**SOFTWARE DESCRIPTION**

## Introduction to Python

In today’s blog, we will provide an introduction to Python programming. This blog is dedicated to all those who are from any domain whether they are students, working employees, mechanical engineers who are willing to learn to program, and newbies. Python is the most widely used programming language by tech giants like Google, Netflix, Facebook. Therefore, it is essential to get started with Python programming. Looks interesting right, then let’s get started.

## What is Python Programming?

With an Introduction to Python, we can understand that it is a high-level object-oriented programming language that was created by Guido van Rossum. Python is also known as a general-purpose programming language, as it is used in the domains given below:

* Web Development
* Software Development
* Game Development
* AI & ML
* Data Analytics

This list can go on as we go but why python is so much popular let’s see it in the next topic.

## Why Python Programming?

Every programming language serves some purpose or use-case according to a domain, and Python is no exception. An introduction to Python programming can help us understand the purpose of the language. For example, Python is widely used in data science, machine learning, and artificial intelligence due to its simplicity and powerful libraries. Similarly, JavaScript is the most popular language among web developers as it gives the developer the power to handle applications via different frameworks like React, Vue, Angular, which are used to build beautiful user interfaces. Similarly, they have pros and cons at the same time. so if we consider python it is general-purpose which means it is widely used in every domain the reason is it’s very simple to understand, scalable because of which the speed of development is so fast. Now you get the idea why besides learning python it doesn’t require any programming background so that’s why it’s popular amongst developers as well. Python has simpler syntax similar to the English language and also the syntax allows developers to write programs with fewer lines of code. Since it is open-source there are many libraries available that make developers’ jobs easy ultimately results in high productivity. They can easily focus on business logic and Its demanding skills in the digital era where information is available in large data sets.

### How do we get started Python Programming?

Now, in the era of the digital world, there is a lot of information available on the internet that might confuse us when we are trying to learn something new. However, with an introduction to Python programming, we can make a good start. What we can do is follow the documentation which is a good starting point for learning Python. Once we are familiar with Python concepts or terminology, we can dive deeper into the language and explore more advanced topics..

### Real-World Examples:

1) **NASA (National Aeronautics and Space Agency):** One of Nasa’s Shuttle Support Contractors, United Space Alliance developed a Workflow Automation System (WAS) which is fast. Internal Resources Within critical project stated that:

“Python allows us to tackle the complexity of programs like the WAS without getting bogged down in the language”.

Nasa also published a website (https://code.nasa.gov/) where there are 400 open source projects which use python.

2) **Netflix**: There are various projects in Netflix which use python as follow:

* Central Alert Gateway
* Chaos Gorilla
* Security Monkey
* Chronos

Amongst all projects, Regional failover is the project they have as the system decreases outage time from 45 minutes to 7 minutes with no additional cost.

3) **Instagram:**Instagram also uses python extensively. They have built a photo-sharing social platform using Django which is a web framework for python. Also, they are able to successfully upgrade their framework without any technical challenges.

### Applications of Python Programming:

1) **Web Development**: Python offers different frameworks for web development like Django, Pyramid, Flask. This framework is known for security, flexibility, scalability.

2) **Game Development:** PySoy and PyGame are two python libraries that are used for game development

3) **Artificial Intelligence and Machine Learning:** There is a large number of open-source libraries which can be used while developing AI/ML applications.

4) **Desktop GUI:** Desktop GUI offers many toolkits and frameworks using which we can build desktop applications.PyQt, PyGtk, PyGUI are some of the GUI frameworks.

**Numpy**

## What is NumPy?

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python.

## Why Use NumPy?

In Python we have lists that serve the purpose of arrays, but they are slow to process.

NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.

Arrays are very frequently used in data science, where speed and resources are very important.

## Why is NumPy Faster Than Lists?

NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently. This behavior is called locality of reference in computer science. This is the main reason why NumPy is faster than lists. Also it is optimized to work with latest CPU architectures.

## Which Language is NumPy written in?

NumPy is a Python library and is written partially in Python, but most of the parts that require fast computation are written in C or C++.

**Pandas**

## What is Pandas?

Pandas is a Python library used for working with data sets.

It has functions for analyzing, cleaning, exploring, and manipulating data.

The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.

## Why Use Pandas?

Pandas allows us to analyze big data and make conclusions based on statistical theories.

Pandas can clean messy data sets, and make them readable and relevant.

Relevant data is very important in data science.

## What Can Pandas Do?

Pandas gives you answers about the data. Like:

* Is there a correlation between two or more columns?
* What is average value?
* Max value?
* Min value?

Pandas are also able to delete rows that are not relevant, or contains wrong values, like empty or NULL values. This is called cleaning the data.

## What is Python Pandas used for?

The Pandas library is generally used for data science, but have you wondered why? This is because the Pandas library is used in conjunction with other libraries that are used for data science.

It is built on top of the **NumPy library**which means that a lot of the structures of NumPy are used or replicated in Pandas.

The data produced by Pandas is often used as input for plotting functions in **[Matplotlib](https://www.geeksforgeeks.org/python-introduction-matplotlib/)**, statistical analysis in **SciPy**, and[machine learning algorithms](https://www.geeksforgeeks.org/machine-learning-algorithms/) in **[Scikit-learn](https://www.geeksforgeeks.org/learning-model-building-scikit-learn-python-machine-learning-library/)**.

You must be wondering, Why should you use the Pandas Library. Python’s Pandas library is the best tool to analyze, clean, and manipulate data.

Here is a list of things that we can do using Pandas.

* Data set cleaning, merging, and joining.
* Easy handling of missing data (represented as NaN) in floating point as well as non-floating point data.
* Columns can be inserted and deleted from DataFrame and higher-dimensional objects.
* Powerful group by functionality for performing split-apply-combine operations on data sets.
* Data Visualization.

## Getting Started with Pandas

Let’s see how to start working with the Python Pandas library:

### Installing Pandas

The first step in working with Pandas is to ensure whether it is installed in the system or not.  If not, then we need to install it on our system usingthe **pip command**.

Follow these steps to install Pandas:

**Step 1:** Type ‘cmd’ in the search box and open it.

**Step 2:** Locate the folder using the cd command where the **python-pip file** has been installed.  
**Step 3**: After locating it, type the command:

## Data Structures in Pandas Library

Pandas generally provide two data structures for manipulating data. They are:

* **Series**
* **DataFrame**

## ****Pandas Series****

A Pandas Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, Python objects, etc.). The axis labels are collectively called **indexes**.

The Pandas Series is nothing but a column in an Excel sheet. Labels need not be unique but must be of a hashable type.

The object supports both integer and label-based indexing and provides a host of methods for performing operations involving the index.

## Conclusion

This tutorial provides a solid foundation for mastering the Pandas library, from basic operations to advanced techniques. We have also covered the Pandas data structures (series and DataFrame) with examples.

After completing this tutorial, you will gain a complete idea of what is Python Pandas. What is Pandas used for? and how to use Python Pandas.

As you apply these skills to your projects, you will discover how Pandas enhances your ability to explore, clean, and analyze data, making it an indispensable tool in the data scientist’s toolkit.

**Matplotlib**

**Matplotlib is a cross-platform, data visualization and graphical plotting library (histograms, scatter plots, bar charts, etc) for Python and its numerical extension NumPy. As such, it offers a viable open source alternative to MATLAB. Developers can also use matplotlib’s APIs (Application Programming Interfaces) to embed plots in GUI applications.**

A Python matplotlib script is structured so that a few lines of code are all that is required in most instances to generate a visual data plot. The matplotlib scripting layer overlays two APIs:

* The pyplot API is a hierarchy of Python code objects topped by matplotlib.pyplot
* An OO (Object-Oriented) API collection of objects that can be assembled with greater flexibility than pyplot. This API provides direct access to Matplotlib’s backend layers.

### Matplotlib and Pyplot in Python

The pyplot API has a convenient MATLAB-style stateful interface. In fact, the matplotlib Python library was originally written as an open source alternative for MATLAB. The OO API and its interface is more customizable and powerful than pyplot, but considered more difficult to use. As a result, the pyplot interface is more commonly used, and is referred to by default in this article.

Understanding matplotlib’s pyplot API is key to understanding how to work with plots:

* **matplotlib.pyplot.figure: Figure** is the top-level container. It includes everything visualized in a plot including one or more **Axes**.
* **matplotlib.pyplot.axes**:**Axes** contain most of the elements in a plot**: Axis, Tick, Line2D, Text,**etc., and sets the coordinates. It is the area in which data is plotted. Axes include the X-Axis, Y-Axis, and possibly a Z-Axis, as well.

For more information about the pyplot API and interface, refer to [***What Is Pyplot In Matplotlib***](https://www.activestate.com/resources/quick-reads/how-to-display-a-plot-in-python/)

### Installing Matplotlib

Matplotlib and its dependencies can be downloaded as a binary (pre-compiled) package from the Python Package Index (PyPI), and installed with the following command:

Matplotlib is also available as uncompiled source files from GitHub. Compiling from source will require your local system to have the appropriate compiler for your OS, all dependencies, setup scripts, configuration files, and patches available. This can result in a fairly complex installation. Alternatively, consider using the [ActiveState Platform](https://platform.activestate.com/create-account" \t "_blank) to automatically build matplotlib from source and package it for your OS.

### Matplotlib UI Menu

When matplotlib is used to create a plot, a User Interface (UI) and menu structure are generated. The UI can be used to customizing the plot, as well as to pan/zoom and toggle various elements.

### Matplotlib and NumPy

Numpy is a package for scientific computing. Numpy is a required dependency for matplotlib, which uses numpy functions for numerical data and multi-dimensional arrays as shown in the following code snippet:

### Matplotlib and Pandas

Pandas is a library used by matplotlib mainly for data manipulation and analysis. Pandas provides an in-memory 2D data table object called a Dataframe. Unlike numpy, pandas is not a required dependency of matplotlib.

#### Why use ActiveState Python for Data Science

While the open source distribution of Python may be satisfactory for an individual, it doesn’t always meet the support, security, or platform requirements of large organizations.

This is why organizations choose ActiveState Python for their data science, big data processing, statistical analysis needs and Python programming needs.

Pre-bundled with the most important packages Data Scientists need, ActiveState Python is pre-compiled so you and your team don’t have to waste time configuring the open source distribution. You can focus on what’s important–spending more time building algorithms and predictive models against your big data sources, and less time on system configuration.

ActiveState Python is 100% compatible with the open source Python distribution, and provides the security and commercial support that your organization requires.

With ActiveState Python you can explore and manipulate data, run statistical analysis, perform Python plotting and deliver visualizations to share insights with your business users and executives sooner–no matter where your data lives.