Web Scrapping

Documentation ~ Nevin Tom 09/03/2024

Task 1

# Web Scraper for Fake Job Listings

This Python script is designed to scrape job listings from the "Fake Jobs" website (<https://realpython.github.io/fake-jobs/>) and save the data to a local file. The script utilizes the **requests** library for making HTTP requests, **BeautifulSoup** for parsing HTML, and performs the scraping and file writing tasks on a recurring basis every 24 hours.

Features

* Scrapes job listings from the "Fake Jobs" website
* Extracts job title, company, and location for each listing
* Saves the extracted data to a local file named **PythonFakeJobs.txt**
* Runs the scraping process every 24 hours in an infinite loop

import requests  
import os  
import time  
  
from bs4 import BeautifulSoup

The code starts by importing the necessary libraries: **requests** for making HTTP requests, **os** for interacting with the operating system, **time** for adding delays, and **BeautifulSoup** from the **bs4** library for web scraping and parsing HTML.

def webScrap(url):  
 response = requests.get(url)  
 if response.status\_code == 200:  
 soup = BeautifulSoup(response.text, 'html.parser')  
 job\_listings = soup.find\_all('div', class\_='card-content')  
 dataToSave = []  
 for job in job\_listings:  
 title\_element = job.find('h2', class\_='title')  
 company\_element = job.find('h3', class\_='company')  
 location\_element = job.find('p', class\_='location')  
 title = title\_element.text.strip() if title\_element else 'No Title'  
 company = company\_element.text.strip() if company\_element else 'No Company'  
 location = location\_element.text.strip() if location\_element else 'No Location'  
 dataToSave.append(f"Job Title: {title}\nCompany: {company}\nLocation: {location}\n---\n")  
 writeToFile(dataToSave)  
 else:  
 print(f"Failed to retrieve data: status code {response.status\_code}")

The **webScrap** function takes a **url** as input and performs the following tasks:

1. It sends a GET request to the provided URL using **requests.get(url)** and stores the response.
2. If the response status code is 200 (OK):
   * It creates a **BeautifulSoup** object from the response text, specifying **'html.parser'** as the parser.
   * It finds all **div** elements with the class **'card-content'**using **soup.find\_all('div', class\_='card-content')**, which are assumed to contain job listings.
   * It initializes an empty list **dataToSave** to store the job data.
   * For each job listing:
     + It finds the **h2** element with the class **'title'**, **h3** element with the class **'company'**, and **p** element with the class **'location'**.
     + It extracts the text from these elements, strips any leading/trailing whitespace, and assigns it to the **title**, **company**, and **location** variables, respectively. If any of these elements are not found, a default value is assigned.
     + It appends a formatted string containing the job title, company, and location to the **dataToSave** list, separated by newlines and a horizontal rule.
   * It calls the **writeToFile** function, passing the **dataToSave** list as an argument.
3. If the response status code is not 200, it prints an error message indicating the failure to retrieve data and the corresponding status code.

def createNameofFile2Save():  
 current\_working\_directory = os.getcwd()  
 fileName = f"{current\_working\_directory}\\PythonFakeJobs.txt"  
 return fileName

The **createNameofFile2Save** function retrieves the current working directory using **os.getcwd()** and constructs a file path by appending **"\\PythonFakeJobs.txt"** to the directory path. It returns this file path as a string.

def writeToFile(data):  
 data = "\n".join(data)  
 with open(createNameofFile2Save(), 'w') as file:  
 file.write(data)  
 print("File created/updated with job data")

The **writeToFile** function takes a list of strings **data** as input and performs the following tasks:

1. It joins the elements of the **data** list using the newline character **'\n'**as the separator, creating a single string.
2. It opens the file specified by the **createNameofFile2Save** function in write mode **('w')** using the **open** function and the **with** statement for automatic resource management.
3. It writes the joined **data** string to the file.
4. It prints a message indicating that the file has been created/updated with job data.

def main():  
 url = 'https://realpython.github.io/fake-jobs/'  
 webScrap(url)  
 while True:  
 time.sleep(86400)  
 webScrap(url)

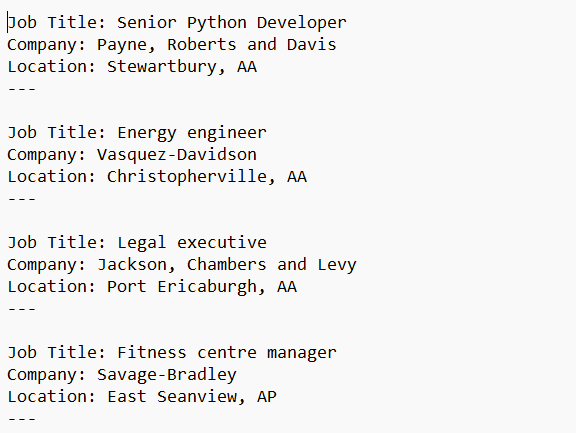
The **main** function is the entry point of the program. It performs the following tasks:

1. It assigns the URL **'https://realpython.github.io/fake-jobs/'** to the **url** variable.
2. It calls the **webScrap** function with the **url** as an argument, scraping the job data and writing it to the file.
3. It enters an infinite loop using **while True**.
4. Inside the loop:
   * It sleeps for 86400 seconds (24 hours) using **time.sleep(86400)**.
   * It calls the **webScrap** function again with the same **url**, repeating the scraping and file writing process.

if \_\_name\_\_ == "\_\_main\_\_":  
 main()

*This line checks if the script is being run as the main program (not imported as a module). If it is the main program, it calls the* ***main*** *function to execute the code.*

# Stored Data in Textfile



In summary, this code scrapes job listings from the **'https://realpython.github.io/fake-jobs/'** website, extracts the job title, company, and location for each listing, and writes the data to a file named **'PythonFakeJobs.txt'** in the current working directory. The scraping process is repeated every 24 hours in an infinite loop.