**Introduction to Python**

* General-purpose
* High level programming language.
* Guido van Rossum started implementing Python in 1989.
* It was initially designed by Guido van Rossum in 1991
* Developed by Python Software Foundation.

**Professionally**, Python is great for backend web development, data analysis, artificial intelligence, and scientific computing. Many developers have also used Python to build productivity tools, games, and desktop apps.

Interesting fact: Python is named after the comedy television show Monty Python’s Flying Circus. It is not named after the Python snake.

**Features of Python:**

1. Readable: Python is a very readable language.

2. Easy to Learn: Learning python is easy as it uses simple english words and high level programming language, which means it is easy to understand the language and thus easy to learn.

3. Cross platform: Python is available and can run on various operating systems such as Mac, Windows, Linux, Unix etc. This makes it a cross platform and portable language.

4. Open Source: Python is a open source programming language.

5. Large standard library: Python comes with a large standard library that has some handy codes and functions which we can use while writing code in Python.

6. Free: Python is free to download and use. This means you can download it for free and use it in your application. See: Open Source Python License. Python is an example of a FLOSS (Free/Libre Open Source Software), which means you can freely distribute copies of this software, read its source code and modify it.

7. Supports exception handling: An exception is an event that can occur during program exception and can interrupt the normal flow of program. Python supports exception handling which means we can write less error prone code and can test various scenarios that can cause an exception later on.

8. Advanced features: Supports generators and list comprehensions.

9. Automatic memory management: Python supports automatic memory management which means the memory is cleared and freed automatically. You do not have to bother clearing the memory.

**Applications of Python:**

* Web development
* Machine learning
* Data Analysis
* Scripting
* Game development
* Embedded applications
* Desktop applications

1. Web development – Web framework like Django and Flask are based on Python. They help you write server side code which helps you manage database, write backend programming logic, mapping urls etc.

2. Machine learning – There are many machine learning applications written in Python. Machine learning is a way to write a logic so that a machine can learn and solve a particular problem on its own. For example, products recommendation in websites like Amazon, Flipkart, eBay etc. is a machine learning algorithm that recognises user’s interest. Face recognition and Voice recognition in your phone is another example of machine learning.

3. Data Analysis – Data analysis and data visualisation in form of charts can also be developed using Python.

4. Scripting – Scripting is writing small programs to automate simple tasks such as sending automated response emails etc. Such type of applications can also be written in Python programming language.

5. Game development – You can develop games using Python.

6. You can develop Embedded applications in Python.

7. Desktop applications – You can develop desktop application in Python using library like TKinter or QT.

# How to Install Python

You can install Python on any operating system such as Windows, Mac OS X, Linux/Unix and others.

To install the Python on your operating system, go to this link: <https://www.python.org/downloads/>. You will see a screen like this.

This is the official Python website and it will detect the operating system and based on that it would recommend you to download Python. I would recommend you to download the latest version of Python 3 (Python 3.8.1).

Installation steps are pretty simple. You just have to accept the agreement and finish the installation.

There are two major Python versions- Python 2 and Python 3. Both are quite different.

To check if you have python installed on a Windows PC, search in the start bar for Python or run the following on the Command Line (cmd.exe):

C:\Users\Your Name>python --version

To check if you have python installed on a Linux or Mac, then on linux open the command line or on Mac open the Terminal and type:

python --version

Python is an interpreted programming language, this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

C:\Users\Your Name>python helloworld.py

Where "helloworld.py" is the name of your python file.

Let's write our first Python file, called helloworld.py, which can be done in any text editor.

helloworld.py

print("Hello, World!")

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

C:\Users\Your Name>python helloworld.py

The output should read:

Hello, World!

Congratulations, you have written and executed your first Python program.

## The Python Command Line

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.

Type the following on the Windows, Mac or Linux command line:

C:\Users\Your Name>python

Or, if the "python" command did not work, you can try "py":

C:\Users\Your Name>py

From there you can write any python, including our hello world example from earlier in the tutorial:

C:\Users\Your Name>python  
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32  
Type "help", "copyright", "credits" or "license" for more information.  
>>> print("Hello, World!")

Which will write "Hello, World!" in the command line:

C:\Users\Your Name>python  
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32  
Type "help", "copyright", "credits" or "license" for more information.  
>>> print("Hello, World!")  
Hello, World!

Whenever you are done in the python command line, you can simply type the following to quit the python command line interface:

exit()

# Python - Environment Setup

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

Open a terminal window and type "python" to find out if it is already installed and which version is installed.

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python <https://www.python.org/>

You can download Python documentation from <https://www.python.org/doc/>

quick overview of installing Python on various platforms −

### Unix and Linux Installation

Here are the simple steps to install Python on Unix/Linux machine.

* Open a Web browser and go to <https://www.python.org/downloads/>.
* Follow the link to download zipped source code available for Unix/Linux.
* Download and extract files.
* Editing the *Modules/Setup* file if you want to customize some options.
* run ./configure script
* make
* make install

This installs Python at standard location */usr/local/bin* and its libraries at */usr/local/lib/pythonXX* where XX is the version of Python.

### Windows Installation

Here are the steps to install Python on Windows machine.

* Open a Web browser and go to <https://www.python.org/downloads/>.
* Follow the link for the Windows installer *python-XYZ.msi* file where XYZ is the version you need to install.
* To use this installer *python-XYZ.msi*, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
* Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

### Macintosh Installation

Recent Macs come with Python installed, but it may be several years out of date. See [http://www.python.org/download/mac/](https://www.python.org/download/mac/) for instructions on getting the current version along with extra tools to support development on the Mac. For older Mac OS's before Mac OS X 10.3 (released in 2003), MacPython is available.

Jack Jansen maintains it and you can have full access to the entire documentation at his website − <http://www.cwi.nl/~jack/macpython.html>. You can find complete installation details for Mac OS installation.

## Setting up PATH

Programs and other executable files can be in many directories, so operating systems provide a search path that lists the directories that the OS searches for executables.

The path is stored in an environment variable, which is a named string maintained by the operating system. This variable contains information available to the command shell and other programs.

The **path** variable is named as PATH in Unix or Path in Windows (Unix is case sensitive; Windows is not).

In Mac OS, the installer handles the path details. To invoke the Python interpreter from any particular directory, you must add the Python directory to your path.

## Setting path at Unix/Linux

To add the Python directory to the path for a particular session in Unix −

* **In the csh shell** − type setenv PATH "$PATH:/usr/local/bin/python" and press Enter.
* **In the bash shell (Linux)** − type export PATH="$PATH:/usr/local/bin/python" and press Enter.
* **In the sh or ksh shell** − type PATH="$PATH:/usr/local/bin/python" and press Enter.
* **Note** − /usr/local/bin/python is the path of the Python directory

## Setting path at Windows

To add the Python directory to the path for a particular session in Windows −

**At the command prompt** − type path %path%;C:\Python and press Enter.

**Note** − C:\Python is the path of the Python directory

## Python Environment Variables

Here are important environment variables, which can be recognized by Python −

|  |  |
| --- | --- |
| Sr.No. | Variable & Description |
| 1 | **PYTHONPATH**  It has a role similar to PATH. This variable tells the Python interpreter where to locate the module files imported into a program. It should include the Python source library directory and the directories containing Python source code. PYTHONPATH is sometimes preset by the Python installer. |
| 2 | **PYTHONSTARTUP**  It contains the path of an initialization file containing Python source code. It is executed every time you start the interpreter. It is named as .pythonrc.py in Unix and it contains commands that load utilities or modify PYTHONPATH. |
| 3 | **PYTHONCASEOK**  It is used in Windows to instruct Python to find the first case-insensitive match in an import statement. Set this variable to any value to activate it. |
| 4 | **PYTHONHOME**  It is an alternative module search path. It is usually embedded in the PYTHONSTARTUP or PYTHONPATH directories to make switching module libraries easy. |

## Running Python

There are three different ways to start Python −

### Interactive Interpreter

You can start Python from Unix, DOS, or any other system that provides you a command-line interpreter or shell window.

Enter **python** the command line.

Start coding right away in the interactive interpreter.

$python # Unix/Linux

or

python% # Unix/Linux

or

C:> python # Windows/DOS

Here is the list of all the available command line options −

|  |  |
| --- | --- |
| Sr.No. | Option & Description |
| 1 | **-d**  It provides debug output. |
| 2 | **-O**  It generates optimized bytecode (resulting in .pyo files). |
| 3 | **-S**  Do not run import site to look for Python paths on startup. |
| 4 | **-v**  verbose output (detailed trace on import statements). |
| 5 | **-X**  disable class-based built-in exceptions (just use strings); obsolete starting with version 1.6. |
| 6 | **-c cmd**  run Python script sent in as cmd string |
| 7 | **file**  run Python script from given file |

### Script from the Command-line

A Python script can be executed at command line by invoking the interpreter on your application, as in the following −

$python script.py # Unix/Linux

or

python% script.py # Unix/Linux

or

C: >python script.py # Windows/DOS

**Note** − Be sure the file permission mode allows execution.

### Integrated Development Environment

You can run Python from a Graphical User Interface (GUI) environment as well, if you have a GUI application on your system that supports Python.

* **Unix** − IDLE is the very first Unix IDE for Python.
* **Windows** − PythonWin is the first Windows interface for Python and is an IDE with a GUI.
* **Macintosh** − The Macintosh version of Python along with the IDLE IDE is available from the main website, downloadable as either MacBinary or BinHex'd files.

If you are not able to set up the environment properly, then you can take help from your system admin. Make sure the Python environment is properly set up and working perfectly fine.

**Note** − All the examples given in subsequent chapters are executed with Python 2.4.3 version available on CentOS flavor of Linux.

We already have set up Python Programming environment online, so that you can execute all the available examples online at the same time when you are learning theory. Feel free to modify any example and execute it online.