

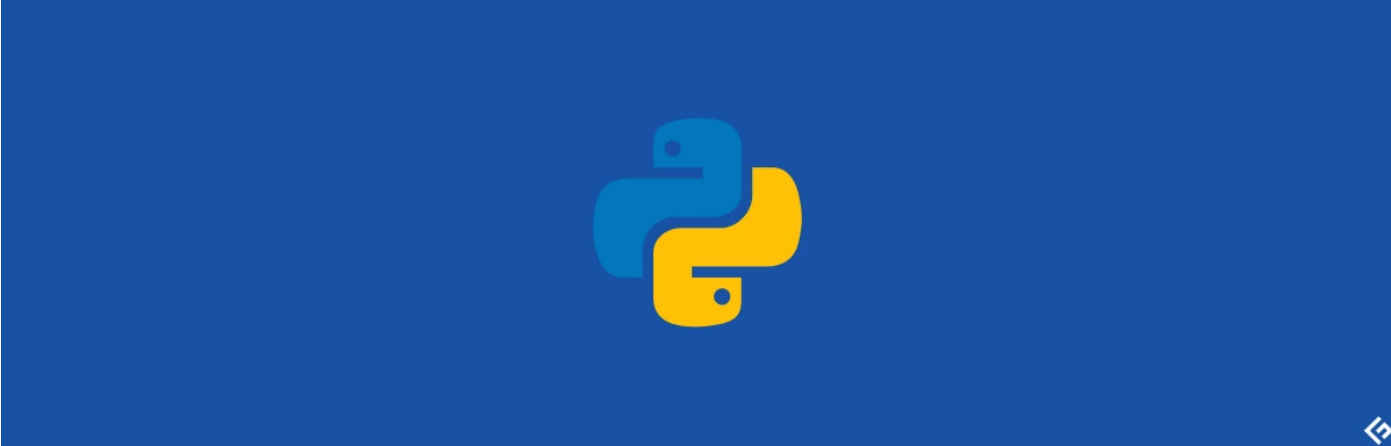
11 Python Libraries and Modules Every Developer Should Know



By **Hafeezul Kareem Shaik** in **Development** on December 15, 2020



Geekflare is supported by our audience. We may earn affiliate commissions from buying links on this site.



Libraries and Modules make the life of a programmer smooth.

When you are working with projects, you may encounter scenarios where you won't be able to solve with the standard coding of a programming language. We need some libraries and modules to overcome those problems.

Luckily Python supports a plethora of modules and libraries. Python has **built-in** modules as well as **third-party** libraries and modules for the development. We will see both integrated and third-party modules, which are very beneficial for Python projects. Let's explore the built-in modules first.

Built-in Modules

Python comes with plenty of built-in modules for different use cases. We will study the modules one by one according to the usage.

1 Collections – Container datatypes

Python has different types of **collections** to store the collection of data. For example, **tuple**, **list**, **dict**, **etc..**, are some of the built-in collections of Python. The **collections** module provides additional features to the built-in collections.

If you take **deque** data collection from the **collections** module, it more like a Python list. But, we can **push** and **pop** the elements from both sides. It's faster than the **list**. You can use the **deque** based on your needs. Let's see some real coding with **collections.deque** data collection.

```
import collections
nums = [1, 2, 3]
# creating deque collection from the list
deque = collections.deque(nums)

print(deque)

# adding an element at the end
deque.append(4)

print(deque)

# adding element at the starting
deque.appendleft(0)

print(deque)

# removing the element at the end
deque.pop()

print(deque)

# removing element at the starting
deque.popleft()

print(deque)
```

Run the above code; see the results. We have other data collections as well in the **collections** module.

Some of them are:

| | |
|---------|--|
| Counter | Returns a dict which contains the frequency of elements from the list. |
| | It's a subclass of the dict class. |

| | |
|------------|--|
| UserDict | Used for a quick subclass of the dict. |
| UserString | Used for a quick subclass of the str. |

Go to the documentation of the **collections** module to explore all the data collections and methods.

Quick Note:- Use the **dir(object)** built-in method of Python to see all the methods of an object.

2 CSV – file Handling

We can use the **CSV** (comma-separated values) files to store the tabular data. The most commonly used format for importing and exporting the data from spreadsheets and databases. Python comes with a module called CSV to handle the CSV files.

Let’s see one example of reading the data from a CSV file.

Create a file with the name **sample.csv** in your laptop and paste the following data.

| Name, Age, Graduation Year |
|----------------------------|
| Hafeez, 21, 2021 |
| Aslan, 23, 2019 |
| Rambabu, 21, 2021 |

We have methods to read and write in the CSV module. We will see how to read the data from the CSV files using the CSV module.

```
import csv

with open('sample.csv') as file:
    # creating the reader
    reader = csv.reader(file)

    # reading line by line using loop
    for row in reader:
        # row is a list containing elements from the CSV file
        # joining the list using join(list) method
        print(','.join(row))
```

Run the above code to see the results.

We will also have an object called `csv.writer()` to write the data into the **CSV** file. Play with the other methods on your own using the `dir()` and `help()` built-in methods. We have another module called **JSON**, which is used for handling the **JSON** files. It’s also a built-in module.

3 Random – generation

Python has a module called **random** that allows generating the data randomly. We can produce anything randomly using different ways of the **random** module. You can use this module in applications like tic-tac-toe, a dice game, etc.,

Let’s see a simple program to generate random integers from a given range.

```
import random

# generating a random number from the range 1-100
print(random.randint(1, 100))
```

You check the other methods of the **random** module using `dir()` and `help()` methods. Let’s write a small and simple game using the **random** module. We can call it a **Number Guessing Game**.

What is the Number Guessing Game?

The program will generate a random number in the range of 1 – 100. The user will guess the number until it matches the random number generated by the program. Every time you will print whether the user number is less than the random number or higher than the random number. Then, the source code will display the number of guesses.

See the below code for the above program.

```
# importing random module
import random

# generating random number
random_number = random.randint(1, 100)

# initializing no. of guess to 0
guess_count = 0

# running loop until user guess the random number
```

```
        user_guessed_number = int(input("Enter a number in the range of 1-100\n"))\n\n        # checking for the equality\n        if user_guessed_number == random_number:\n            print(f"You have guessed the number in {guess_count} guesses")\n            # breaking the loop\n            break\n        elif user_guessed_number < random_number:\n            print("Your number is low")\n        elif user_guessed_number > random_number:\n            print("Your number is high")\n\n        # incrementing the guess count\n        guess_count += 1
```

4 Tkinter – GUI applications

Tkinter is a built-in module for the development of **GUI (Graphical User Interface)** applications. It is convenient for beginners. We can develop **GUI** applications like **calculator, login system, text editor, etc.,** There are many resources out there to learn the **GUI** development with **Tkinter**.

The best support is to follow the official **docs**. To get started with the **Tkinter**, go to the docs and start creating beautiful **GUI** applications.

Third-party Modules

5 Requests – HTTP requests

Requests module is used to send all kinds of **HTTP** requests to the server. It allows **HTTP/1.1** requests to send. We can also add headers, data, and other things using Python dictionaries. As it is a third-party module, we have to install it. Run the following command in the terminal or command-line to install the **requests** module.

```
pip install requests
```

It's straightforward to work with the **requests** module. We can start working with the **requests** without any prior knowledge. Let's see how to send a get request and what it returns.

```
import requests\n\n# sening a get request\nrequest = requests.get("https://www.google.com/")\n\n#\nprint(request.status_code)\nprint(request.url)\nprint(request.request)
```

The above code will print the status_code, URL, and request method (GET, POST). You will get the source of the **URL** as well. You can access it with the **request.content** bytes. Go to the **docs** of the **requests** module and explore more.

6 BeautifulSoup4 – web scraping

BeautifulSoup library is used for the web scraping. It's a handy module to work with. Even beginners can start working with it using the **docs**. See the sample code to scrap the customer reports details.

You can install **BeautifulSoup** by typing the following command in the terminal/command-line.

```
pip install beautifulsoup4
```

And, a simple program for your first scraping.

```
## Scrping the ConsumerReport products list using BeautifulSoup\n\n## importing bs4, requests modules\nimport bs4\nimport requests\n\n## initializing url\nurl = "https://www.consumerreports.org/cro/a-to-z-index/products/index.htm"\n\n## getting the reponse from the page using get method of requests module\npage = requests.get(url)\n\n## storing the content of the page in a variable\nhtml = page.content
```

```
## see the class or id of the tag which contains names ans links
div_class = "crux-body-copy"

## getting all the divs using find_all method
div_tags = soup.find_all("div", class_=div_class) ## finding divs whichs

## we will see all the tags with a tags which has name and link inside tl
for tag in div_tags:
    print(tag)
```

Run the above code to see the magic of web scraping. There are more [web scraping frameworks](#) out there for you to try.

Data Science and Machine Learning

There are some libraries out there specially created for data science and machine learning. All these are developed in **C**. They are lightning-fast.

7 Numpy

Numpy is used for scientific computation.

It allows us to work multidimensional arrays. Arrays implementation is not there in Python. Mainly the developers use **numpy** in their machine learning projects. It's easy to learn and open-source library. Almost every machine learning engineer or data scientist uses this module for complex mathematical computations.

Run the following command to install the **numpy** module.

```
pip install numpy
```

8 Pandas

Pandas is a data analysis module. We can filter the data most effectively using the **pandas** library. It offers different types of data structures that are handy to work. It also provides file handling with different file formats.

Install the module using the following command.

```
pip install pandas
```

9 Matplotlib

Matplotlib is a 2D graph plotting library. You can visualize the data using **Matplotlib**.

We can generate images of the figures in different formats. We plot different types of diagrams like bar charts, error charts, histograms, scatterplots, etc., You can install the **matplotlib** using the following command.

```
pip install matplotlib
```

Quick Note:- You can install **Anaconda** to get all the libraries and modules required for Data Science.

If you are serious about learning Python for data science and ML then check out this brilliant [Udemy course](#).

Web Frameworks

We can find many web frameworks in *Python*. We will discuss two frameworks that are widely used by the developers. The two frameworks are **Django** and **Flask**.

10 Django

Django is an open-source web framework developed in Python. It is convenient to create websites with **Django**. We can generate any kinds of sites using this framework. Some of the most popular sites built with Django are **Instagram, bitbucket, Disqus, Mozilla Firefox, etc.,**

- We can build complex websites quickly with the features of Django.
- The Django already does a lot of the tasks required for web development.
- It also provides security for the attacks **SQL Injection, cross-site scripting, cross-site request forgery, and clickjacking.**
- We can build any website from the content management system to social sites.

The documentation of Django is unambiguous. You have to familiar with the Python for Django. But don't worry if you're not. [Learning Django](#) is easy.

Flask is a micro web framework developed in Python.

It is more pythonic than Django. It has excellent documentation [here](#). It uses the **Jinja** template engine. It is very complex to create big websites Flask. Most of the features like URL routing, Request dispatching, Secure cookies, Sessions, etc., are present in both **Django** and **Flask**.

Choose the framework based on the complexity of your website. Django is gaining popularity among developers. It's the most used framework for web development in Python.

Conclusion

I hope you got to know about different modules, libraries, and frameworks for Python.


Everyone once a beginner.

Whatever you want to start, first go the documentation and start learning it. If you can't understand the docs, then find crash courses on the [educational websites](#).


Enjoyed reading the article? How about sharing with the world? [🐦](#) [f](#) [in](#) [💬](#)

Tagged in: Python

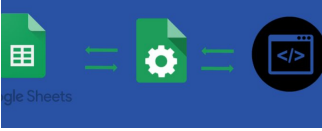
More great readings on Development

- 


Implementing HTML Editor Within Your Apps is Easy With Froala – Next Generation WYSIWYG Editor

Bipasha Nath on April 28, 2022
- 


Python Programs on String Operations

Bala Priya C on April 28, 2022
- 


10 Tools to Turn Your Google Sheets Into an API

Shalabh Garg on April 22, 2022
- 

10 Best API Development and Testing Tools

Durga Prasad Acharya on April 21, 2022
- 

NumPy reshape(): How to Reshape NumPy Arrays in Python

Bala Priya C on April 26, 2022
- 

9 Best WYSIWYG Editor to Integrate Into Your Application [Developer-Friendly]

Durga Prasad Acharya on April 19, 2022

Join Geekflare Newsletter

All

▼

Join Newsletter

| | | |
|---------------------|---------------|--------------------|
| PRODUCTS | COMPANY | GEEKFLARE ARTICLES |
| Geekflare API | Advertise | DevOps |
| Geekflare Tools | About | Cloud Computing |
| Tech Articles | Terms | Security |
| Finance Articles | Privacy | Sysadmin |
| Compiler | Disclosure | Development |
| | Sitemap | Smart Things |
| | RSS Feed | Growing Business |
| GEEKFLARE TOOLS | GEEKFLARE API | |
| Website Audit | DNS Lookup | |
| TTFB Test | Is Site Up? | |
| TLS Scanner | Lighthouse | |
| WordPress Scanner | Loadtime | |
| DNS Lookup | Screenshot | |
| Secure Headers Test | TLS Scan | |
| Screenshot | Broken Link | |