Python is simple to use, but it is a real programming language, offering much more structure and support for large programs than shell scripts or batch files can offer. On the other hand, Python also offers much more error checking than C, and, being a *very-high-level language*, it has high-level data types built in, such as flexible arrays and dictionaries. Because of its more general data types of Python is applicable to a much larger problem domain than Awk or even Perl, yet many things are at least as easy in Python as in those languages.

Python allows you to split your program into modules that can be reused in other Python programs. It comes with a large collection of standard modules that you can use as the basis of your programs — or as examples to start learning to program in Python. Some of these modules provide things like file I/O, system calls, sockets, and even interfaces to graphical user interface toolkits like Tk.

Python is an interpreted language, which can save you considerable time during program development because no compilation and linking is necessary. The interpreter can be used interactively, which makes it easy to experiment with features of the language, to write throw-away programs, or to test functions during bottom-up program development. It is also a handy desk calculator.

Python enables programs to be written compactly and readably. Programs written in Python are typically much shorter than equivalent C, C++, or Java programs, for several reasons:

- the high-level data types allow you to express complex operations in a single statement;
- statement grouping is done by indentation instead of beginning and ending brackets;
- no variable or argument declarations are necessary.

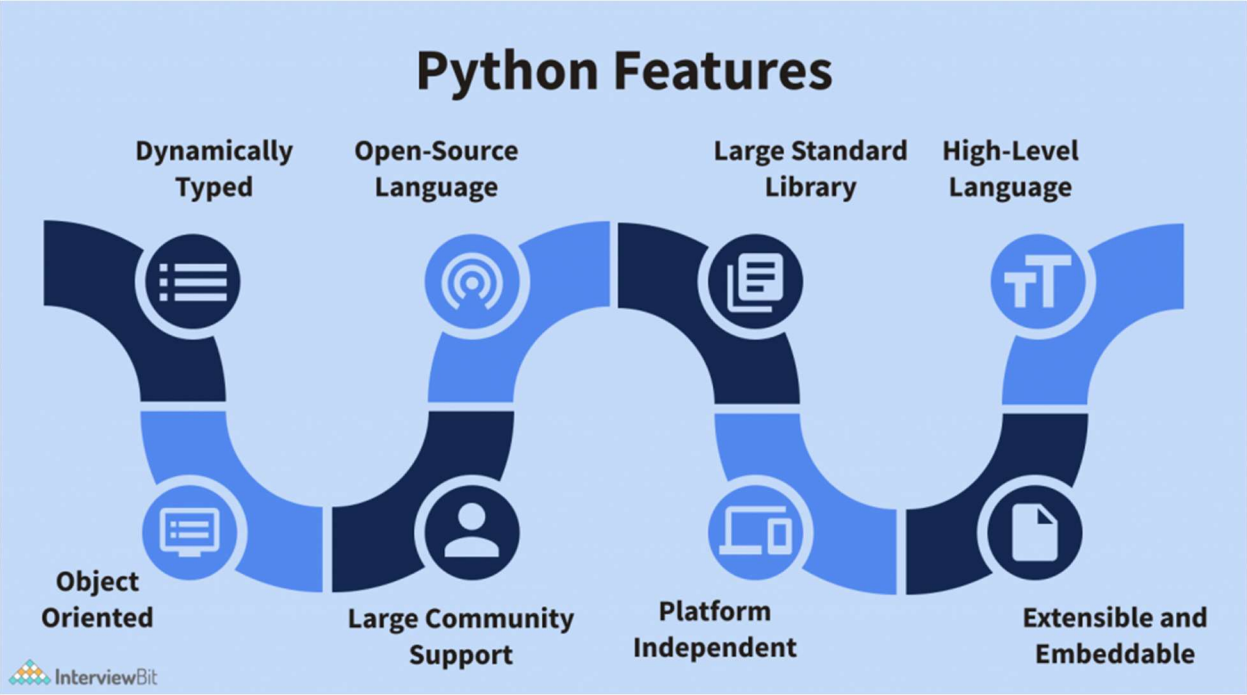
Python is *extensible*: if you know how to program in C it is easy to add a new built-in function or module to the interpreter, either to perform critical operations at maximum speed, or to link Python programs to libraries that may only be available in binary form (such as a vendor-specific graphics library). Once you are really hooked, you can link the Python interpreter into an application written in C and use it as an extension or command language for that application.

By the way, the language is named after the BBC show “Monty Python’s Flying Circus” and has nothing to do with reptiles. Making references to Monty Python skits in documentation is not only allowed, it is encouraged!

Invoking the Interpreter

The Python interpreter is usually installed as `/usr/local/bin/python3.11` on those machines where it is available; putting `/usr/local/bin` in your Unix shell's search path makes it possible to start it by typing the command:

Key Features Of Python



1. Easy To Learn and Readable Language

Python is extremely easy to learn. Its syntax is super simple and the learning curve of Python is very smooth. It is extremely easy to learn and code in Python and the indentation used instead of curly braces in Python makes it very easy to read Python code. Perhaps, because of this a lot of schools and universities, and colleges are teaching Python to their students who are beginning their journey with coding.

2. Interpreted Language

Python is an interpreted language (an interpreted language is a programming language that is generally interpreted, without compiling a program into machine instructions. It is one where the instructions are not directly executed by the target machine, but instead, read and executed by some other program known as the interpreter) and an IDLE (Interactive Development Environment) is packaged along with Python. It is nothing but an interpreter which follows the REPL (Read Evaluate Print Loop) structure just like in Node.js. IDLE executes and displays the output of one line of Python code at a time. Hence, it displays errors when we are running a line of Python code and displays the entire stack trace for the error.

3. Dynamically Typed Language

Python is a dynamically typed language. In other words, in Python, we do not need to declare the data types of the variables which we define. It is the job of the Python interpreter to determine the data types of the variables at runtime based on the types of the parts of the expression. Though it makes coding easier for programmers, this property might create runtime errors. To be specific, Python follows duck typing. It means that “If it looks like a duck, swims like a duck and quacks like a duck, it must be a duck.”

4. Open Source And Free

Python is an open-source programming language and one can download it for free from Python’s official website. The community of Python users is constantly contributing to the code of Python in order to improve it.

5. Large Standard Library

One of the very important features because of which Python is so famous in today’s times is the huge standard library it offers to its users. The standard library of Python is extremely large with a diverse set of packages and modules like itertools, functools, operator, and many more with common and important functionalities in them. If the code of some functionality is already present in these modules and packages, the developers do not need to rewrite them from scratch, saving both time and effort on the developer’s end. Moreover, the developers can now focus on more important things concerning their projects. Also, Python provides the PyPI (Python Package Index) which contains more packages that we can install and use if we want even more functionality.

6. High-Level Language

A high-level language (HLL) is a programming language that enables a programmer to write programs that are more or less independent of a particular type of computer. These languages are said to be high level since they are very close to human languages and far away from machine languages. Unlike C, Python is a high-level language. We can easily understand Python and it is closer to the user than middle-level languages like C. In Python, we do not need to remember system architecture or manage the memory.

7. Object Oriented Programming Language

Python supports various programming paradigms like structured programming, functional programming, and object-oriented programming. However, the most important fact is that the Object-Oriented approach of Python allows its users to implement the concepts of Encapsulation, Inheritance, Polymorphism, etc. which is extremely important for the coding done in most Software Industries as objects map to entities in the real world and a lot of real-world problems can be solved using the Object-Oriented Approach.

8. Large Community Support

With one of the biggest communities on StackOverflow and Meetup, Python has gained its popularity over the years. If we need any kind of help related to Python, the huge community is always there to answer our queries. A lot of questions about Python have already been answered on these sites and Python users can reference them as per requirement.

9. Platform Independent

Platform independence is yet another amazing feature of Python. In other words, it means that if we write a program in Python, it can run on a variety of platforms, for instance, Windows, Mac, Linux, etc. We do not have to write separate Python code for different platforms.

10. Extensible and Embeddable

Python is an Embeddable language. We can write some Python code into C or C++ language and also we can compile that code in C/C++ language. Python is also extensible. It means that we can extend our Python code in various other languages like C++, etc. too.

11. Graphical User Interface (GUI) Support

Yet another interesting feature of Python is the fact that we can use it to create GUI (Graphical User Interfaces). We can use Tkinter, PyQt, wxPython, or Pyside for doing the same. Python also features a large number of GUI frameworks available for it and various other cross-platform solutions. Python binds to platform-specific technologies.

Python Use Cases and Applications

1. Web Applications. Python aids in the rapid development of web applications that are both scalable and secure.
2. Data Science Implementations.
3. Artificial Intelligence.
4. Game Development.
5. Internet of things.
6. Enterprise applications.
7. Image Recognition and text processing.

Python Interactive Mode

When commands are read from a tty, the interpreter is said to be in *interactive mode*. In this mode it prompts for the next command with the *primary prompt*, usually three greater-than signs (>>>); for continuation lines it prompts with the *secondary prompt*, by default three dots (...). The interpreter prints a welcome message stating its version number and a copyright notice before printing the first prompt:

```
$ python3.11
Python 3.11 (default, April 4 2021, 09:25:04)
[GCC 10.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Continuation lines are needed when entering a multi-line construct. As an example, take a look at this if statement:

```
>>> the_world_is_flat = True
>>> if the_world_is_flat:
...     print("Be careful not to fall off!")
...
Be careful not to fall off!
```

Source Code Encoding

By default, Python source files are treated as encoded in UTF-8. In that encoding, characters of most languages in the world can be used simultaneously in string literals, identifiers and comments — although the standard library only uses ASCII characters for identifiers, a convention that any portable code should follow. To display all these characters properly, your editor must recognize that the file is UTF-8, and it must use a font that supports all the characters in the file.

Numbers

The interpreter acts as a simple calculator: you can type an expression at it and it will write the value. Expression syntax is straightforward: the operators `+`, `-`, `*` and `/` work just like in most other languages (for example, Pascal or C); parentheses (()) can be used for grouping. For example:

Strings

Besides numbers, Python can also manipulate strings, which can be expressed in several ways. They can be enclosed in single quotes (`'...'`) or double quotes (`"..."`) with the same result. `\` can be used to escape quotes:

In the interactive interpreter, the output string is enclosed in quotes and special characters are escaped with backslashes. While this might sometimes look different from the input (the enclosing quotes could change), the two strings are equivalent. The string is enclosed in double quotes if the string contains a single quote and no double quotes, otherwise it is enclosed in single quotes. The print() function produces a more readable output, by omitting the enclosing quotes and by printing escaped and special characters:

Literal Collections

In Python, there are 4 different types of Literal Collections. They represent more complicated and complex data and assist Python scripts to be more extensible. Let us look at each one of them in detail.

List Literals

The elements in a list are of many data types. The values in the List are surrounded by square brackets ([]) and separated by commas (,). List values can be changed i.e., they are mutable.

```
cars = ['Tata', 'BMW', 'Audi', 'Ferrari']
student = ['John', 20, 9876432113, 'Mumbai']

print(cars)
print(student)
```

Tuple

Just like a List, a tuple is also a collection of various data types. It is surrounded by parentheses '(),' and each element is separated by a comma (,). It is unchangeable (immutable).

```
num = (1, 2, 4, 5, 7, 8)
student = ('John', 20, 9876432113, 'Mumbai')

print(num)
print(student)
```

Dictionary

The data is stored in the dictionary as a key-value pair. It is surrounded by curly braces '{}', and each pair is separated by commas (,). A dictionary can hold various types of data. Dictionaries are subject to change.

```
student = {'Name':'David', 'Age':22, 'Sex':'Male', 'City':'California', }
print(student)

print(student.keys())
print(student.values()))
```

Python Flask :

One of the popular Python frameworks used by developers for web development is Flask. In this article, you will get introduced to Python Flask framework. Along with this, we will also see some of the basic implementations along with some HTTP methods.

A web framework is a software architecture that contains tools and libraries used to develop a web application in a fast and efficient way. Flask is a microframework written in Python. It was developed by Armin Ronacher and has a BSD license. It is based on the Werkzeug toolkit and Jinja2 template.

1. WSGI (Web Server Gateway Interface): It is used as a universal interface between the web server and the web application
2. **Werkzeug**: It is a WSGI toolkit and is used for implementing requests, response objects, and other utilities. It is used to build a web framework on top of it.
3. **Jinja2**: It is a templating engine that combines a template and a data source to develop a dynamic website.

It provides only core functionalities including form validation, upload handling, object-relational mappers, open authentication, etc. Using these, one can build both small- and large-scale websites. It does not have a database abstraction layer, form validation, and additional functionalities but it provides extensions to implement these.

Advantages of Python Flask

1. It is a lightweight and modular design
2. Contains a built-in development server and a fast debugger.
3. Provides integrated unit testing support
4. RESTful request dispatching.
5. Jinja2 Template.
6. Provides support for secure cookies.

Datatypes in Python :

There are different types of data types in Python. Some built-in Python data types are:

- **Numeric data types**: *int, float, complex*
- **String data types**: *str*
- **Sequence types**: *list, tuple, range*
- **Binary types**: *bytes, bytearray, memoryview*
- **Mapping data type**: *dict*
- **Boolean type**: *bool*
- **Set data types**: *set, frozenset*

Setting Flask Environment

Before we use the Flask module, we need to install the required packages. The prerequisite for this is that we need 2.6 or higher versions of Python installed on the device.

Then we need to install the Python virtual environment. We can do this by writing the below command in the command prompt.

Example of entailing virtualenv:

```
pip install virtualenv
```

If you have already installed it, then we get the below output.

Requirement already satisfied: virtualenv **in** c:\users\.....\lib\site-packages (20.7.2)

Now we create a folder and initiate the virtual environment in the folder. For this, we write the below codes.

```
D:\> mkdir flask_proj
D:\> cd flask_proj
D:\flask_proj> virtualenv env
```

Now we activate the environment.

```
D:\flask_proj> env\scripts\activate
(env) PS D:\flask_proj>
```

Now we install the flask package.

Installing Flask Package:

```
(env) PS D:\flask_proj> pip install flask
```

Python Flask Framework Implementation

Let us see a simple example of the web application that prints ‘PythonGeeks’.

Example of flask:

Let the below code be saved as ‘demo.py’ in the folder we created above.

```
from flask import Flask #importing the module

app=Flask(__name__) #instantiating flask object

@app.route('/') #defining a route in the application

def func(): #writing a function to be executed
```

```
return 'PythonGeeks'

if __name__=='__main__': #calling main

app.debug=True #setting the debugging option for the application instance

app.run() #launching the flask's integrated development webserver
```

Now in the command prompt, we change the directory to the folder and run the python file by writing the below command.

```
python demo.py
```

In this example,

1. On running the code in a virtual environment, Flask starts the server on 127.0.0.1 and port 5000 by default. To accept connection from any remote address, we can use the host as '0.0.0.0.'

2. On typing the requirements on the browser, the webserver sends the request to the Flask application instance

3. The application has information on the code that needs to run for each URL and it maps the URL to the Python functions

4. The route defined using the decorator @app.route() defines handles the URL and the associated function. It binds the URL '/' to the function func(). We can also use app.add_url_rule() function.

5. Using the run() function we start the flask's integrated development webserver