NSSA 220
Task Automation with Interpreted Languages

Web Scraping in Python

Instructor: Dr. Fahed Jubair

RIT DUBAI

Review
HTML

- HTML stands for Hyper Text Markup Language
- HTML is the standard language for creating websites
- HTML uses predefined tags to describe elements in a webpage
- To display a website, a browser reads the HTML code to learn how to display the website by relying on the tags inside the code

Review
HTML Example

```
<!DOCTYPE html>
<html>
    <head>
        <title>Page Title</title>
    </head>
    <body>
        <h1>My First Heading</h1>
        <p>My first paragraph.</p>
    </body>
</html>
```

- <!DOCTYPE html> defines an HTML5 document
- <html> defines the root element of an HTML page
- <head> defines the head element
- <title> specifies the title of the page
- <body> defines the body element
- <h1> defines a large heading
- <p> defines a paragraph

Web Scraping

- A web scrapper is a program that extracts information from websites
- A web scrapper typically follows the following steps:
  - Fetch a website (i.e., HTML code)
  - Read the content of the website
  - Extract information of interest
  - Store information in a local file or in a database

Web Scraping in Python

- Several libraries are available in Python for fetching web pages, and for parsing their content
- We will consider the following two modules:
  - Requests module: useful for making HTTP requests to URLs
  - BeautifulSoup module: useful for parsing HTML and XML files and extracting information
- Use the below pip commands to install these modules:

```
pip install requests
pip install bs4
```

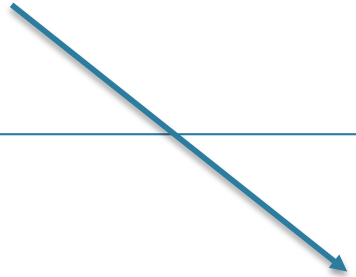
Requests module

- Run the below python program.
- The output is big so redirect the output to a file, called page.html

```
import requests

URL = "https://en.m.wikipedia.org/wiki/Dubai"
page = requests.get(URL)

print(page.text)
```



Note that we use a GET request

BeutifulSoup module

- Run the below python program.

```
from bs4 import BeautifulSoup

f = open('page.html').read()

soup = BeautifulSoup(f, 'html.parser')

print(soup)
print(soup.find('title'))
print(soup.find('title').getText())
print(soup.find('h1'))
print(soup.find('h1').getText())
```

Exercise
Problem Statement

- Visit https://realpython.github.io/fake-jobs/
- The website contains a listing of fake jobs
- Do “right click”, and then choose “view source page” to see the html source code of the website
- Do “right click”, and then choose “inspect” to see the html code corresponding to elements inside the website
- Exercise: write a python scripts that prints all job title names shown on the website

Exercise
Inspecting Website Elements

- Do “right click”, and choose “inspect”
- Try to find which HTML tag corresponds to job titles
- Upon inspection, job title tags are identified by the class “title is-5”
- Now, let use requests module to fetch the website, and use buetifulsoup to extract job title names based on the aforementioned class

Exercise
Writing the script

```
import requests
from bs4 import BeautifulSoup
URL = "https://realpython.github.io/fake-jobs/"
page = requests.get(URL)
soup = BeautifulSoup(page.text,"html.parser")

titles = soup.find_all(class_="title is-5")
for title in titles:
    print(title.getText())
```

Exercise
Your turn

- Modify the previous script to create a dataframe that contains titles, companies, location, and dates for all jobs on the website. Also, write the dataframe to a CSV file.
- Below are the expected first five rows of the dataframe

job title	company	location	date
Senior Python Developer	Payne, Roberts and Davis	Stewartbury, AA	4/8/21
Energy engineer	Vasquez-Davidson	Christopherville, AA	4/8/21
Legal executive	Jackson, Chambers and Levy	Port Ericaburgh, AA	4/8/21
Fitness centre manager	Savage-Bradley	East Seanview, AP	4/8/21
Product manager	Ramirez Inc	North Jamieview, AP	4/8/21
Medical technical officer	Rogers-Yates	Davidville, AP	4/8/21