# Twitter and Facebook Scrapping

First I begun by installing Twint to use to scrap Twitter, the following document will explain the steps I took and the results I received.

## Twitter

### How to Install and Use Twint

**There are different websites that explain how to install and use Twint. I have mentioned those websites along with how to install it in case you are using Mac M1 just like me.**

- **Github Twint repository: https://github.com/twintproject/twint**
- **How to Use Twint to Scrape Twitter Data: A Step-by-Step Tutorial: https://aitechtrend.com/how-to-use-twint-to-scrape-twitter-data-a-step-by-step-tutorial/**
- ***Twint: Twitter Scraping Without Twitter's API: https://basilkjose.medium.com/twint-twitter-scraping-without-twitters-api-aca8ba1b210e***

### Installing Twint on Mac M1

1. **To install Twint on Mac M1, first open the terminal and run:**

```
pip3 install twint
```

OR

```
pip3 install --user --upgrade git+https://github.com/twintproject/twint.git@origin/master#egg=twint
```

*OR*

```
pip install --user git+https://github.com/woluxwolu/twint.git
```

2. The expected error you will encounter is:

```
WARNING: Skipping /opt/homebrew/lib/python3.11/site-packages/numpy-1.25.1-py3.11.egg-info due to invalid metadata entry 'name'
```

```
WARNING: Skipping
/opt/homebrew/lib/python3.11/site-packages/numpy-1.25.1-py3.11.egg-info due to
invalid metadata entry 'name'
Collecting twint
  Cloning https://github.com/twintproject/twint.git (to revision origin/master) to
/private/var/folders/jy/jw39h2vx0cb_5g4scvrzm7p00000gn/T/pip-install-d5p2vkq1/twint_f
cbe5de9bb794fd5abff90e4ef184274
  Running command git clone --filter=blob:none --quiet
https://github.com/twintproject/twint.git
/private/var/folders/jy/jw39h2vx0cb_5g4scvrzm7p00000gn/T/pip-install-d5p2vkq1/twint_f
cbe5de9bb794fd5abff90e4ef184274
  WARNING: Did not find branch or tag 'origin/master', assuming revision or ref.
  Running command git checkout -q origin/master
  Resolved https://github.com/twintproject/twint.git to commit origin/master
  Preparing metadata (setup.py) ... done
Requirement already satisfied: aiohttp in /opt/homebrew/lib/python3.11/site-packages
(from twint) (3.8.4)
Collecting aiodns (from twint)
  Using cached aiodns-3.0.0-py3-none-any.whl (5.0 kB)
Requirement already satisfied: beautifulsoup4 in
/opt/homebrew/lib/python3.11/site-packages (from twint) (4.12.0)
Collecting cchardet (from twint)
  Using cached cchardet-2.1.7.tar.gz (653 kB)
  Preparing metadata (setup.py) ... done
Collecting dataclasses (from twint)
  Using cached dataclasses-0.6-py3-none-any.whl (14 kB)
Collecting elasticsearch (from twint)
  Obtaining dependency information for elasticsearch from
https://files.pythonhosted.org/packages/1d/8f/cc916b381db044072966510b4c4472773e
e60dc6589e6f8856eaecfe4aa6/elasticsearch-8.8.2-py3-none-any.whl.metadata
  Using cached elasticsearch-8.8.2-py3-none-any.whl.metadata (4.9 kB)
Collecting pysocks (from twint)
  Using cached PySocks-1.7.1-py3-none-any.whl (16 kB)
Requirement already satisfied: pandas in /opt/homebrew/lib/python3.11/site-packages
(from twint) (1.5.3)
Collecting aiohttp_socks (from twint)
  Using cached aiohttp_socks-0.8.0-py3-none-any.whl (9.4 kB)
Collecting schedule (from twint)
  Using cached schedule-1.2.0-py2.py3-none-any.whl (11 kB)
Collecting geopy (from twint)
  Using cached geopy-2.3.0-py3-none-any.whl (119 kB)
Requirement already satisfied: fake-useragent in
/opt/homebrew/lib/python3.11/site-packages (from twint) (1.1.3)
Collecting googletransx (from twint)
```

```
  Using cached googletransx-2.4.2-py3-none-any.whl
Collecting pycares>=4.0.0 (from aiodns->twint)
  Using cached pycares-4.3.0-cp311-cp311-macosx_10_9_universal2.whl (136 kB)
Requirement already satisfied: attrs>=17.3.0 in
/opt/homebrew/lib/python3.11/site-packages (from aiohttp->twint) (23.1.0)
Requirement already satisfied: charset-normalizer<4.0,>=2.0 in
/opt/homebrew/lib/python3.11/site-packages (from aiohttp->twint) (3.1.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
/opt/homebrew/lib/python3.11/site-packages (from aiohttp->twint) (6.0.4)
Requirement already satisfied: async-timeout<5.0,>=4.0.0a3 in
/opt/homebrew/lib/python3.11/site-packages (from aiohttp->twint) (4.0.2)
Requirement already satisfied: yarl<2.0,>=1.0 in
/opt/homebrew/lib/python3.11/site-packages (from aiohttp->twint) (1.9.2)
Requirement already satisfied: frozenlist>=1.1.1 in
/opt/homebrew/lib/python3.11/site-packages (from aiohttp->twint) (1.4.0)
Requirement already satisfied: aiosignal>=1.1.2 in
/opt/homebrew/lib/python3.11/site-packages (from aiohttp->twint) (1.3.1)
Requirement already satisfied: python-socks[asyncio]<3.0.0,>=2.0.0 in
/opt/homebrew/lib/python3.11/site-packages (from aiohttp_socks->twint) (2.3.0)
Requirement already satisfied: soupsieve>1.2 in
/opt/homebrew/lib/python3.11/site-packages (from beautifulsoup4->twint) (2.4)
Collecting elastic-transport<9,>=8 (from elasticsearch->twint)
  Using cached elastic_transport-8.4.0-py3-none-any.whl (59 kB)
Collecting geographiclib<3,>=1.52 (from geopy->twint)
  Using cached geographiclib-2.0-py3-none-any.whl (40 kB)
Requirement already satisfied: requests in /opt/homebrew/lib/python3.11/site-packages
(from googletransx->twint) (2.31.0)
Requirement already satisfied: python-dateutil>=2.8.1 in
./Library/Python/3.11/lib/python/site-packages (from pandas->twint) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
/opt/homebrew/lib/python3.11/site-packages (from pandas->twint) (2023.3)
Requirement already satisfied: numpy>=1.21.0 in
/opt/homebrew/lib/python3.11/site-packages (from pandas->twint) (1.24.2)
Requirement already satisfied: urllib3<2,>=1.26.2 in
/opt/homebrew/lib/python3.11/site-packages (from
elastic-transport<9,>=8->elasticsearch->twint) (1.26.16)
Requirement already satisfied: certifi in /opt/homebrew/lib/python3.11/site-packages
(from elastic-transport<9,>=8->elasticsearch->twint) (2023.5.7)
Collecting cffi>=1.5.0 (from pycares>=4.0.0->aiodns->twint)
  Using cached cffi-1.15.1-cp311-cp311-macosx_11_0_arm64.whl (174 kB)
Requirement already satisfied: six>=1.5 in
./Library/Python/3.11/lib/python/site-packages (from
python-dateutil>=2.8.1->pandas->twint) (1.16.0)
Requirement already satisfied: idna>=2.0 in
/opt/homebrew/lib/python3.11/site-packages (from yarl<2.0,>=1.0->aiohttp->twint) (3.4)
```

```
Collecting pycparser (from cffi>=1.5.0->pycares>=4.0.0->aiodns->twint)
  Using cached pycparser-2.21-py2.py3-none-any.whl (118 kB)
Using cached elasticsearch-8.8.2-py3-none-any.whl (393 kB)
Building wheels for collected packages: twint, cchardet
  Building wheel for twint (setup.py) ... done
  Created wheel for twint: filename=twint-2.1.21-py3-none-any.whl size=38850
sha256=853a5fccbeb7cda7558daf97232f9a316d698ae2d5b1a39ff4ef4d358d3f8d94
  Stored in directory:
/private/var/folders/jy/jw39h2vx0cb_5g4scvrzm7p00000gn/T/pip-ephem-wheel-cache-dj
3406yc/wheels/2f/b4/c2/62cb9f848b3c730dfe970633d17fd0c92f386766d32253ef25
  Building wheel for cchardet (setup.py) ... error
  error: subprocess-exited-with-error

  × python setup.py bdist_wheel did not run successfully.
  │ exit code: 1
  ╰─> [23 lines of output]
      running bdist_wheel
      running build
      running build_py
      creating build
      creating build/lib.macosx-13-arm64-cpython-311
      creating build/lib.macosx-13-arm64-cpython-311/cchardet
      copying src/cchardet/version.py -> build/lib.macosx-13-arm64-cpython-311/cchardet
      copying src/cchardet/__init__.py ->
build/lib.macosx-13-arm64-cpython-311/cchardet
      running build_ext
      building 'cchardet._cchardet' extension
      creating build/temp.macosx-13-arm64-cpython-311
      creating build/temp.macosx-13-arm64-cpython-311/src
      creating build/temp.macosx-13-arm64-cpython-311/src/cchardet
      creating build/temp.macosx-13-arm64-cpython-311/src/ext
      creating build/temp.macosx-13-arm64-cpython-311/src/ext/uchardet
      creating build/temp.macosx-13-arm64-cpython-311/src/ext/uchardet/src
      creating build/temp.macosx-13-arm64-cpython-311/src/ext/uchardet/src/LangModels
      clang -Wsign-compare -Wunreachable-code -fno-common -dynamic -DNDEBUG -g
-fwrapv -O3 -Wall -isysroot
/Library/Developer/CommandLineTools/SDKs/MacOSX13.sdk -Isrc/ext/uchardet/src
-I/opt/homebrew/opt/python@3.11/Frameworks/Python.framework/Versions/3.11/include
/python3.11 -c src/cchardet/_cchardet.cpp -o
build/temp.macosx-13-arm64-cpython-311/src/cchardet/_cchardet.o
      src/cchardet/_cchardet.cpp:196:12: fatal error: 'longintrepr.h' file not found
        #include "longintrepr.h"
                 ^~~~~~~~~~~~~~~
      1 error generated.
```

```
    error: command '/usr/bin/clang' failed with exit code 1
    [end of output]

  note: This error originates from a subprocess, and is likely not a problem with pip.
  ERROR: Failed building wheel for cchardet
  Running setup.py clean for cchardet
Successfully built twint
Failed to build cchardet
ERROR: Could not build wheels for cchardet, which is required to install
pyproject.toml-based projects
WARNING: Skipping
/opt/homebrew/lib/python3.11/site-packages/numpy-1.25.1-py3.11.egg-info due to
invalid metadata entry 'name'
WARNING: Skipping
/opt/homebrew/lib/python3.11/site-packages/numpy-1.25.1-py3.11.egg-info due to
invalid metadata entry 'name'
WARNING: Skipping
/opt/homebrew/lib/python3.11/site-packages/numpy-1.25.1-py3.11.egg-info due to
invalid metadata entry 'name'
```

3. To solve this error, install the following libraries:

```
pip install cython
```

then

```
pip install datadotworld
```

*Link:*
*https://stackoverflow.com/questions/55164178/failed-building-wheel-for-cchardet-when-installing-datadotworld*

*4. Now rerun:*

```
pip3 install twint
```

OR

```
pip3 install --user --upgrade
git+https://github.com/twintproject/twint.git@origin/master#egg=twint
```

*5.* Open your .zshrc add this and safe:

```
export PYTHON_BIN_PATH="$(python3 -m site --user-base)/bin"
export PATH="$PATH:$PYTHON_BIN_PATH"
```

***This should solve the error, however, seeing as how twitter has currently closed all APIs for the time being, Twitter is no longer a viable option for Scarpping. The next social media plateform I scrapped was Facebook.***

# Facebook

## How to Install and Use Facebook

**There are different websites that explain how to install and use Facebook Post Scraper. I have mentioned those websites along with how to install it in case you are using Mac M1 just like me.**

- **Github repository:** *https://github.com/brutalsavage/facebook-post-scraper*
- **How to Scrape Data from Facebook Accounts | Python Tutorial:** *https://youtu.be/WnFAaWYecJs*

## Installing Facebook Scrapper on Mac M1

1. **To install Facebook Post Scrapper on Mac M1, first open the terminal and run:**

```
pip3 install facebook_page_scraper
```

2. Also run the following code in the terminal:

```
brew install geckodriver
```

in case you do not have Hebrew installed, install it and add it to the .zhsrc file as a path.

3. Download Firefox.

## Code for Scrapping of Single Page

This code is the same as the single page code except it uses a for loop to iterate through the pages list.

```
## Import
from facebook_page_scraper import Facebook_scraper
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC


# Instantiate the Facebook_scraper class

posts_count = 5000
```

```
browser = "firefox"
proxy = "IP:PORT" #if proxy requires authentication then user:password@IP:PORT
timeout = 10000 #600 seconds
headless = True

# Dir for output if we scrape directly to CSV
# Make sure to create this folder

directory = "/Users/izzymohamed/Desktop/Scrapping"  #TODO: CHANGE DIRECTORY


## Single Scrapping
page_name = "metaai"
# initializing a scraper

scraper = Facebook_scraper(page_name, posts_count, browser, proxy=proxy,
timeout=timeout, headless=headless)
# Running the scraper in two ways:
# 1. Scraping and printing out the result into the console window:

# json_data = scraper.scrap_to_json()
# print(json_data)

# 2. Scraping and writing into output CSV file:

filename = page_name

scraper.scrap_to_csv(filename, directory)
```

## Code for Scrapping of Multiple Pages

```
## Import
from facebook_page_scraper import Facebook_scraper
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
# Instantiate the Facebook_scraper class

posts_count = 5000
browser = "firefox"
proxy = "IP:PORT" #if proxy requires authentication then user:password@IP:PORT
timeout = 600000 #600 seconds
```

```
headless = True

# Dir for output if we scrape directly to CSV
# Make sure to create this folder

directory = "/Users/izzymohamed/Desktop/Scrapping"  #TODO: CHANGE DIRECTORY

## Multipage Scrapping
To use the following code, changes will need to be made in the
facebook_page_scrapper library.
# instantiate the Facebook_scraper class

page_list =
['openai.research','metaai','GitHubCopilot','huggingface','profile.php?id=1000946814851
85','master.of.code.global']

for page in page_list:
    # our proxy for this scrape

    ## proxy = f'username:password@us.smartproxy.com:{proxy_port}'

    # initializing a scraper

    scraper = Facebook_scraper(page, posts_count, browser, proxy=proxy,
timeout=timeout, headless=headless)

    # Running the scraper in two ways:
    # 1. Scraping and printing out the result into the console window:

    # json_data = scraper.scrap_to_json()
    # print(json_data)

    # 2. Scraping and writing into output CSV file:

    filename = page
    scraper.scrap_to_csv(filename, directory)
```