GENERAL ASSEMBLY

APIS AND SCRAPING

APIS AND SCRAPING

- API Review
- Scraping Review

THE BASIC PROCEDURE

- Using the API docs, get a URL that requests the desired data.
- Try the URL in your browser. Does it return the desired data?
- Request the data in Python using the requests module. Convert the response from JSON to Python data structures.
- Use Python indexing to access the desired data.

WORDNIK EXAMPLE

- Using the API docs, get a URL that requests the desired data.

  http://developer.wordnik.com/docs.html

- Try the URL in your browser. Does it return the desired data?

http://api.wordnik.com/v4/word.json/python/definitions?
limit=200&
includeRelated=true&
useCanonical=false&
includeTags=false&
api_key=a2a73e7b926c924fad7001ca3111acd55

WORDNIK EXAMPLE

- Using the API docs, get a URL that requests the desired data.

  http://developer.wordnik.com/docs.html
- Try the URL in your browser. Does it return the desired data?
- Request the data in Python using the requests module. Convert the response from JSON to Python data structures.

```
import requests
r = requests.get(URL)
word_definition = r.json()
```

WORDNIK EXAMPLE

```
[{'attributionText': 'from The American Heritage® Dictionary of the English
Language, 4th Edition’,
'partOfSpeech': 'noun’,
'text': 'Any of various nonvenomous snakes of the family Pythonidae, found chiefly
in Asia, Africa, and Australia, that coil around and suffocate their prey. Pythons
often attain lengths of 6 meters (20 feet) or more.’,
'word': 'python’,
...
}]
```

a list [] of dictionaries {}

WORDNIK EXAMPLE


```
[{'attributionText': 'from The American Heritage® Dictionary of the English
Language, 4th Edition’,
'partOfSpeech': 'noun’,
'text': 'Any of various nonvenomous snakes of the family Pythonidae, found chiefly
in Asia, Africa, and Australia, that coil around and suffocate their prey. Pythons
often attain lengths of 6 meters (20 feet) or more.’,
'word': 'python’,
...
}]
```


word_definition[0]['text']

a list [] of dictionaries {}

THE BASIC PROCEDURE

- Using the Web Inspector, identify ids or classes that uniquely identify the data to scrape

- Use requests to get the HTML into Python
- Use BeautifulSoup to convert the HTML string into Python data structures
- Get the ids/classes using **select**. Get the text using the property **text**. To get all of the text in all descendents, use **get_text()**.

THE BASIC PROCEDURE

- Using the Web Inspector, identify ids or classes that uniquely identify the data to scrape

  http://www.nasdaq.com/symbol/yhoo/after-hours

  Stock price represented by id = ?

THE BASIC PROCEDURE

- Using the Web Inspector, identify ids or classes that uniquely identify the data to scrape
- Use requests to get the HTML into Python
- Use BeautifulSoup to convert the HTML string into Python data structures

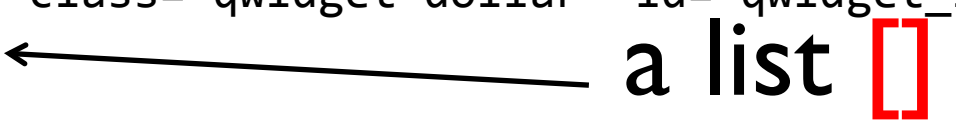
```
import requests
from bs4 import BeautifulSoup

r = requests.get('http://www.nasdaq.com/symbol/yhoo/after-hours’)
soup = BeautifulSoup(r.content)
```

THE BASIC PROCEDURE

- Using the Web Inspector, identify ids or classes that uniquely identify the data to scrape
- Use requests to get the HTML into Python
- Use BeautifulSoup to convert the HTML string into Python data structures
- Get the ids/classes using **select**. Get the text using the property **text**. To get all of the text in all descendents, use **get_text()**.

```
In [15]: soup.select('#qwidget_lastsale')
Out[15]: [<div class="qwidget-dollar" id="qwidget_lastsale">
$40.47</div>]
```



THE BASIC PROCEDURE

- Using the Web Inspector, identify ids or classes that uniquely identify the data to scrape
- Use requests to get the HTML into Python
- Use BeautifulSoup to convert the HTML string into Python data structures
- Get the ids/classes using **select**. Get the text using the property **text**. To get all of the text in all descendents, use **get_text()**.

```
In [15]: soup.select('#qwidget_lastsale')[0].text
Out[15]: '$40.47'
```

OTHER USEFUL BEAUTIFULSOUP METHODS

find_all(id="main-news-story")

find_all(class_="news-story")

- "class_" because "class" is a Python reserved word

get_text()

- concatenates all text nodes (i.e. strips HTML tags)