Data X Berkeley

SOLUTIONS Breakout: Web scraping & web crawling

Author List: Alexander Fred Ojala

Original Sources: https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (https://www.crummy.com/software/BeautifulSoup/bs4/doc/) & https://www.dataquest.io/blog/web-scraping-tutorial-python/ (https://www.dataquest.io/blog/web-scraping-tutorial-python/)

License: Feel free to do whatever you want to with this code

Breakout problem

In this breakout you should extract live weather data in Berkeley from:

http://forecast.weather.gov/MapClick.php?lat=37.87158815800046&lon=-122.27274583799971 (http://forecast.weather.gov/MapClick.php?lat=37.87158815800046&lon=-122.27274583799971)

- Task scrape
  - period / day (as Tonight, Friday, FridayNight etc.)
  - the temperature for the period (as Low, High)
  - the short description (e.g. Mostly Clear, Sunny etc.)
  - the long weather description (e.g. Partly cloudy, with a low around 49..)

Store the scraped data strings in a Pandas DataFrame

Hint: The weather information is found in a div tag with `id='seven-day-forecast'`

The first row of your DataFrame should be similar to the below screenshot (with the same columns):

	day	temp	short_desc	desc
0	Tonight	Low: 46 °F	Mostly Clear	Tonight: Mostly clear, with a low around 46. North wind 6 to 8 mph.

Your solution

In [101]:

```
import requests # The requests library is an
# HTTP library for getting and posting content etc.
import bs4 as bs # BeautifulSoup4 is a Python library
# for pulling data out of HTML and XML code.
# We can query markup languages for specific content
import pandas as pd
```

In [102]:

```
source = requests.get("http://forecast.weather.gov/MapClick.php?lat=37.871588158
00046&lon=-122.27274583799971")
# a GET request will download the HTML webpage.
```

In [103]:

```
# Convert source.content to a beautifulsoup object
# beautifulsoup can parse (extract specific information) HTML code
soup = bs.BeautifulSoup(source.content, features='html.parser')
# we pass in the source content
# features specifies what type of code we are parsing,
# here 'html.parser' specifies that we want beautiful soup to parse HTML code
```

In [104]:

```
seven_day_forecast = soup.find(id='seven-day-forecast')
```

In [105]:

```
arr = []

for ele in seven_day_forecast.find_all(class_="forecast-tombstone"):
    new_arr = []
    for item in ele.find_all('p'):
        img = item.find_all('img')
        if( len(img) != 0):
            new_arr.append(img[0].get('title'))
        else:
            new_arr.append(item.text)
    arr.append(new_arr)
df = pd.DataFrame(arr)
df.rename(columns = {0: 'day', 1: 'desc', 2: 'short-desc', 3: 'temp'}, inplace =
True)
df = df[['day', 'temp', 'short-desc', 'desc']]
df
```

Out[105]:

	day	temp	short-desc	desc
0	Today	High: 62 °F	Sunny	Today: Sunny, with a high near 62. East southe...
1	Tonight	Low: 42 °F	Mostly Clear	Tonight: Mostly clear, with a low around 42. C...
2	Saturday	High: 63 °F	Sunny	Saturday: Sunny, with a high near 63. East nor...
3	SaturdayNight	Low: 44 °F	Mostly Clear	Saturday Night: Mostly clear, with a low aroun...
4	Sunday	High: 63 °F	Mostly Sunny	Sunday: Mostly sunny, with a high near 63. Nor...
5	SundayNight	Low: 44 °F	Partly Cloudy	Sunday Night: Partly cloudy, with a low around...
6	Monday	High: 61 °F	Sunny	Monday: Sunny, with a high near 61.
7	MondayNight	Low: 45 °F	Mostly Clear	Monday Night: Mostly clear, with a low around 45.
8	Tuesday	High: 62 °F	Sunny	Tuesday: Sunny, with a high near 62.

In [ ]:

```

```