LinkedIn™ Python Job Posting Scraper

August 2024

CROW WAREZ
www.sosa.tv

Table of Contents

LinkedIn Job Posting Scraper

Summary

This is a basic manual on how you could set up a LinkedIn scraper using Python.

Creating a LinkedIn Job Posting Scraper in Python involves using web scraping techniques to extract data from LinkedIn job listings.

Keep in mind that web scraping LinkedIn is against their terms of service, so it's essential to use this information responsibly and consider using LinkedIn's official API if you need data for legitimate purposes.

I created this guide for educational use, to demonstrate how to extract job listings automatically so they can be imported into Excel, filtered and tracked more efficiently.

Visit www.sosa.tv for information about some data analytics projects I've worked on and other cool stuff.

You'll need the following libraries / packages / modules:

BeautifulSoup

chromedriver_autoinstaller

os (python module)

Pandas

Selenium

What is Python?

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

Install Python

I recommend Anaconda to install Python and packages. This distribution has many nice feats and it makes it very easy to code, debug and manage libraries/packages/modules.

Nevertheless, there are many ways to install Python as shown below.

Step 1- Check if Python is Installed

Before you begin, check if Python is already installed on your computer. Open your command prompt (Windows) or terminal (macOS and Linux) and type the following command:

```
python --version
```

If Python is installed, it will display the version number (e.g., Python 3.9.1). If it's not installed, then go to Step 2 and install Python.

Step 2 - Download Python

I personally use Anaconda, but below you can find a couple of alternatives to download the distribution of your choice.

When choosing the version, it would be best to choose the latest version unless you have a specific reason to download an earlier one. Then download the installer, run it and follow the instructions. If you are not sure as to the selections offered in the installer, just keep it simple and select only the basics as you can.

Python.org
https://www.python.org/downloads/

Anaconda
https://www.anaconda.com/download

Microsoft Store Package
https://apps.microsoft.com/detail/9pjpw5ldxlz5?ocid=webpdpshare

---

Step 3 - Verify Python Installation

Once the installation is complete, open your command prompt or terminal again and type/run the same command on step 1:

```
python --version
```

You should see the installed Python version, if not, try re-installing or choose another distribution.

Step 4 - Install a Code Editor (Optional)

While Python includes the IDLE development environment, many developers prefer using code editors like Visual Studio Code, PyCharm, or Jupyter Notebook (Jupyter is included with Anaconda) for a more robust coding experience. Download and install a code editor of your choice.

Visual Studio Code

Mac, Windows and Linux
https://code.visualstudio.com/download

PyCharm

Mac
https://www.jetbrains.com/pycharm/download/?section=mac

Windows
https://www.jetbrains.com/pycharm/download/?section=windows

Anaconda

Mac, Windows and Linux
https://www.anaconda.com/download

Installing the libraries

I recommend Anaconda to manage these libraries and/or other packages, in my opinion, it's one of the easiest ways to manage Python on your machine. However, if you want to install them via terminal/prompt, you can run these commands (pip package needed, instructions available at the bottom of this section).

Also, JupyterLab is a great notebook to test/debug your code.

It would be best to create a new environment for this project, to avoid any conflicts with other projects you might be working on.

Install Selenium

```
pip install selenium
```

Install chromedriver_autoinstaller

```
pip install chromedriver_autoinstaller
```

Install Pandas

```
pip install pandas
```

Install BeautifulSoup

```
pip install beautifulsoup4
```

Optional:

PIP Installation:

Linux / MacOS

```
python get-pip.py
```

Windows

```
py get-pip.py
```

The Code

Note: A pound sign # means that that line is a comment, thus not run but serving as text for reference, instructions, descriptions, etc.

This script was developed and tested using Jupyter Labs, however it can be used with Visual Studio or other notebook distributions you might find better for you.

The content was separated into parts, it is recommended to test or debug each part before moving to the next one.

Part 1 – Import packages and libraries

Python packages are a set of python modules, while python libraries are a group of python functions aimed to carry out special tasks.

We add the following lines of code at the top of our py file for this:

Code:

```
 # Import necessary packages and libraries
import time
import random
import pandas as pd
from bs4 import BeautifulSoup
from selenium import webdriver
from selenium.webdriver.support.ui import WebDriverWait
```

```
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By
import chromedriver_autoinstaller
```

At the end of this document there is a Glossary with descriptions of what each of these packages/libraries do.

Part 2 – Set up chromedriver

ChromeDriver is a separate executable or a standalone server that WebDriver (Selenium in this case) uses to launch Google Chrome. In our project, WebDriver refers to a collection of APIs used to automate the testing of web applications.

We will use it as an emulator of what you as a Chrome user would do by clicking, copying and pasting the jobs data into a csv file.

It's done by using the code below:

```
# Install the latest chromedriver if necessarychromedriver_autoinstaller.install()

chromedriver_autoinstaller.install()


# Configure Chrome options

options = webdriver.ChromeOptions()

options.add_argument("--start-maximized")


# Launch Chrome browser

browser = webdriver.Chrome(options=options)
```

Part 3 – Open LinkedIn in search mode and set up how many pages to retrieve

The url to be used now, was copied and pasted here, by opening Chrome (without logging in to LinkedIn), search for the keywords “VBA Analyst” and “Toronto” as the location. You can modify keywords (job and location) as needed.

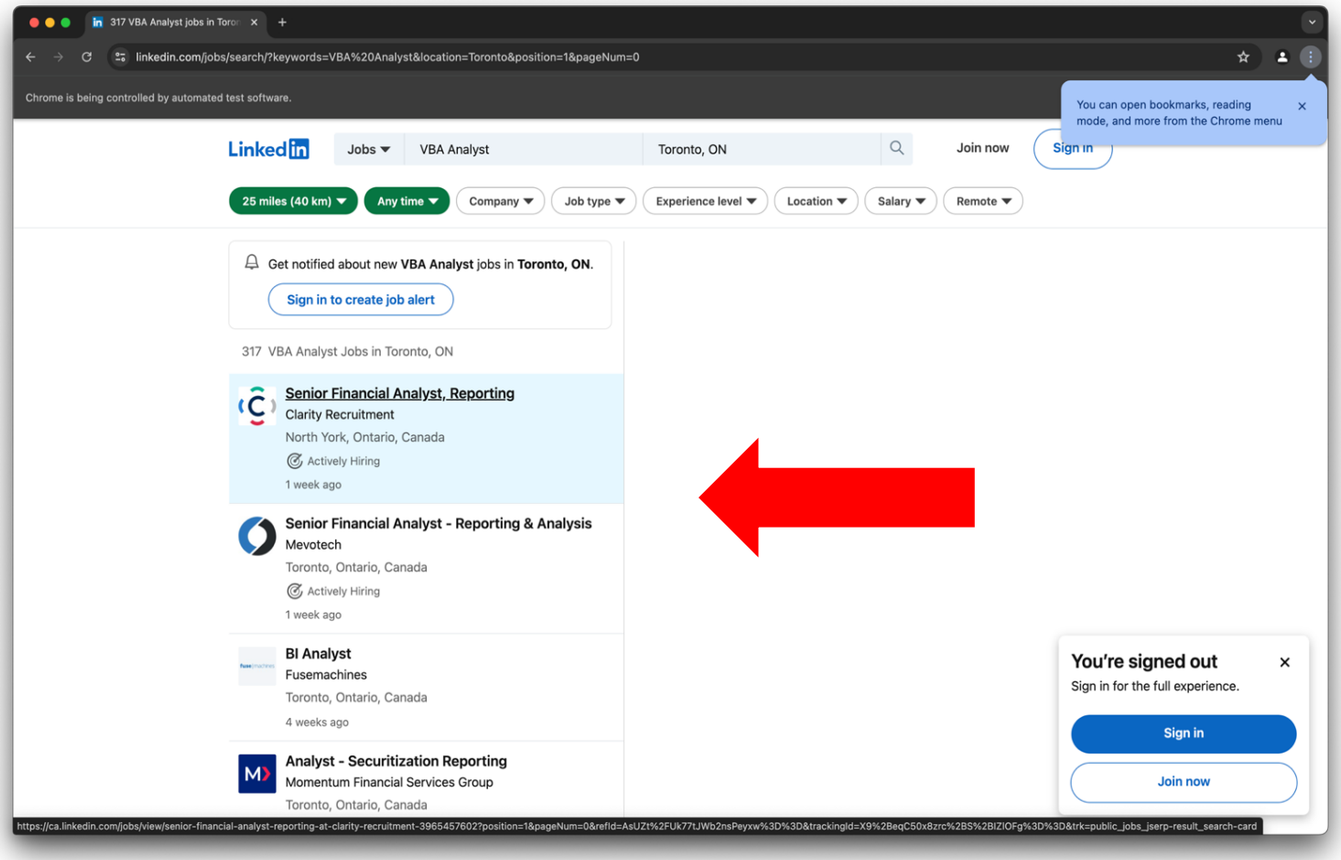
Code:

```
# Open LinkedIn job search page (modify keywords as needed)

browser.get(f'https://www.linkedin.com/jobs/search/?keywords=Business%20Analyst&location=Toronto&position=1&pageNum=0')


# Set the number of pages to scrape

pages = 10
```



Part 4 - Loop through the number of pages to retrieve postings

When the results are displayed at the left side of the browser (as of July 2024), at the very bottom there is a button with the likes of *"Show More Jobs"*, each time the button appears will be a page, resulting in the number of pages we will be scraping.

Code:

```
# Loop through the specified number of pages to retrieve job postings
for i in range(pages):
    print(f'Scraping page {i + 1}')
    browser.execute_script("window.scrollTo(0, document.body.scrollHeight);")

    try:
        # Click on the "see more jobs" button if present
        element = WebDriverWait(browser, 5).until(
            EC.presence_of_element_located(
                (By.XPATH, "//button[text()='See more jobs']")
            )
        )
        element.click()
    except Exception:
        pass
```

Part 5 - # Scrape!

In this section we'll use BeautifulSoup to parse the html of the website so we can find elements more efficiently.

Code:

```
# Scrape job postings

jobs = []

soup = BeautifulSoup(browser.page_source, "html.parser")

job_listings = soup.find_all("div", class_="base-card")


for job in job_listings:


    job_title = job.find("h3", class_="base-search-card__title").text.strip()

    job_company = job.find("h4", class_="base-search-card__subtitle").text.strip()

    job_location = job.find("span", class_="job-search-card__location").text.strip()

    apply_link = job.find("a", class_="base-card__full-link")["href"]

    job_ID = apply_link.split('?')[0][-10:]


    browser.get(apply_link)
```

```
time.sleep(random.choice(range(5, 11)))


try:
    description_soup = BeautifulSoup(browser.page_source, "html.parser")
    job_description = description_soup.find("div", class_="jobs-apply-button--top-card").text.strip()
except AttributeError:
    job_description = None


jobs.append({
    "job ID": job_ID,
    "title": job_title,
    "company": job_company,
    "location": job_location,
    "link": apply_link,
    "job description": job_description
})
```

Part 6- Save data into a csv file, exclude index column

Code:

```
# Save data into a CSV file

df = pd.DataFrame(jobs)

df.to_csv("jobs.csv", index=False)
```

Glossary

Anaconda

Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS.

The Anaconda distribution can be downloaded for free here: https://www.anaconda.com/download

API

An API, short for “application programming interface,” facilitates communication between two computer systems. Let's consider a mobile weather application that provides users with real-time weather updates.

BeautifulSoup

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.

Documentation: https://beautiful-soup-4.readthedocs.io/en/latest/

Chromedriver

ChromeDriver is a standalone server that implements the W3C WebDriver standard. WebDriver is an open source tool built for automated testing of web apps across many browsers. Its interface allows for control and introspection of user agents locally or remotely using capabilities.

Documentation: https://developer.chrome.com/docs/chromedriver

chromedriver_autoinstaller

A library that automatically downloads the version of ChromeDriver compatible with the client's version of Chrome. Supports all versions of Chrome.

It can be installed by running in command line:

```
pip install chromedriver_autoinstaller
```

Project site: https://pypi.org/project/chromedriver-autoinstaller/

Command Prompt

Command Prompt, also known as cmd.exe or cmd, is the default command-line interpreter for the OS/2, eComStation, ArcaOS, Microsoft Windows, and ReactOS operating systems. On Windows CE .NET 4.2, Windows CE 5.0 and Windows Embedded CE 6.0 it is referred to as the Command Processor Shell.

Commands are the instructions we give to the application to execute certain tasks.

In Unix / MacOS the command prompt is called "Terminal", "Command", "Prompt" or "Shell", depending on the OS you are running. In my case I use MacOS and the app is called "Terminal".

Debugging

The process of identifying and removing errors from a program. This can involve using tools like print statements, logging, or interactive debuggers.

Distribution

A Python distribution is an archive file that contains one or more Python package. The distribution file is what the end-user will download and install. It's typically just one package. There are two common types of distributions: Source distributions and the wheel binary format.

Elements

An HTML element is a type of HTML document component, one of several types of HTML nodes. The first used version of HTML was written by Tim Berners-Lee in 1993 and there have since been many versions of HTML.

Exception

An exception is an unexpected behavior (wrong or not) that occurs during software execution. This can interrupt the normal flow of execution and needs proper handling. These “unexpected behaviors” may be in methods of your own software or third-party libraries/components.

Indentation

In the written form of many languages, indentation describes empty space, a.k.a. white space, used around text to signify an important aspect of the text such as: Beginning of a paragraph Hierarchy – subordinate concept Quotation Many computer languages use block indentation to demarcate blocks of source code.

JupyterLab

JupyterLab is a web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality.

Documentation: https://jupyterlab.readthedocs.io/en/latest/

Library

Collections of pre-written code and functions that extend the capabilities of the Python programming language. They provide a wide range of tools and modules for various tasks, making it easier for developers to work on specific tasks without reinventing the wheel.

Module

A module is a file containing Python definitions and statements. The file name is the module name with the suffix .py appended. Within a module, the module's name (as a string) is available as the value of the global variable __name__ .

os (python module)

The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The *os* and *os. path* modules include many functions to interact with the file system.

Documentation: https://docs.python.org/3/library/os.html

Package

Container for storing multiple Python modules. We can install packages in Python using the pip package manager.

Pandas

Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks, in this project, it is used to save the scraped data into a csv file.

Documentation: https://pandas.pydata.org/docs/

Parsing

It means to resolve (a sentence) into its component parts and describe their syntactic roles or simply it is an act of parsing a string or a text.

PIP

The standard package manager for Python. It allows you to install and manage packages that aren't part of the Python standard library.

Documentation: https://pip.pypa.io/en/stable/

PyCharm

PyCharm is an integrated development environment used for programming in Python. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems, and supports web development with Django. PyCharm is developed by the Czech company JetBrains.

Documentation: https://www.jetbrains.com/help/pycharm/getting-started.html

Python

Python is an open-source programming language, having features like object-oriented, interpreted and high-level too. It is a dynamically typed programming language, which is easy to use with redable and user-friendly syntax. It has huge libraries, frameworks and large community support.

More documentation: https://www.python.org/doc/essays/blurb/

Selenium

Selenium with Python is used to carry out automated test cases for browsers or web applications. You can easily use it to simulate tests such as tapping on a button, entering content to the structures, skimming the entire site, etc.

Documentation: https://www.selenium.dev/documentation/

Visual Studio

Visual Studio is an integrated development environment developed by Microsoft. It is used to develop computer programs including websites, web apps, web services and mobile apps.

Documentation: https://learn.microsoft.com/en-us/visualstudio/

WebDriver

Selenium WebDriver is a web framework that permits you to execute cross-browser tests. This tool is used for automating web-based application testing to verify that it performs expectedly. Selenium WebDriver allows you to choose a programming language to create test scripts.

Documentation: https://www.selenium.dev/documentation/webdriver/

Web Scraping

Process of using bots to extract content and data from a website. Unlike screen scraping, which only copies pixels displayed onscreen, web scraping extracts underlying HTML code and, with it, data stored in a database.