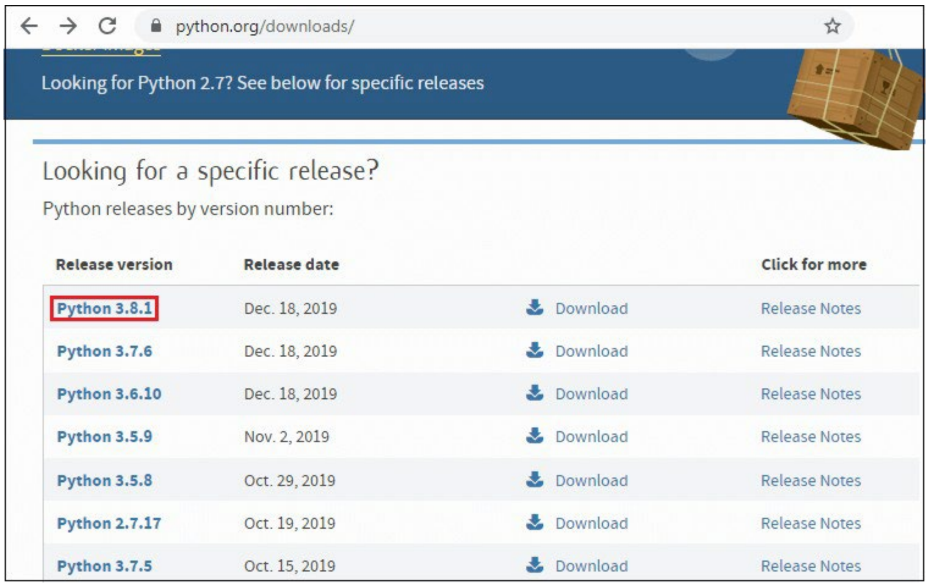


python.org/downloads/

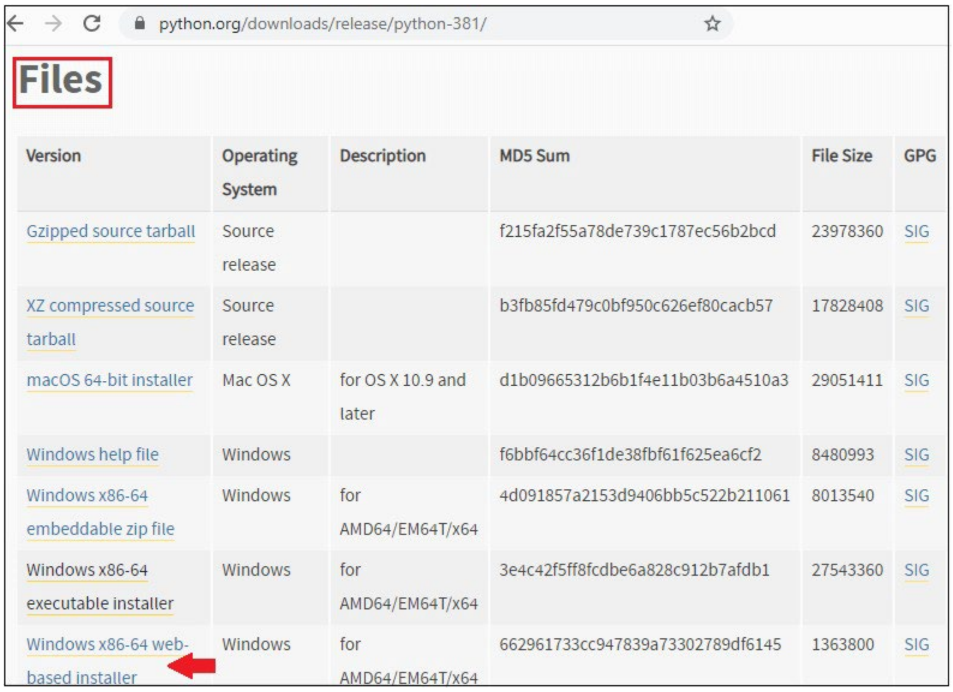
Looking for Python 2.7? See below for specific releases

Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.8.1	Dec. 18, 2019	Download	Release Notes
Python 3.7.6	Dec. 18, 2019	Download	Release Notes
Python 3.6.10	Dec. 18, 2019	Download	Release Notes
Python 3.5.9	Nov. 2, 2019	Download	Release Notes
Python 3.5.8	Oct. 29, 2019	Download	Release Notes
Python 2.7.17	Oct. 19, 2019	Download	Release Notes
Python 3.7.5	Oct. 15, 2019	Download	Release Notes

- The **Python-3.8.1** version window will appear on the screen, then scroll the page little-bit and find the **File** section, and the click on the **Windows x86-64 web-based installer** link for the Windows operating system as we can see in the below screenshot:



python.org/downloads/release/python-381/

Files

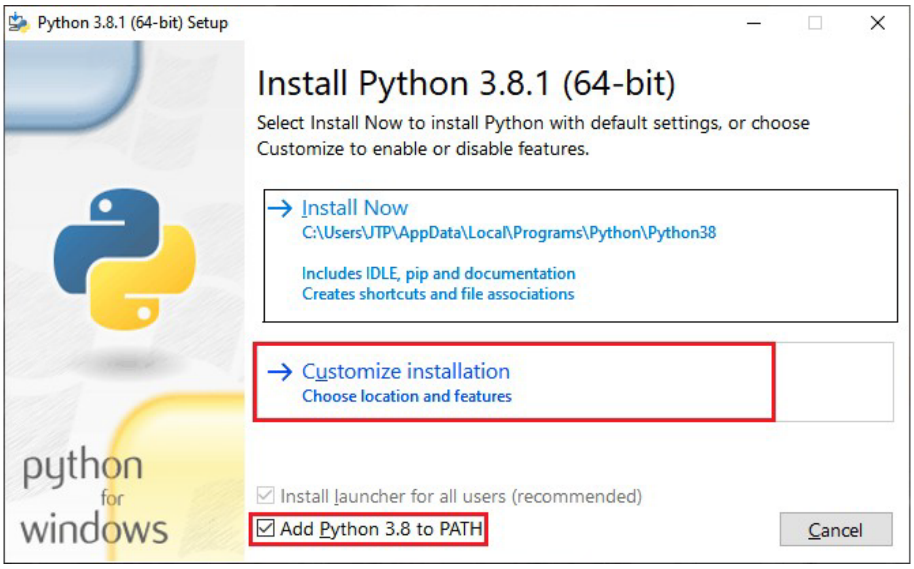
Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		f215fa2f55a78de739c1787ec56b2bcd	23978360	SIG
XZ compressed source tarball	Source release		b3fb85fd479c0bf950c626ef80cacb57	17828408	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	d1b09665312b6b1f4e11b03b6a4510a3	29051411	SIG
Windows help file	Windows		f6bbf64cc36f1de38fbf61f625ea6cf2	8480993	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	4d091857a2153d9406bb5c522b211061	8013540	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	3e4c42f5ff8fcdbe6a828c912b7afdb1	27543360	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	662961733cc947839a73302789df6145	1363800	SIG

Install the Python

After downloading the Python for **Windows-64 bit**, we will be ready to install the Python.

To install the Python, follow the below process:

- Once we double-click on the downloaded executable file, the **Python 3.8.1(64-bit)** setup window will appear on the screen, where we have two options available to install the Python, which are:
  - **Install Now**
  - **Customize installation**
- We will click on the **Customize installation,** and select **Add Python 3.8 to path** checkbox as we can see in the below image:



Python 3.8.1 (64-bit) Setup

Install Python 3.8.1 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

→ Install Now
C:\Users\JTP\AppData\Local\Programs\Python\Python38

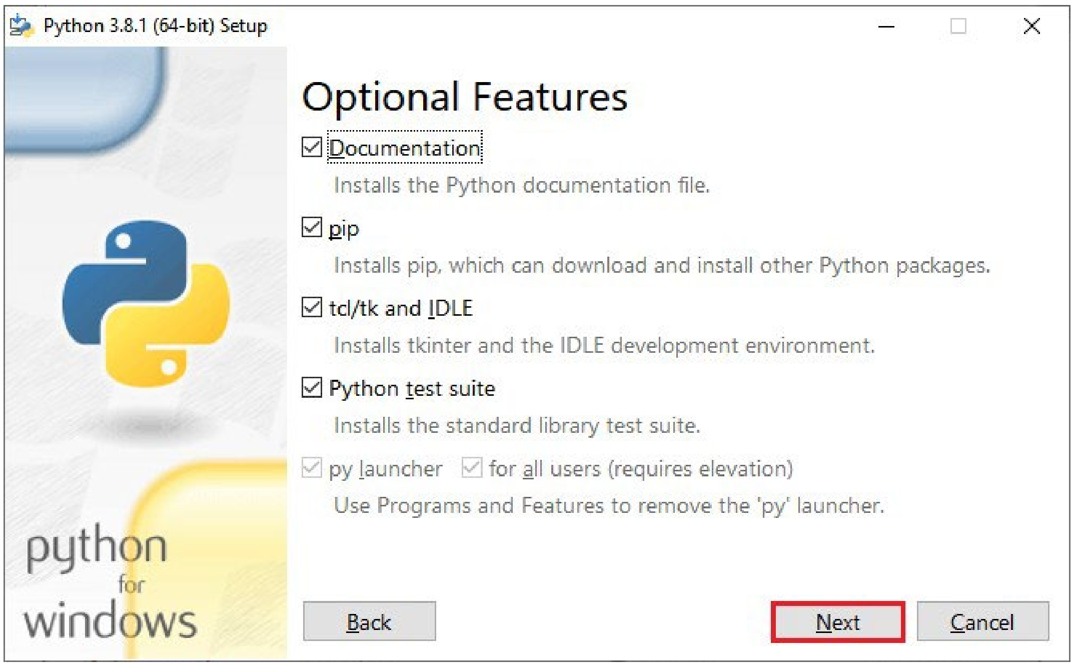
Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.8 to PATH

Cancel

- After, click on the customize installation, the **Optional Features** will appear on the screen, where we can select and deselect the features according to our requirements.
- Then, click on the **Next** button, to proceed further as we can see in the below image:



Python 3.8.1 (64-bit) Setup

Optional Features

☑ Documentation
Installs the Python documentation file.

☑ pip
Installs pip, which can download and install other Python packages.

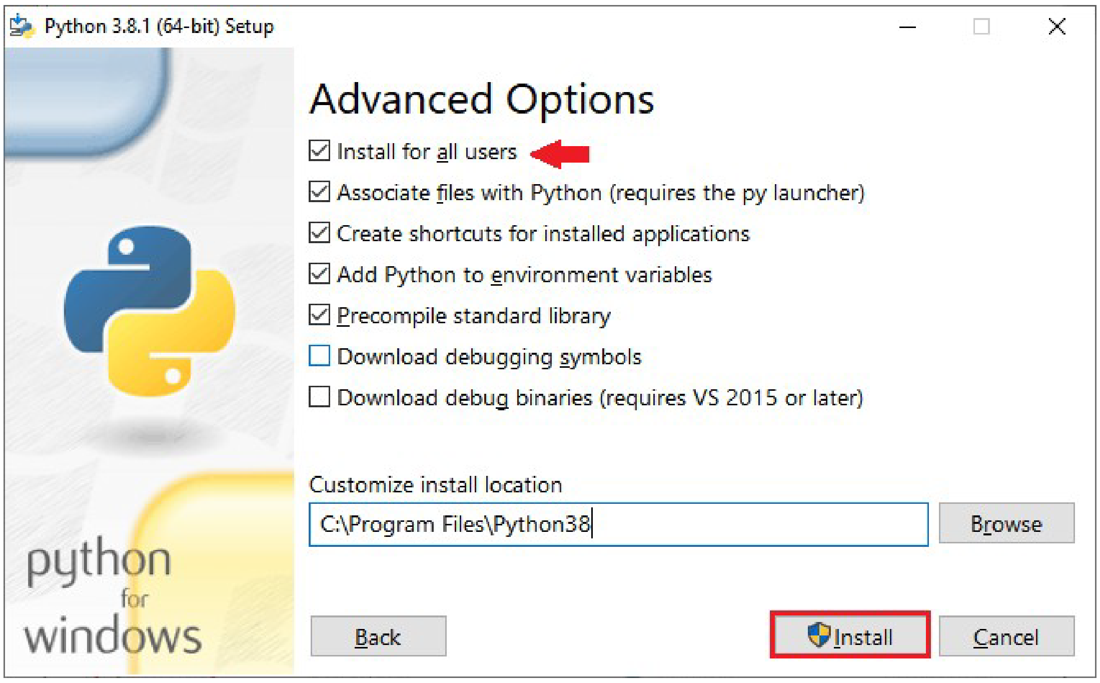
☑ tcl/tk and IDLE
Installs tkinter and the IDLE development environment.

☑ Python test suite
Installs the standard library test suite.

☑ py launcher ☑ for all users (requires elevation)
Use Programs and Features to remove the 'py' launcher.

Back Next Cancel

- Once, we clicked on the Next button; we have a list of **Advanced Options** available, where we can select the options based on our needs and also make sure that the **Install for all users** is selected.
- We can also customize the **install location** according to our convenience by clicking on the **Browse**
- After that, click on the **Install** button, to install the Python as we can see in the below screenshot:



Python 3.8.1 (64-bit) Setup

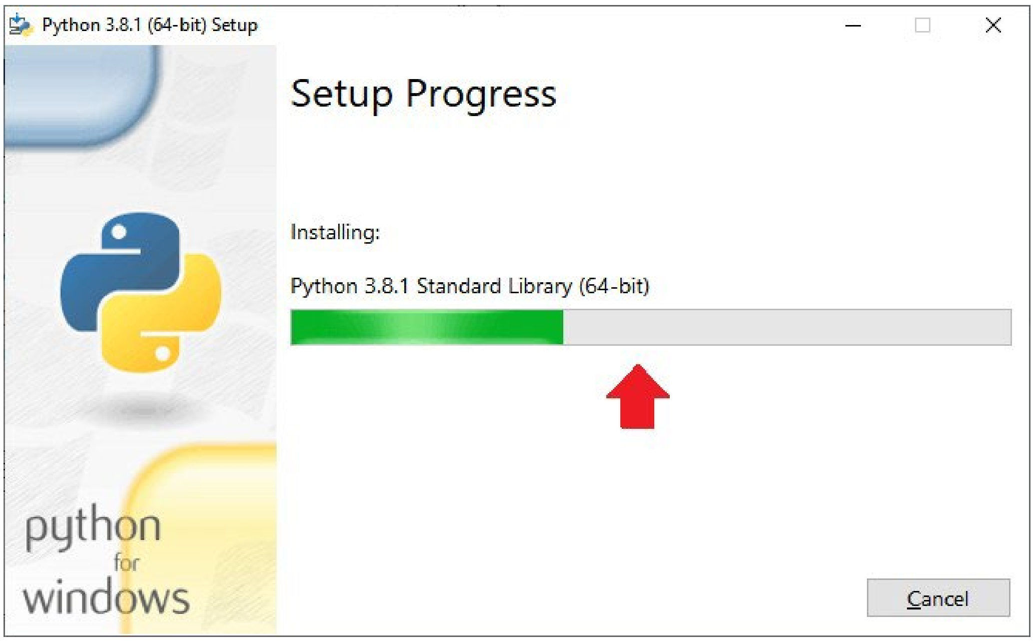
Advanced Options

- [x] Install for all users
- [x] Associate files with Python (requires the py launcher)
- [x] Create shortcuts for installed applications
- [x] Add Python to environment variables
- [x] Precompile standard library
- [ ] Download debugging symbols
- [ ] Download debug binaries (requires VS 2015 or later)

Customize install location

C:\Program Files\Python38

- The installing process is getting started after clicking on the Install button as we can see in the below screenshot:



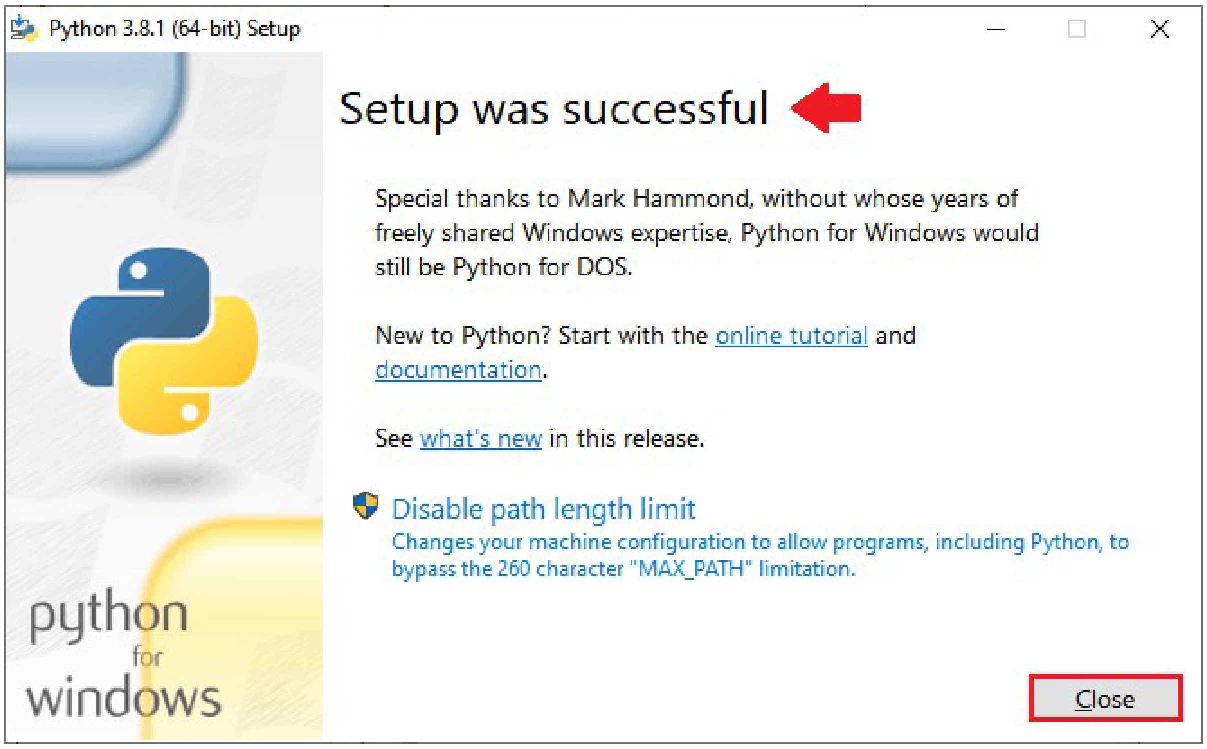
Python 3.8.1 (64-bit) Setup

Setup Progress

Installing:

Python 3.8.1 Standard Library (64-bit)

- When the installation is done, we got the confirmation message as **Setup was successful,** which means that the Python is installed successfully for the **Windows** operating system.
- Then, click on the **Close** button, to close the setup window as we can observe in the below screenshot:


Python 3.8.1 (64-bit) Setup

Setup was successful

Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

New to Python? Start with the online tutorial and documentation.

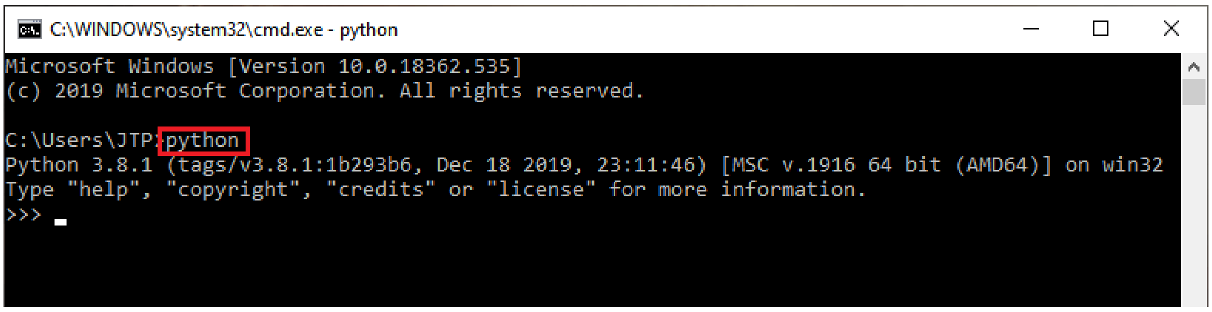
See what's new in this release.

Disable path length limit

Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.

After that, we will check whether Python is installed successfully and working fine or not.

So for this, we will open our command prompt, and type the command as **Python** and press the **Enter key**, and it will open the Python interpreter shell where we can implement the Python program as we can see in the below image:


C:\WINDOWS\system32\cmd.exe - python

```
Microsoft Windows [Version 10.0.18362.535]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\JTP>python
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

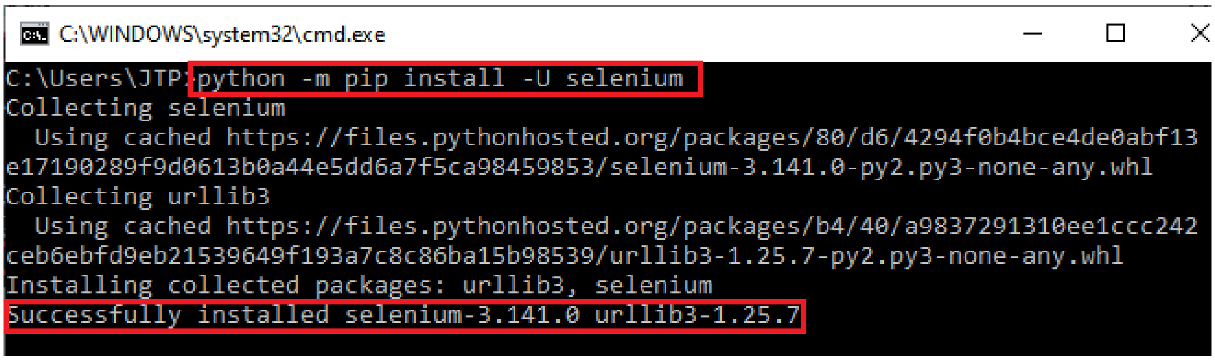
Installing the Selenium libraries in Python

Once we successfully install the Python in our operation system, we will install the Selenium libraries.

For this, we will execute the following command in our command prompt:

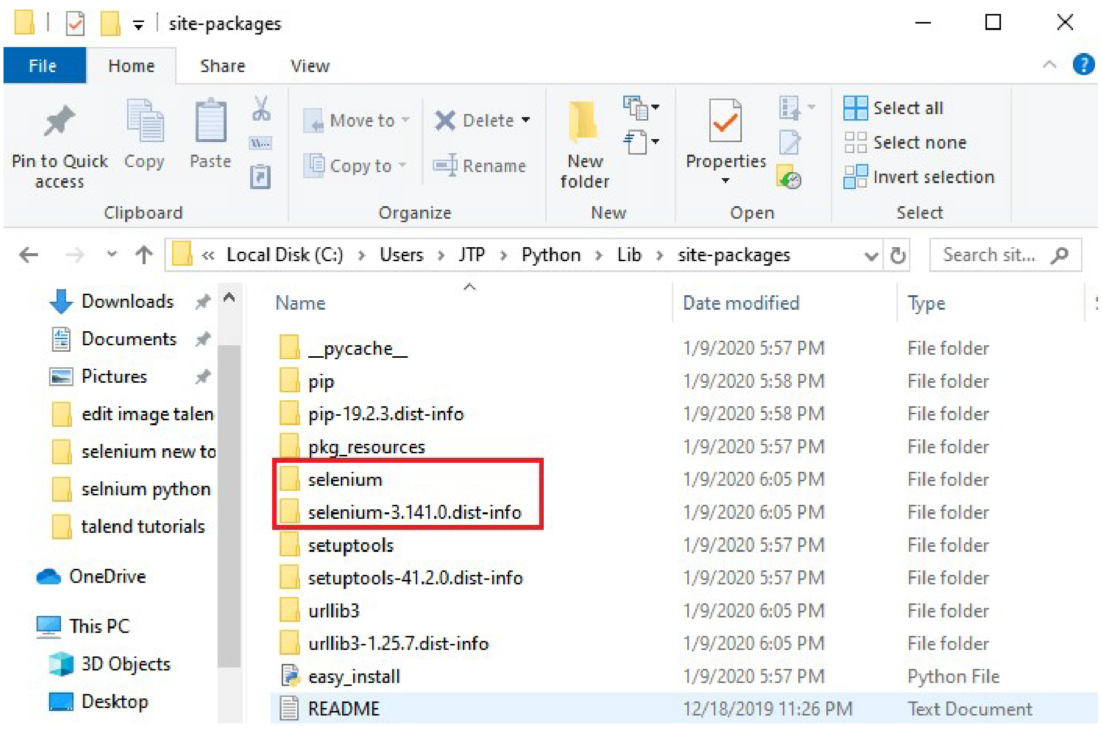
1. Python -m pip install -U Selenium

And, this command will successfully install the latest **Selenium package** i.e., **Selenium -3.141.0** added to the libraries as we can see in the below image:



```
C:\WINDOWS\system32\cmd.exe
C:\Users\JTP>python -m pip install -U selenium
Collecting selenium
  Using cached https://files.pythonhosted.org/packages/80/d6/4294f0b4bce4de0abf13
e17190289f9d0613b0a44e5dd6a7f5ca98459853/selenium-3.141.0-py2.py3-none-any.whl
Collecting urllib3
  Using cached https://files.pythonhosted.org/packages/b4/40/a9837291310ee1ccc242
ceb6ebfd9eb21539649f193a7c8c86ba15b98539/urllib3-1.25.7-py2.py3-none-any.whl
Installing collected packages: urllib3, selenium
Successfully installed selenium-3.141.0 urllib3-1.25.7
```

After that executing the above command, it will create the **Selenium folder** automatically having all the Selenium libraries as we can see in the below screenshot:



site-packages

Local Disk (C:) › Users › JTP › Python › Lib › site-packages

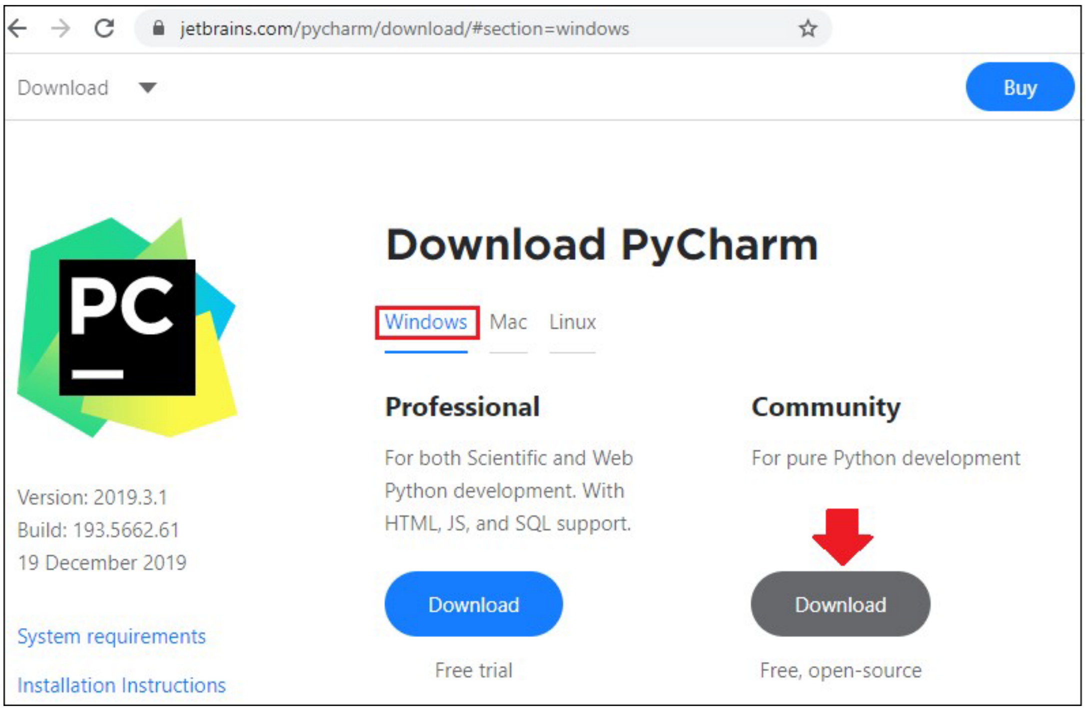
Name	Date modified	Type
__pycache__	1/9/2020 5:57 PM	File folder
pip	1/9/2020 5:58 PM	File folder
pip-19.2.3.dist-info	1/9/2020 5:58 PM	File folder
pkg_resources	1/9/2020 5:57 PM	File folder
selenium	1/9/2020 6:05 PM	File folder
selenium-3.141.0.dist-info	1/9/2020 6:05 PM	File folder
setuptools	1/9/2020 5:57 PM	File folder
setuptools-41.2.0.dist-info	1/9/2020 5:57 PM	File folder
urllib3	1/9/2020 6:05 PM	File folder
urllib3-1.25.7.dist-info	1/9/2020 6:05 PM	File folder
easy_install	1/9/2020 5:57 PM	Python File
README	12/18/2019 11:26 PM	Text Document

Download and install PyCharm

Once we successfully install the Selenium libraries into Python, we are ready to download Python IDE that is PyCharm.

To download the PyCharm, follow the below process:

- Refer the below link, to download the PyCharm https://www.jetbrains.com/pycharm/download/#section=windows
- Once we clicked on the above link, we will get the below window, where will click on the **Download** button under the **Community** section for the **Windows**



jetbrains.com/pycharm/download/#section=windows

Download Buy

Download PyCharm

Windows Mac Linux

Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

Download

Free trial

Community

For pure Python development

Download

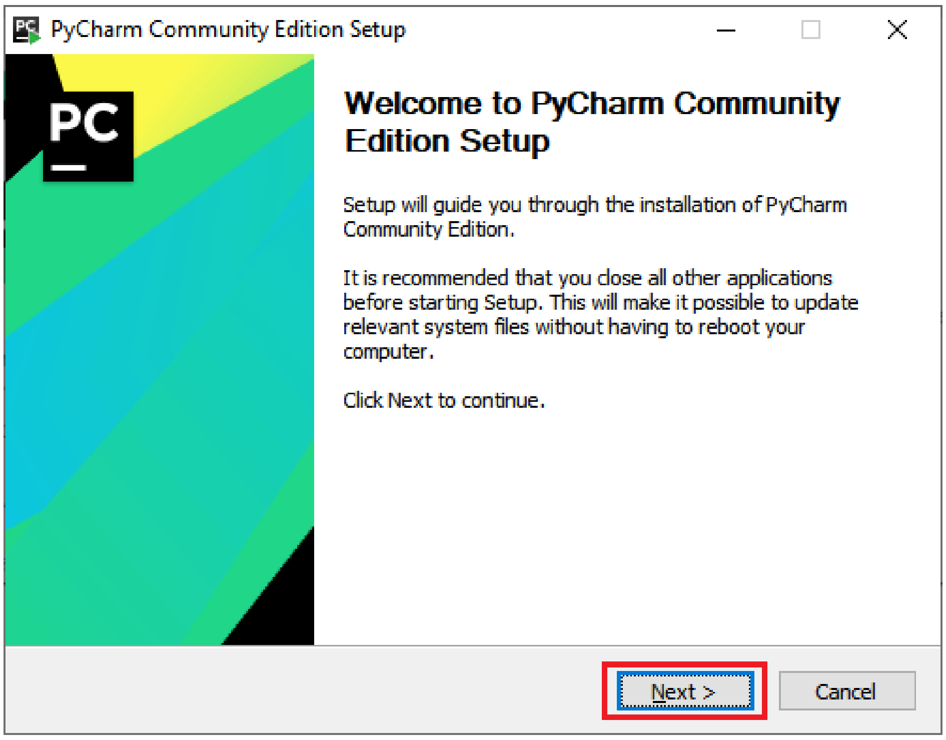
Free, open-source

Version: 2019.3.1
Build: 193.5662.61
19 December 2019

System requirements

Installation Instructions

- After that, double-click on the executable file to install the PyCharm, and the **PyCharm Community Edition Setup** window will appear on the screen, where we click on the **Next** button to proceed further as we can see in the below image:



PyCharm Community Edition Setup

Welcome to PyCharm Community Edition Setup

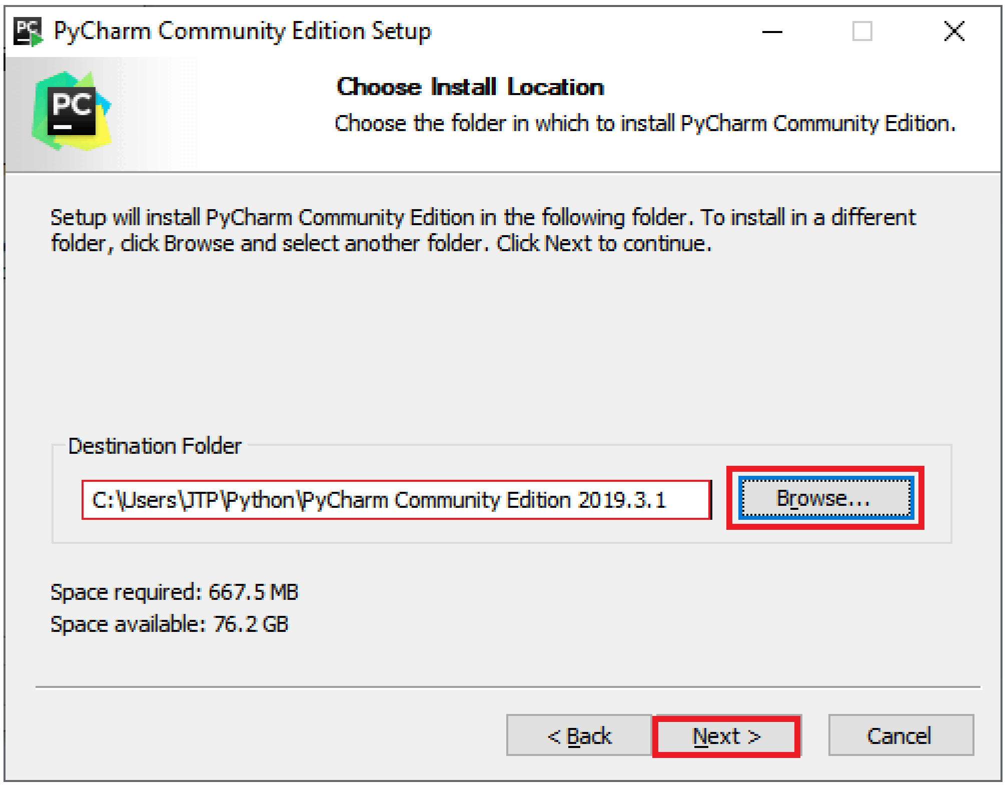
Setup will guide you through the installation of PyCharm Community Edition.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Next > Cancel

- In the next step, we can **Choose Install location** by clicking on the **Browser** button, then click on the **Next** button for further process.



PyCharm Community Edition Setup

Choose Install Location

Choose the folder in which to install PyCharm Community Edition.

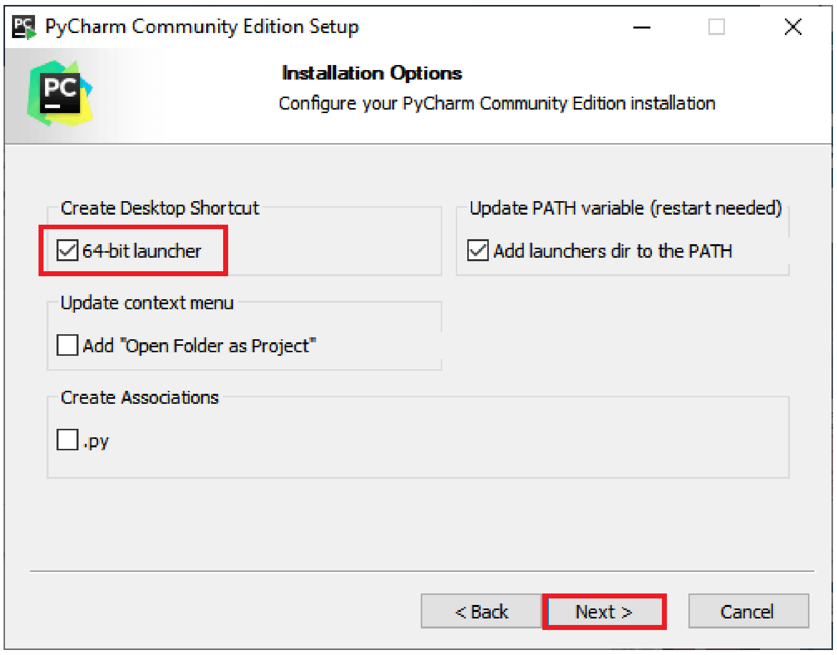
Setup will install PyCharm Community Edition in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder

C:\Users\JTP\Python\PyCharm Community Edition 2019.3.1

Space required: 667.5 MB
Space available: 76.2 GB

- In the next step, we have some **Installation Options** available, and we can select them based on our requirements.
- After that, click on the **Next** button as we can see in the below image:



PyCharm Community Edition Setup

Installation Options

Configure your PyCharm Community Edition installation

Create Desktop Shortcut

☑ 64-bit launcher

Update PATH variable (restart needed)

☑ Add launchers dir to the PATH

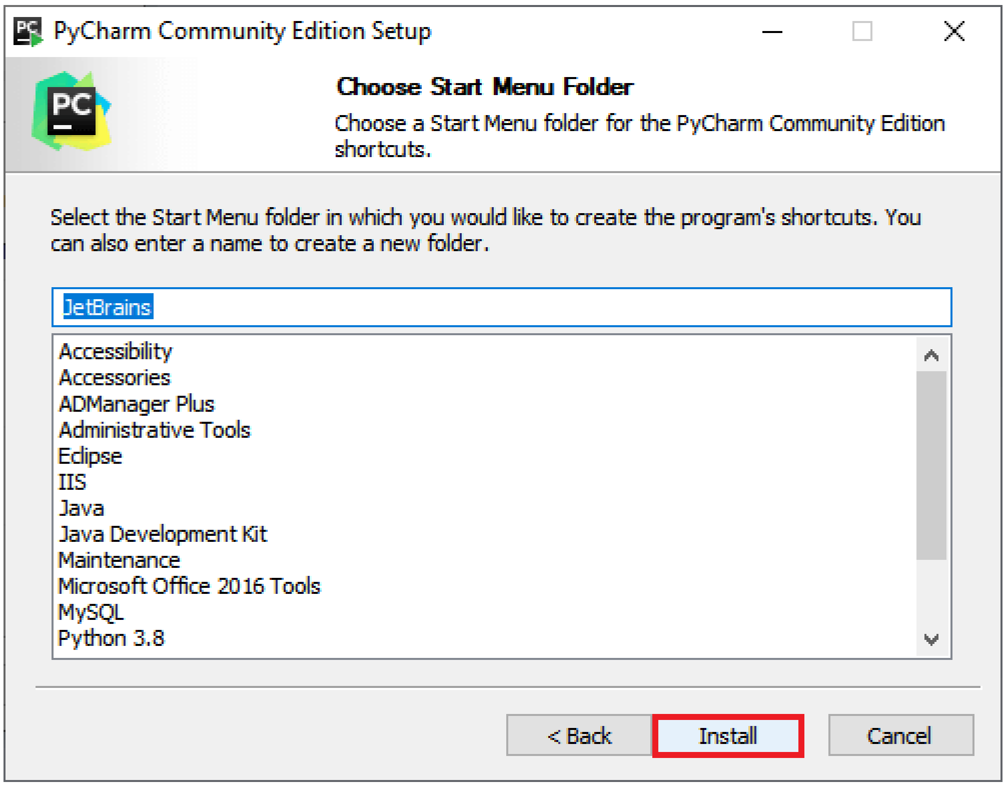
Update context menu

☐ Add "Open Folder as Project"

Create Associations

☐ .py

- Then, click on the **Install** button to install the PyCharm, as we can see in the below screenshot:


PyCharm Community Edition Setup

Choose Start Menu Folder

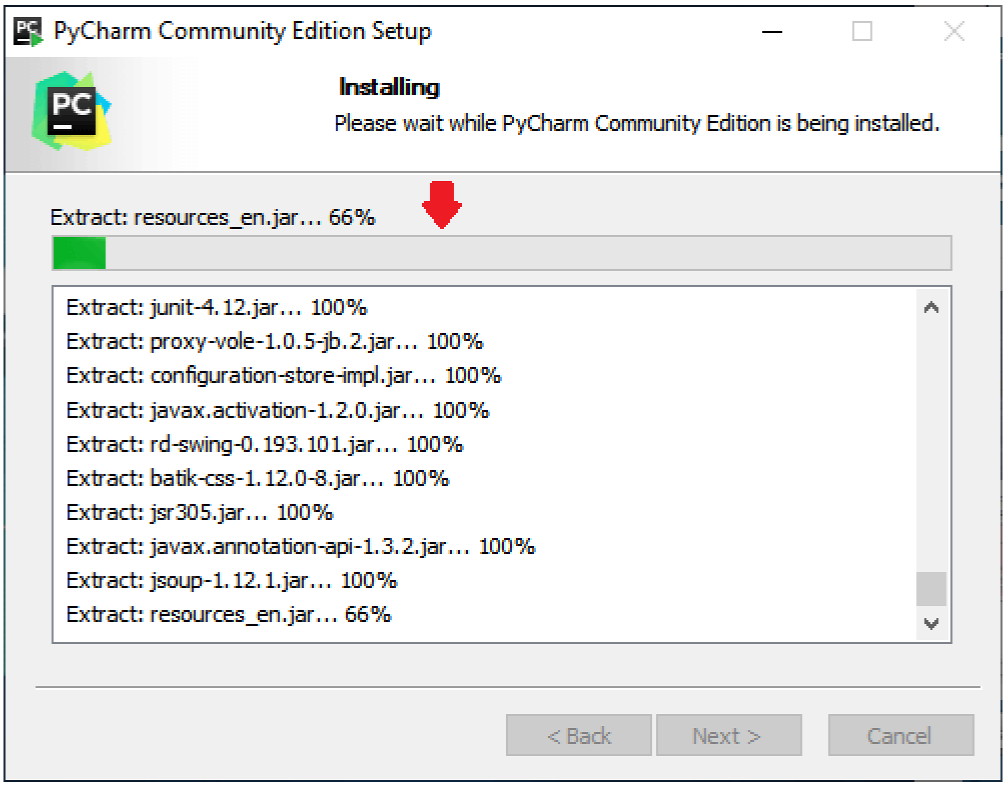
Choose a Start Menu folder for the PyCharm Community Edition shortcuts.

Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

JetBrains

Accessibility
Accessories
ADManager Plus
Administrative Tools
Eclipse
IIS
Java
Java Development Kit
Maintenance
Microsoft Office 2016 Tools
MySQL
Python 3.8

- As we can see in the below image, the installation process is getting started.


PyCharm Community Edition Setup

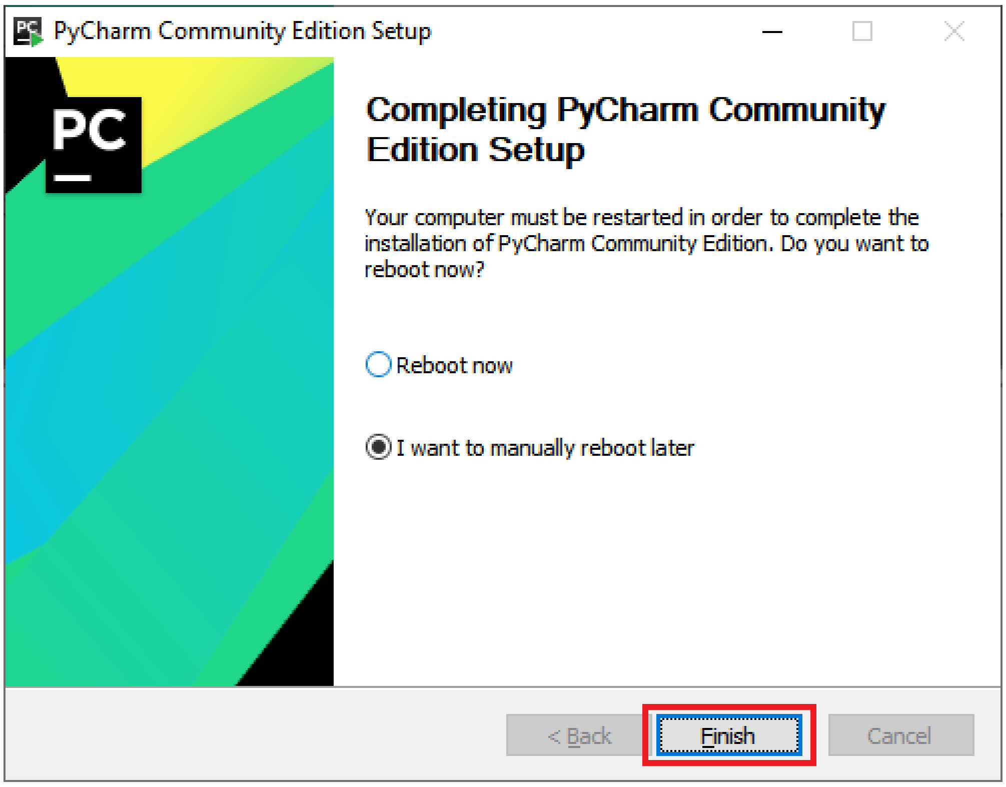
Installing

Please wait while PyCharm Community Edition is being installed.

Extract: resources_en.jar... 66%

```
Extract: junit-4.12.jar... 100%
Extract: proxy-vole-1.0.5-jb.2.jar... 100%
Extract: configuration-store-impl.jar... 100%
Extract: javax.activation-1.2.0.jar... 100%
Extract: rd-swing-0.193.101.jar... 100%
Extract: batik-css-1.12.0-8.jar... 100%
Extract: jsr305.jar... 100%
Extract: javax.annotation-api-1.3.2.jar... 100%
Extract: jsoup-1.12.1.jar... 100%
Extract: resources_en.jar... 66%
```

- Then, click on the **Finish** button to finish the installation process as we can see in the below image:


PyCharm Community Edition Setup

Completing PyCharm Community Edition Setup

Your computer must be restarted in order to complete the installation of PyCharm Community Edition. Do you want to reboot now?

Reboot now [✓ I want to manually reboot later]

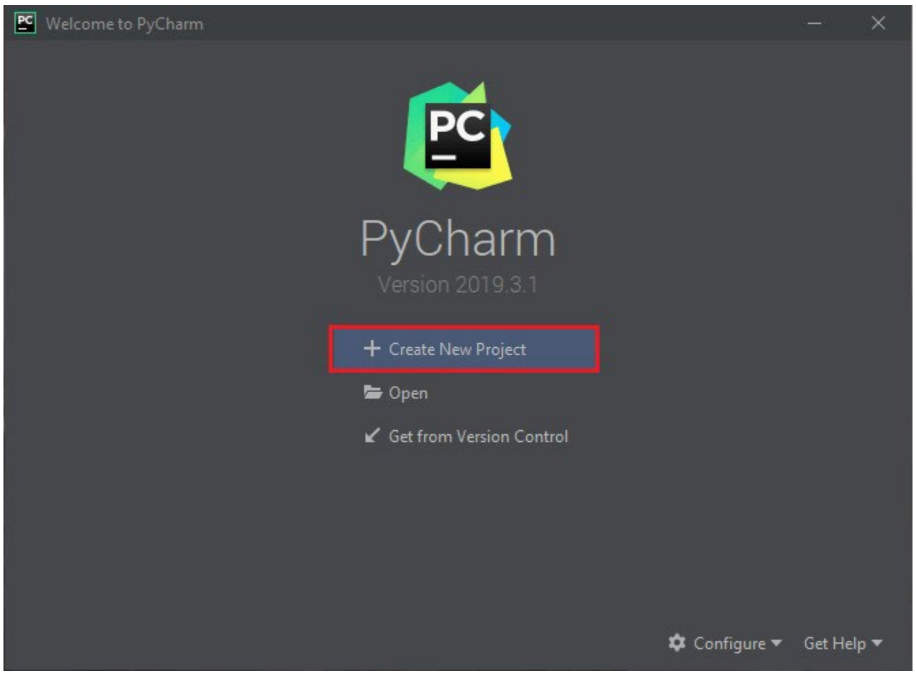
Create a new project and write the Selenium test script

Once we successfully install the PyCharm, we will open the PyCharm IDE for creating a new project.

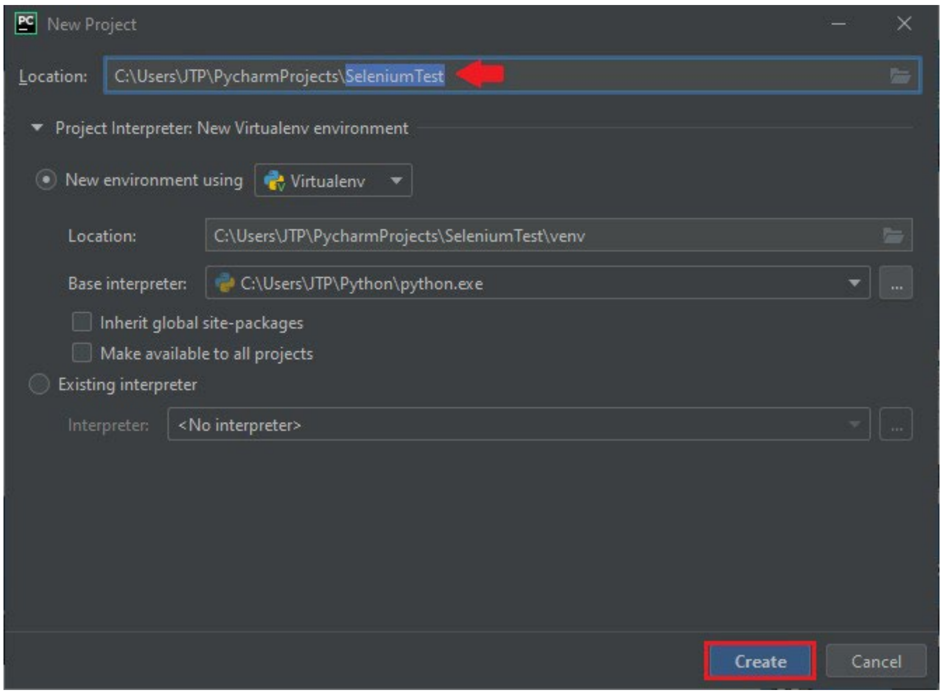
Create a New Project in PyCharm

Follow the below process, to create a new project in PyCharm:

- First, open the PyCharm by Double-click on it, and click on the **Create New Project** as we can see in the below image:



- After that, we will provide the project name as **SeleniumTest**, and click on the **Create** button as we can see in the below image:



Location: C:\Users\JTP\PycharmProjects\SeleniumTest

Project Interpreter: New Virtualenv environment

New environment using: Virtualenv

Location: C:\Users\JTP\PycharmProjects\SeleniumTest\venv

Base interpreter: C:\Users\JTP\Python\python.exe

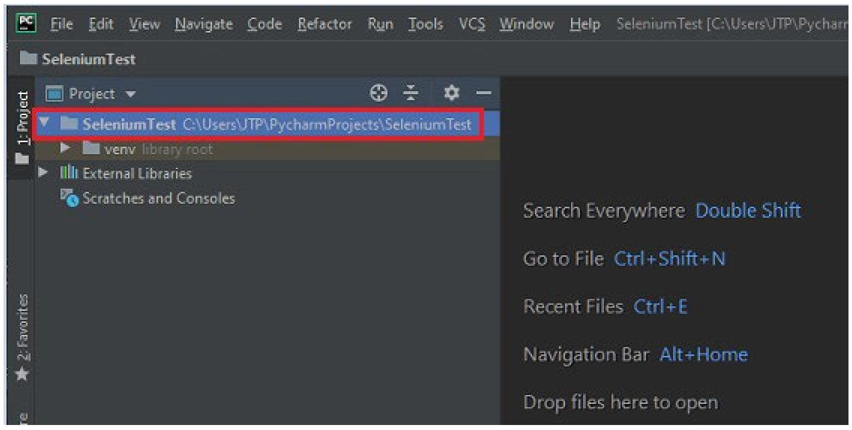
Inherit global site-packages

Make available to all projects

Existing interpreter

Interpreter: <No interpreter>

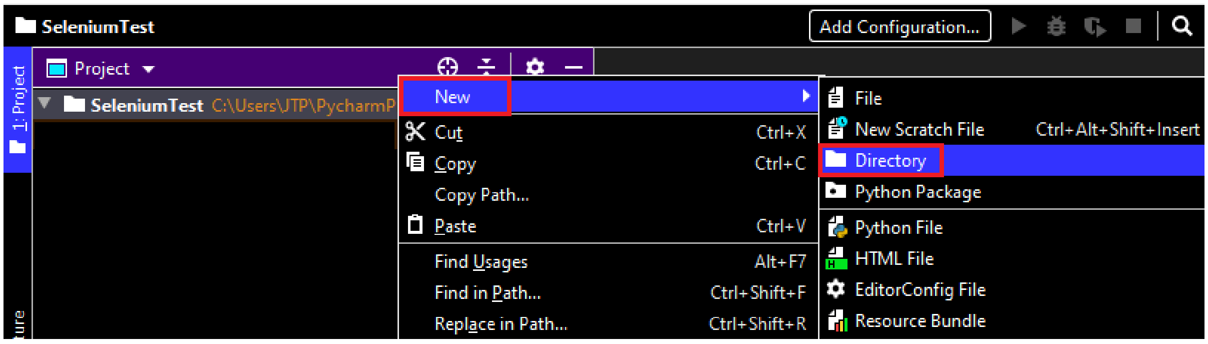
- After clicking on the Create button, we will get the below window:


File Edit View Navigate Code Refactor Run Tools VCS Window Help SeleniumTest [C:\Users\JTP\Pycharm
SeleniumTest
Project
SeleniumTest C:\Users\JTP\PycharmProjects\SeleniumTest
venv library root
External Libraries
Scratches and Consoles
Search Everywhere Double Shift
Go to File Ctrl+Shift+N
Recent Files Ctrl+E
Navigation Bar Alt+Home
Drop files here to open

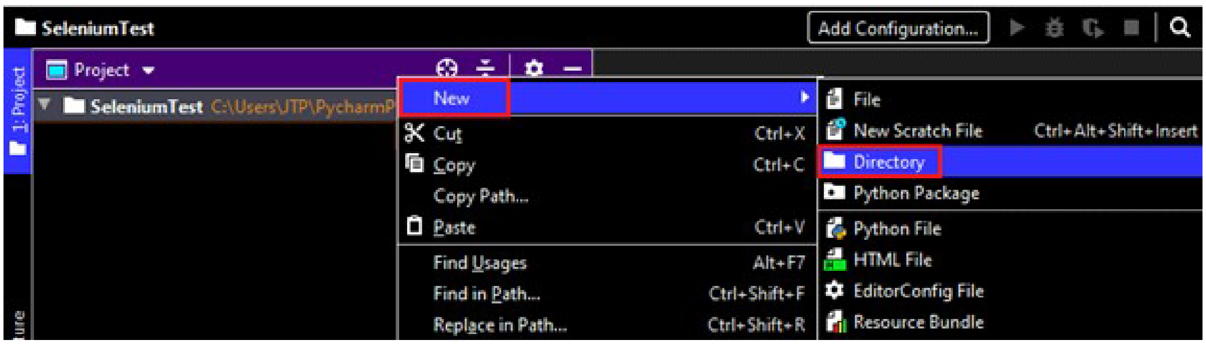
Adding Selenium Test Scripts

For adding the Selenium test scripts in the PyCharm, follow the below process:

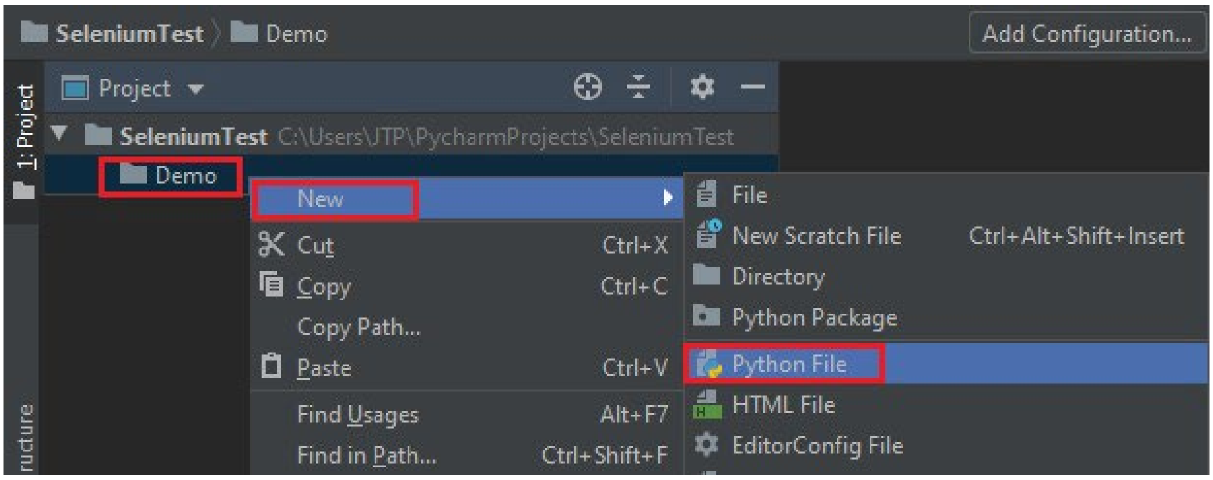
- Right-click on the **SeleniumTest** project, then go to **New,** and we can add any of the options in the given list according to our requirements.
- But, here we are adding the Python file, so for this, we will add the **Directory** which helps us to manage them separately as we can see in the below screenshot:



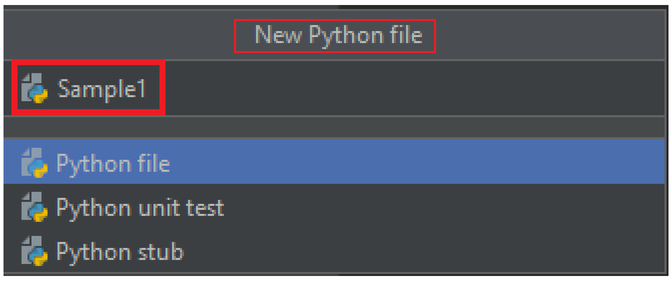
- And, provide the Directory name, in our case we give it as **Demo**
- After that, press the **Enter** key as we can see in the below screenshot:



- After creating a Directory, we will right-click on the **Demo** Directory then go to **New**, and select **Python File** from the pop-up menu as we can see in the below image: **Demo → New → Python File**



- And, we provide a name to python file as **Sample1**.
- Then, press the **Enter** key as we can see in the below image:



- After that, we got the IDE where we can create or write our Selenium test Scripts.

Write the Selenium test script

For our testing purpose, we will first go to the **Google Home page** and search **javatpoint** from there.

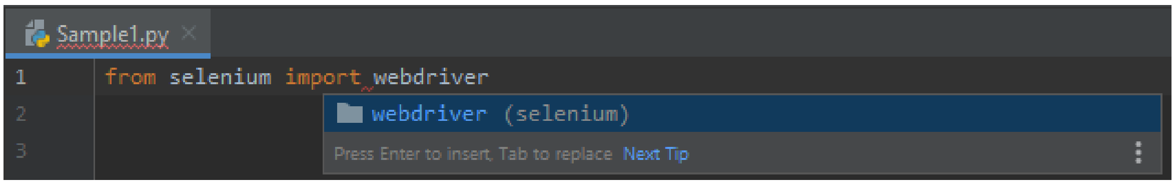
We are creating our sample test script step by step to give you a complete understanding of how we write a Selenium test script in Python programming language.

For this, follow the below steps:

Step1

In the first step, we will type the following statement to import the web driver:

```
1. from selenium import webdriver
```

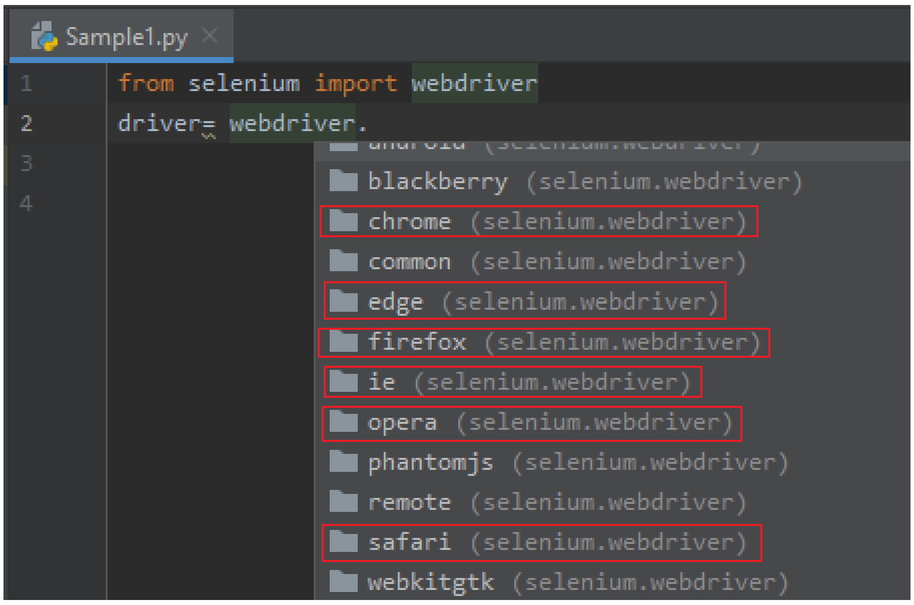


```
from selenium import webdriver
```

Step2

After that, we will open the Google Chrome browser.

As we can see in the below screenshot, we have multiple types of browsers options available, and we can select any browser from the list like **Chrome, Edge, firefox, Internet Explorer, opera, safari, etc**.



```
from selenium import webdriver
driver= webdriver.
```

```
blackberry (selenium.webdriver)
chrome (selenium.webdriver)
common (selenium.webdriver)
edge (selenium.webdriver)
firefox (selenium.webdriver)
ie (selenium.webdriver)
opera (selenium.webdriver)
phantomjs (selenium.webdriver)
remote (selenium.webdriver)
safari (selenium.webdriver)
webkitgtk (selenium.webdriver)
```

Following are the sample code for opening the Google Chrome browser:

1. driver = webdriver.Chrome()

Step3

In the next step, we will be maximizing our browser window size, and the sample code is as below:

1. driver.maximize_window()

Step4

Then, we will navigate to the given URL.

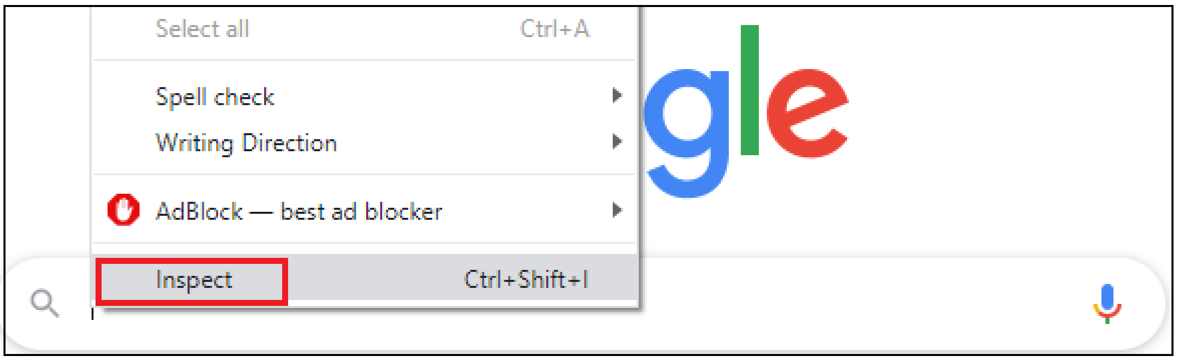
The sample code is as below:

1. driver.get("https://www.google.com/")

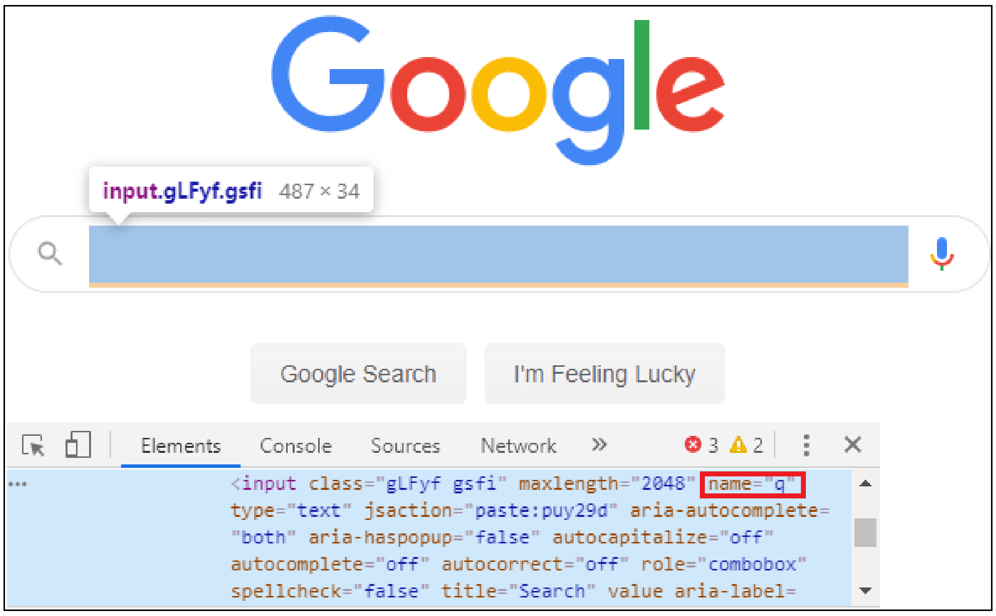
Step5

In this step, we are trying to locate the Google search text box with the help of its **Name** attribute value.

- Right-click on the **Google search** text box, and select the **Inspect** option in the pop-up menu as we can see in the below image:



- The developer tool window will be launched with all the specific codes used in the development of the **Google search** text box.
- And, copy the value of its **Name** attribute, that is "**q**" as we can see in the below image:



```
<input class="gLFyf gsfi" maxlength="2048" name="q"
type="text" jsaction="paste:puy29d" aria-autocomplete=
"both" aria-haspopup="false" autocapitalize="off"
autocomplete="off" autocorrect="off" role="combobox"
spellcheck="false" title="Search" value aria-label=
```

Here the sample code:

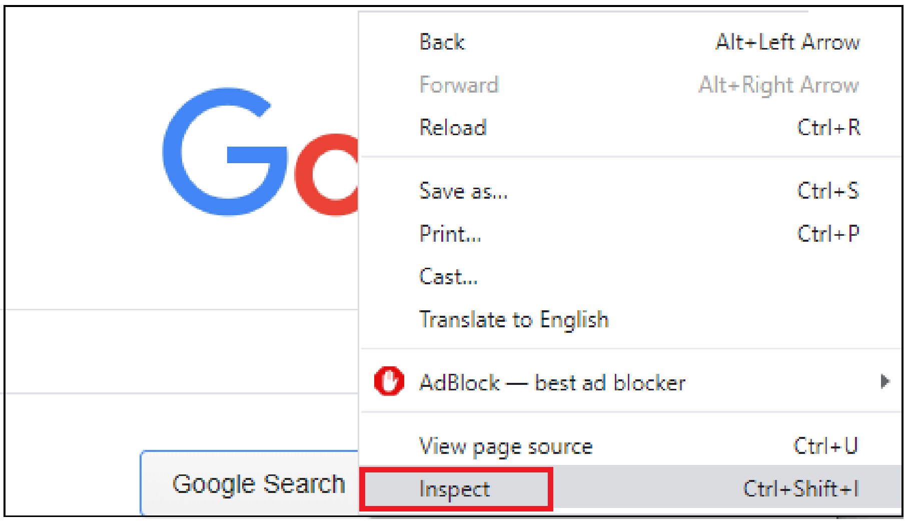
1. driver.find_element_by_name("q").send_keys("javatpoint")

Step6

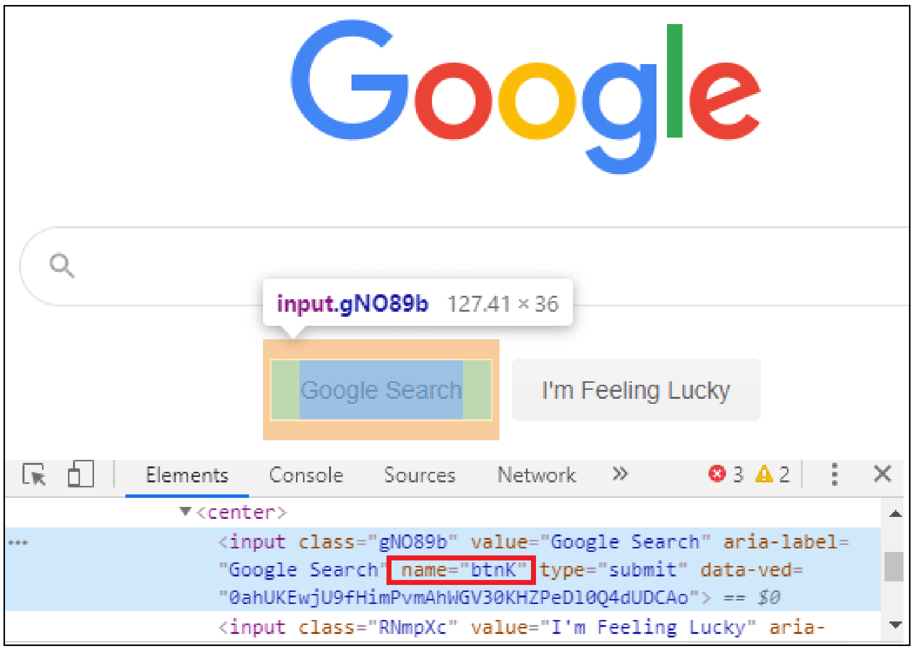
Once we identify the Google search text box, and we will identify the **Google Search button**.

So for this, follow the below process:

- Right-click on the **Google search**button, and select the **Inspect** option from the given pop-up menu as we can see in the below image:



- The developer tool window will be launched with having all the specific codes used in the development of the **Google search** button.
- Then, copy the value of its **name** attribute that is "**btnK**" as we can see in the below image:



```
▼<center>
   <input class="gNO89b" value="Google Search" aria-label=
   "Google Search" name="btnK" type="submit" data-ved=
   "0ahUKEwjU9fHimPvmAhWGV30KHZPeD10Q4dUDCAo"> == $0
   <input class="RNmpXc" value="I'm Feeling Lucky" aria-
```

And, the sample code is as following:

```
1. driver.find_element_by_name("btnK").send_keys(Keys.ENTER)
```

Step7

In the last step, we are closing the browser.

And, the sample code for closing the browser is as follows:

```
driver.close()
```

Our final test script will look like this, after completing all the above steps:

```
from Selenium import webdriver
import time
from Selenium.webdriver.common.keys import Keys
print("sample test case started")
driver = webdriver.Chrome()
#driver=webdriver.firefox()
#driver=webdriver.ie()
#maximize the window size
driver.maximize_window()
#navigate to the url
driver.get("https://www.google.com/")
#identify the Google search text box and enter the value
driver.find_element_by_name("q").send_keys("javatpoint")
time.sleep(3)
#click on the Google search button
driver.find_element_by_name("btnK").send_keys(Keys.ENTER)
time.sleep(3)
#close the browser
driver.close()
print("sample test case successfully completed")
```

Note:

Import time: Time is a Python module, which is used to handle the time-related tasks such as time.sleep().

from Selenium.webdriver.common.keys import Keys:

Here, we are adding Keys libraries from Selenium, like in the above code, we are using the **Enter** key instead of **click()** method to perform a particular scenario.

Run and validate the test scripts

Once we are done with writing the Selenium test script, we will run our test scripts.

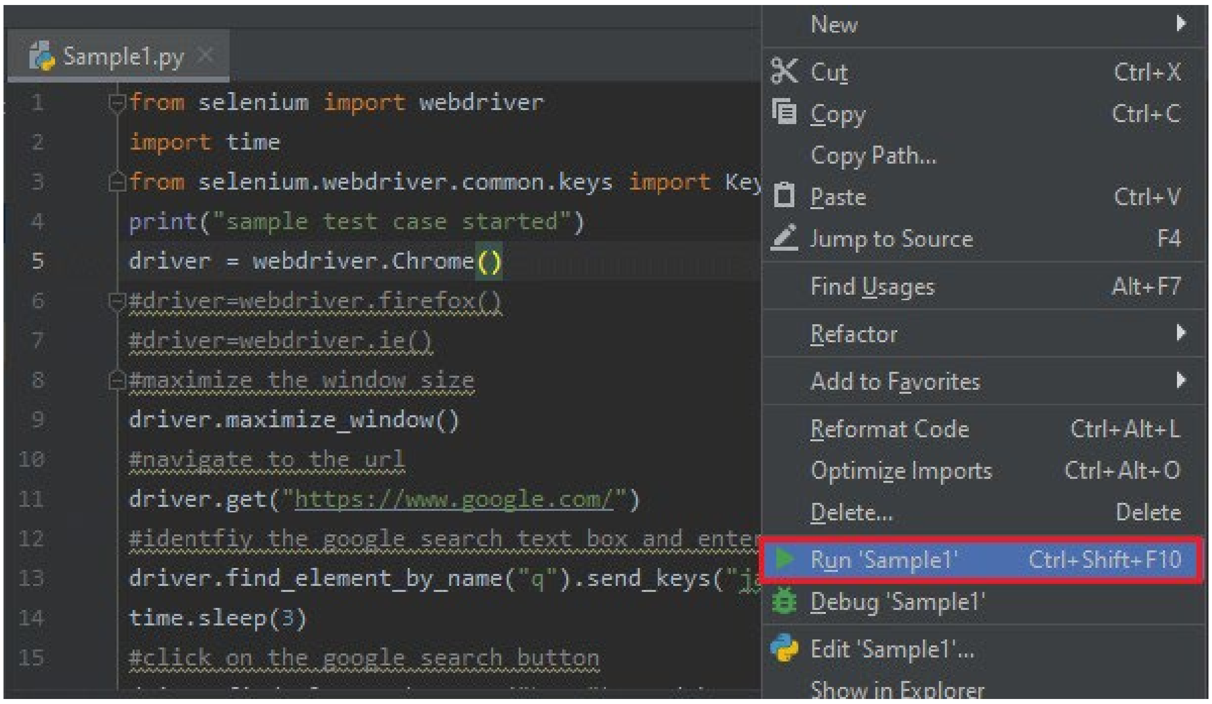
Here we will run our test scripts in two ways:

- **Run in Python IDE**
- **Run in Command Prompt**

Run in Python IDE

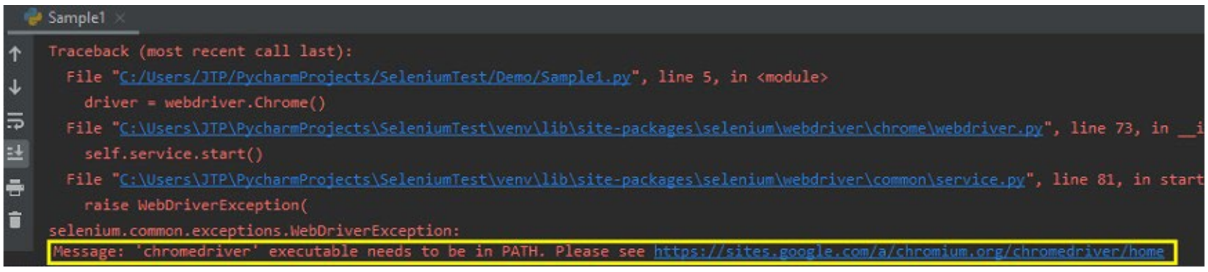
So, for this first, we will see how to run the Selenium test script in Python IDE.

- Right-click on the code, and select **Run 'Sample1'** from the popup menu as we can see in the below screenshot:



```
from selenium import webdriver
import time
from selenium.webdriver.common.keys import Key
print("sample test case started")
driver = webdriver.Chrome()
#driver=webdriver.firefox()
#driver=webdriver.ie()
#maximize the window size
driver.maximize_window()
#navigate to the url
driver.get("https://www.google.com/")
#identfiy the google search text box and ente
driver.find_element_by_name("q").send_keys("j
time.sleep(3)
#click on the google search button
```

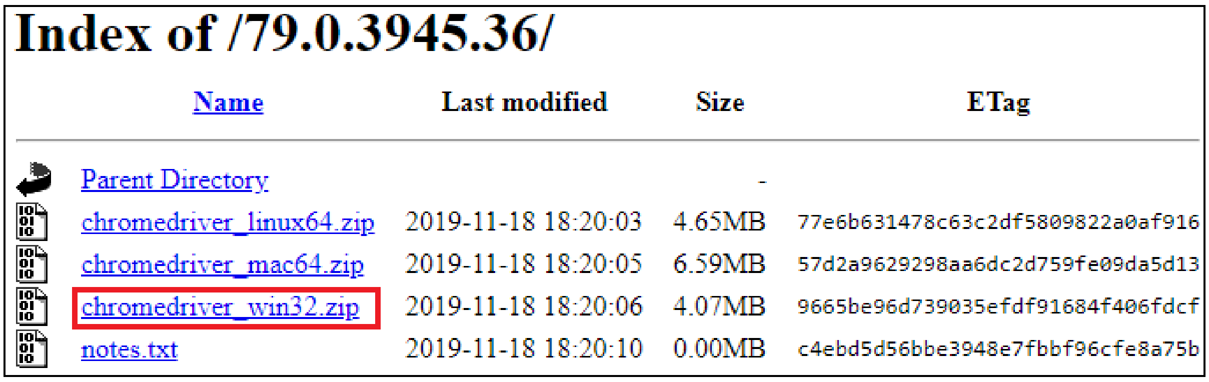
- When we run this script it will give an exception because we don't have the Chrome driver executable file as we can in the below image:



```
Traceback (most recent call last):
  File "C:/Users/JTP/PycharmProjects/SeleniumTest/Demo/Sample1.py", line 5, in <module>
    driver = webdriver.Chrome()
  File "C:\Users\JTP\PycharmProjects\SeleniumTest\venv\lib\site-packages\selenium\webdriver\chrome\webdriver.py", line 73, in __i
    self.service.start()
  File "C:\Users\JTP\PycharmProjects\SeleniumTest\venv\lib\site-packages\selenium\webdriver\common\service.py", line 81, in start
    raise WebDriverException(
selenium.common.exceptions.WebDriverException:
Message: 'chromedriver' executable needs to be in PATH. Please see https://sites.google.com/a/chromium.org/chromedriver/home
```

To overcome this exception, we will download the chrome driver executable from below
link: https://chromedriver.storage.googleapis.com/index.html?path=79.0.3945.36/

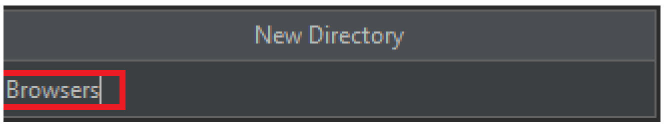
- Once we click on the above link, we will click on the **zip file** based upon our operating system platform. Like we have **Windows platform** that's why we clicked on the**zip** to download the Executable file as we can see in the below screenshot:



Index of /79.0.3945.36/

	Name	Last modified	Size	ETag
	Parent Directory		-	
	chromedriver_linux64.zip	2019-11-18 18:20:03	4.65MB	77e6b631478c63c2df5809822a0af916
	chromedriver_mac64.zip	2019-11-18 18:20:05	6.59MB	57d2a9629298aa6dc2d759fe09da5d13
	chromedriver_win32.zip	2019-11-18 18:20:06	4.07MB	9665be96d739035efdf91684f406fdcf
	notes.txt	2019-11-18 18:20:10	0.00MB	c4ebd5d56bbe3948e7fbbf96cfe8a75b

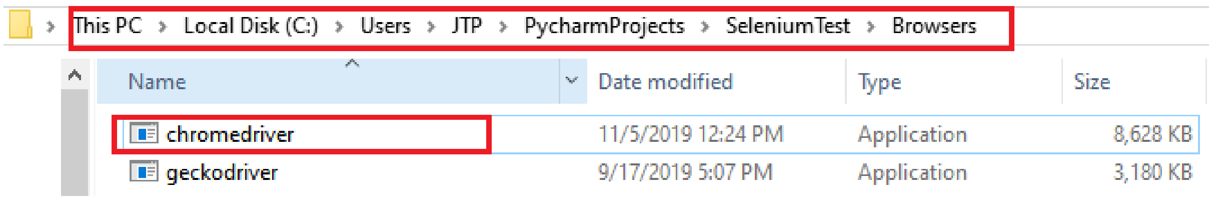
- After downloading the **exe** file, we can paste this file to the Python folder and unzip it.
- Then, we will create one more folder called libraries as **Browsers** in the Python IDE.
- Right-click on the Project**(SeleniumTest)** → **New** → **Directory** as we can see in the below screenshot:



New Directory

Browsers

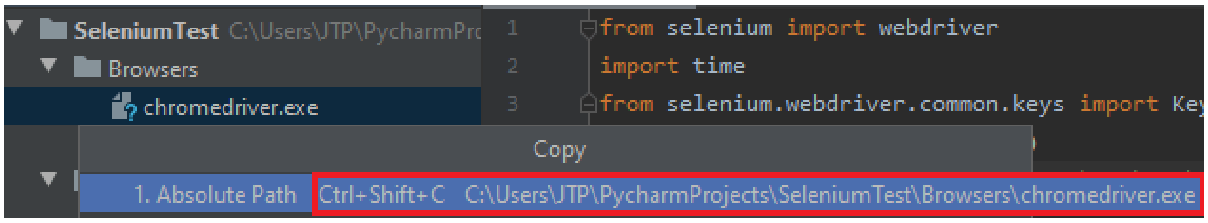
- And, we will add all the driver's executable files in the **Browsers** folder manually.
- For this, we will copy the **chrome driver exe** file from the **Python folder**, and paste in the **Browser** folder as we can see in the below image:



This PC › Local Disk (C:) › Users › JTP › PycharmProjects › SeleniumTest › Browsers

Name	Date modified	Type	Size
chromedriver	11/5/2019 12:24 PM	Application	8,628 KB
geckodriver	9/17/2019 5:07 PM	Application	3,180 KB

- Now go to **PyCharm** IDE, and copy the **Absolute path** of chromdriver.exe file as we can see in the below screenshot:



```
from selenium import webdriver
import time
from selenium.webdriver.common.keys import Key
```

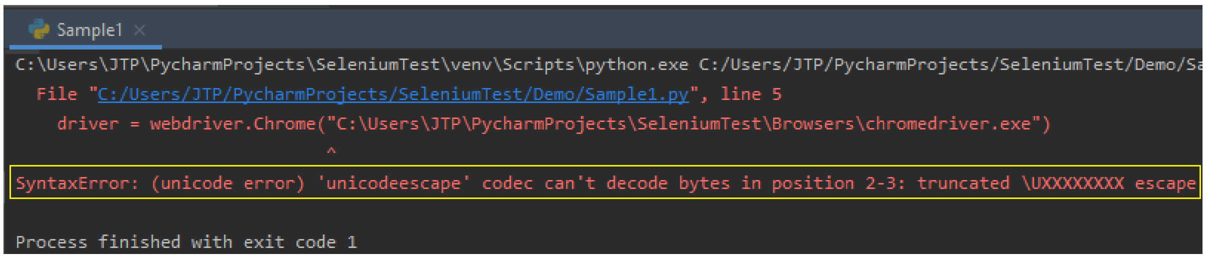
Ctrl+Shift+C C:\Users\JTP\PycharmProjects\SeleniumTest\Browsers\chromedriver.exe

- Replace the statement "driver = webdriver.Chrome()" with a statement given below:

1. driver=webdriver.Chrome(r"C:\Users\JTP\PycharmProjects\SeleniumTest\Browsers\chromedriver.exe")

Note: Here, we will use "r" to overcome the Unicode error.

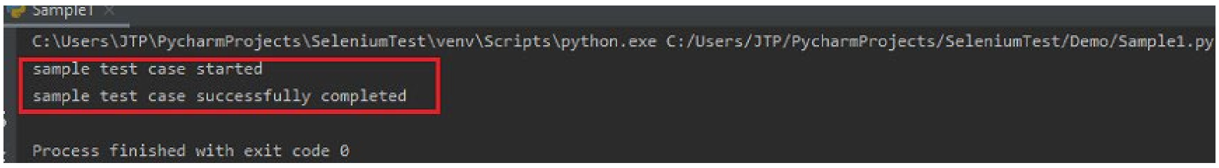
As we can see in the below screenshot, if we do not put r in the code, it will generate the **Syntax Error**.



```
Sample1
C:\Users\JTP\PycharmProjects\SeleniumTest\venv\Scripts\python.exe C:/Users/JTP/PycharmProjects/SeleniumTest/Demo/Sa
  File "C:/Users/JTP/PycharmProjects/SeleniumTest/Demo/Sample1.py", line 5
    driver = webdriver.Chrome("C:\Users\JTP\PycharmProjects\SeleniumTest\Browsers\chromedriver.exe")
                             ^
SyntaxError: (unicode error) 'unicodeescape' codec can't decode bytes in position 2-3: truncated \UXXXXXXXX escape

Process finished with exit code 1
```

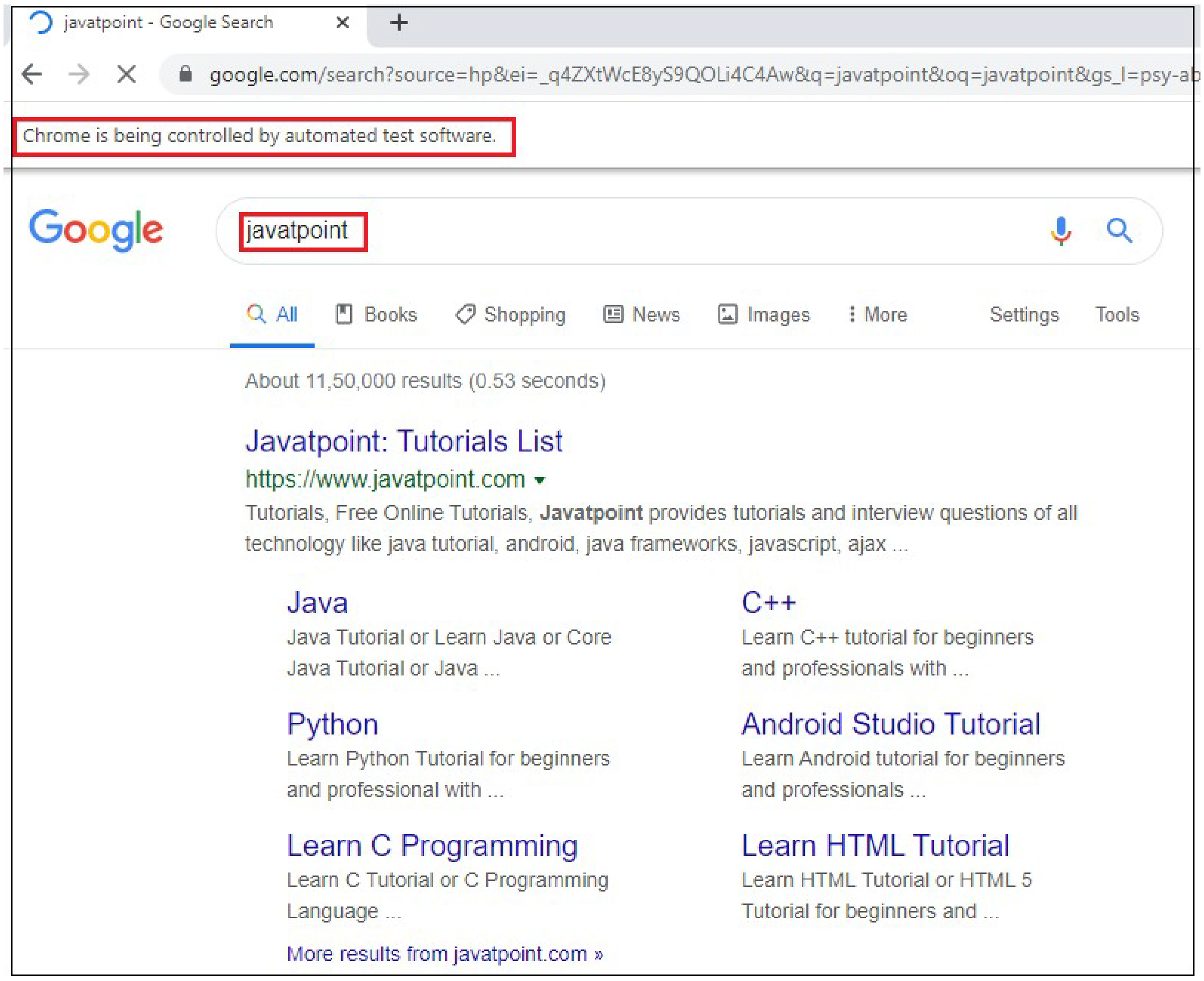
- After that, we will run the **sample1** once again, and it will execute the code successfully as we can see in the below image:



```
Sample1
C:\Users\JTP\PycharmProjects\SeleniumTest\venv\Scripts\python.exe C:/Users/JTP/PycharmProjects/SeleniumTest/Demo/Sample1.py
sample test case started
sample test case successfully completed

Process finished with exit code 0
```

The above test script will launch the Google Chrome browser and automate all the test scenarios.


javatpoint - Google Search

google.com/search?source=hp&ei=_q4ZXtWcE8yS9QOLi4C4Aw&q=javatpoint&oq=javatpoint&gs_l=psy-ab

Chrome is being controlled by automated test software.

javatpoint

All Books Shopping News Images More Settings Tools

About 11,50,000 results (0.53 seconds)

Javatpoint: Tutorials List
https://www.javatpoint.com
Tutorials, Free Online Tutorials, **Javatpoint** provides tutorials and interview questions of all technology like java tutorial, android, java frameworks, javascript, ajax ...

Java
Java Tutorial or Learn Java or Core Java Tutorial or Java ...

C++
Learn C++ tutorial for beginners and professionals with ...

Python
Learn Python Tutorial for beginners and professional with ...

Android Studio Tutorial
Learn Android tutorial for beginners and professionals ...

Learn C Programming
Learn C Tutorial or C Programming Language ...

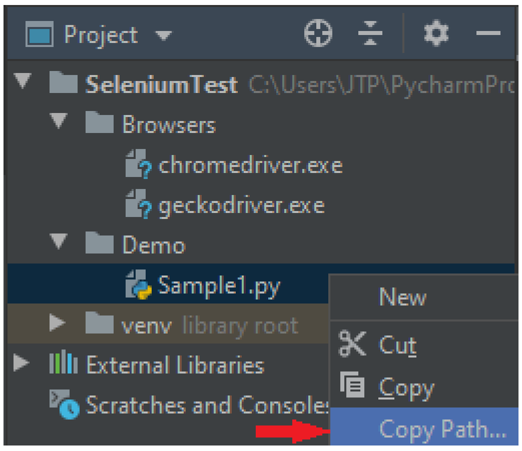
Learn HTML Tutorial
Learn HTML Tutorial or HTML 5 Tutorial for beginners and ...

More results from javatpoint.com »

Run in Command Prompt

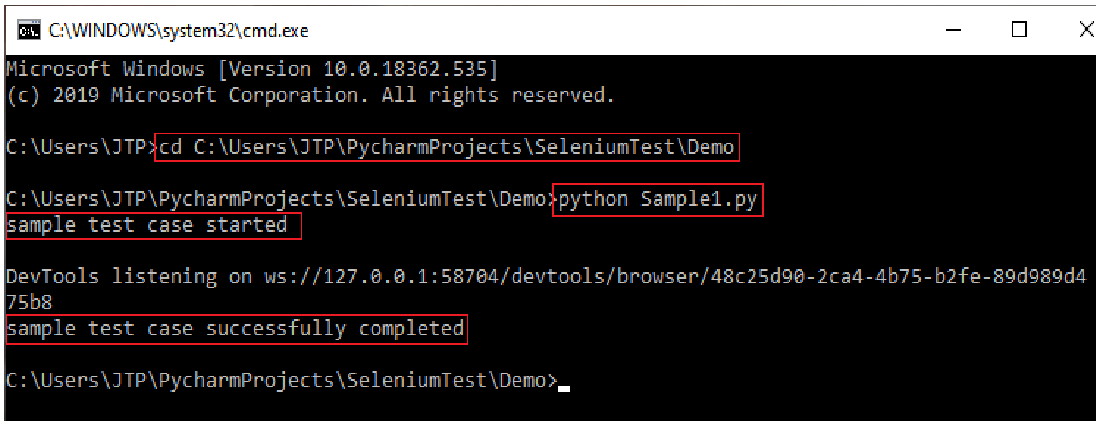
To run the above test script in the Command prompt, follow the below process:

- Copy the location of the **Sample1.py** file as we can see in the below image:



- And paste in the command Prompt, first go to the particular folder then enter the below command:
  Python Sample1.py

- Then, press the **Enter** key as we can see in the below screenshot that the **sample testcase stared.**
- And after automating all the scenarios, it will show the message as a **sample test casesuccessfully completed**.



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.18362.535]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\JTP>cd C:\Users\JTP\PycharmProjects\SeleniumTest\Demo

C:\Users\JTP\PycharmProjects\SeleniumTest\Demo>python Sample1.py
sample test case started

DevTools listening on ws://127.0.0.1:58704/devtools/browser/48c25d90-2ca4-4b75-b2fe-89d989d4
75b8
sample test case successfully completed

C:\Users\JTP\PycharmProjects\SeleniumTest\Demo>_
```

Conclusion: By the end of this experiment, we got a basic understanding of how to use Selenium with Python to test a web application. You will be able to set up a Python virtual environment for Selenium tests, write basic Selenium tests using Python, and integrate Selenium tests with Jenkins. You will alsobe able to expand on this by adding more tests, integrating with other tools, and configuring Jenkins to run tests automatically on a schedule or when code changes are pushed to the repository.