

george12324 - June 23, 2019, 2:15 pm

(Mar 28, 2019 06:22 PM)Aero Blue Wrote: [\[CITE\]](#) Introduction

What is Bs4?

Modules Used:Requests
pip install requests Bs4
pip install bs4 Bs4 and requests both require python 3 (3.6 recommended)

Step 1: Getting the page

```
Code url = "https://example.com/"  
r = requests.get(url)  
page = r.text  
soup = BeautifulSoup(page, "html.parser")
```

print(soup.prettify()) Code Result: "<html>...</html>" If you need more help on using requests, please see [Nachzehrer's Awesome Guide](#)

Step 2: Parsing the page

Method 1: Find using soup.find() or soup.find_all()

Finding all links on a page using soup.find_all()

Example HTML: Somelink, Otherlinklinks = [link.get("href") for link in soup.find_all("a")]print(links)Result: "example.com", "otherlink.net", etc.

Finding text in first <p> tag using soup.find()

Example HTML: <p>This is not the text you are looking for!</p>paragraph = soup.find("p").textprint(paragraph)Result: "This is not the text you are looking for!"

Method 2: Find using soup.select()

Finding by class

Example HTML: <p class="comment">Text to scrape</p>print(soup.select(".comment")) #make sure you are finding the right informationcomments = [comment.text for comment in soup.select(".comment")] #Get text for all comments on pageprint(comments)Result: "Text to scrape", "other comment", "more comments", "etc" Finding by tags

Example HTML: <p>Random text to find</p>print(soup.select("p > span"))blocks = [block.text for block in soup.select("p > span")]print(blocks)Result: "Random text to find", "another block of text" Finding by id

Example HTML: <p>Comments, yay!</p>soup.select("#comments") OR soup.select("p > #comments")comment_list = [comment.text for comment in soup.select("#comments")]print(comment_list)Result: "comments, yay!", "comments are dumb", "more comments"

Useful Links[Latest Version of Python](#)

[BeautifulSoup4 Documentation](#)

[Requests Documentation](#)

[Lengthy Guide on Web Scraping](#)

[Nachzehrer's Guide on Requests](#)

Actually need this

Kallu609 - June 23, 2019, 8:59 pm

[Beautifulsoup : Is there a difference between .find\(\) and .select\(\) - python 3.xx - Stack Overflow](#)

.select() seems to be more performant than .find() so I suggest going with that if performance is considered.

TGilb94 - June 29, 2019, 4:37 pm

This is great man, thank a lot. Much appreciated!