An Introduction to Python & Basic Python Syntax

By Aksadur Rahman

aksadur@yahoo.com

Agenda

A Brief History of Python Versions
Installing Python
Variables
Data Types
Dynamic Types
Python Reserved Words
Naming Conventions
Instruction/Statement
Basic Syntax Comments
Receiving Input
Type Conversion/Casting
Numeric Data Types
Boolean Data Types
Swapping
Strings

An Introduction to Python

An Introduction to Python

What is Python?

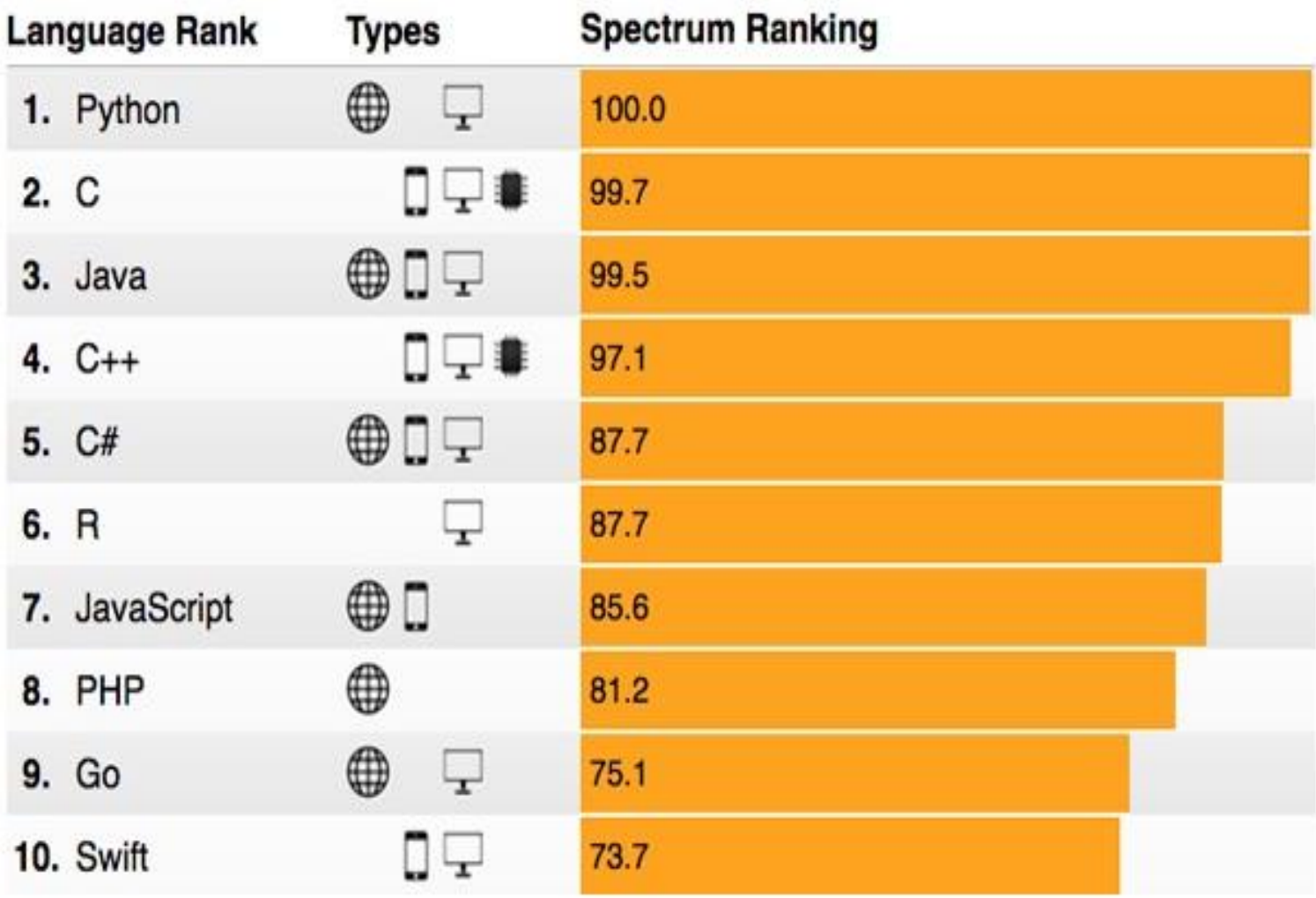
Python is a high-level object-oriented programming language that was created by Guido van Rossum. It is also called general-purpose programming language as it is used in almost every domain we can think of as mentioned below:

- ❑ Web Development
- ❑ Software Development
- ❑ Game Development
- ❑ AI & ML
- ❑ Data Analytics

An Introduction to Python

Why Python Programming?

IEEE spectrum list of top programming language 2021. The list of programming languages is based on popularity.



An Introduction to Python

Python is easy to understand

Java

```
class HelloWorld {
  static public void main( String args[] ) {
    System.out.println( "Hello World!" );
  }}
```

C++

```
#include <iostream.h>
main()
{
    cout << "Hello World!" << endl;
    return 0;
}
```

C#

```
class HelloWorld
{
    static void Main()
    {
        System.Console.WriteLine("Hello, World!");
    }
}
```

Python

```
print("Hello World")
```

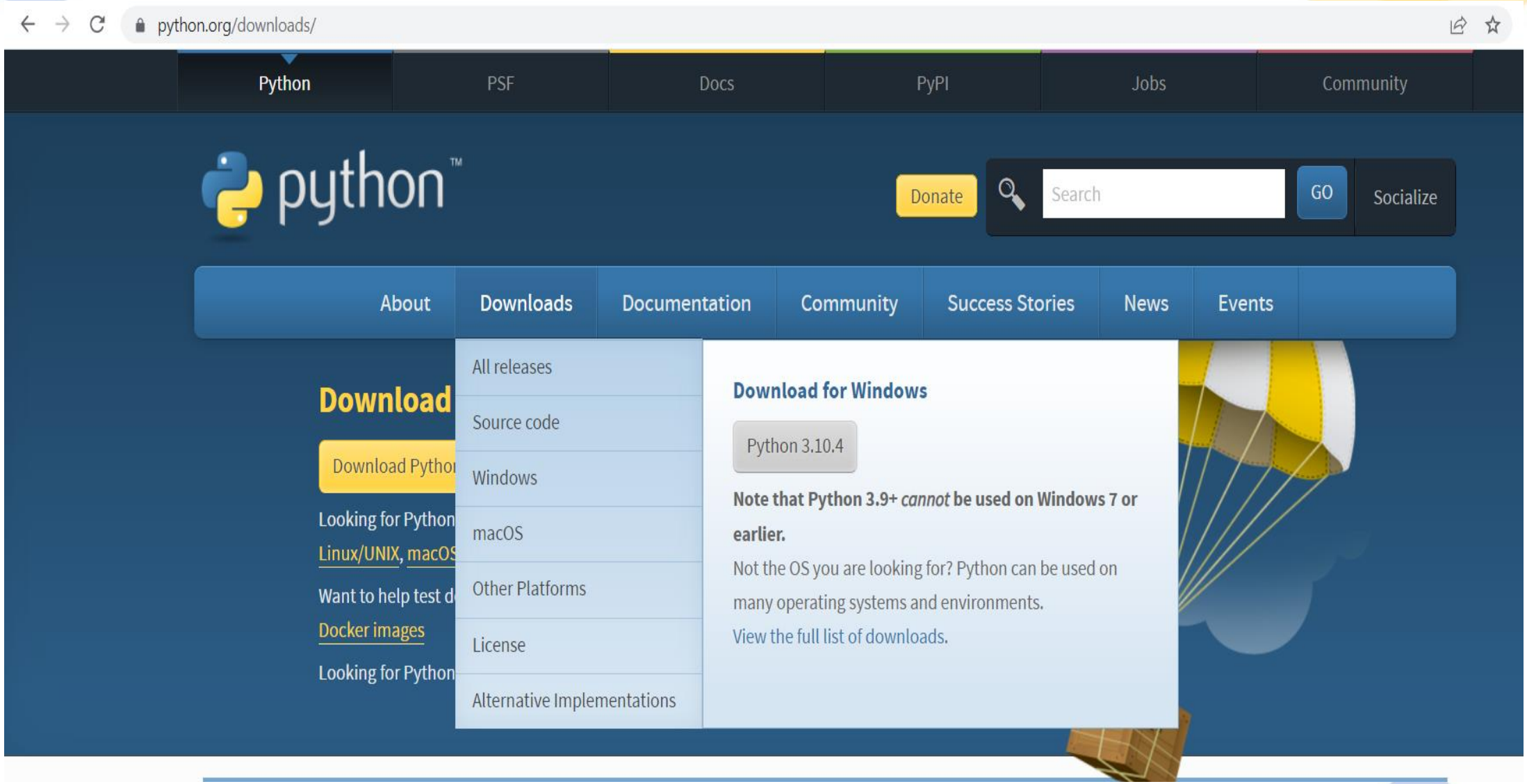
A Brief History of Python Versions

Python Version	Released Date
Python 1.0	January 1994
Python 1.5	December 31, 1997
Python 1.6	September 5, 2000
Python 2.0	October 16, 2000
Python 2.1	April 17, 2001
Python 2.2	December 21, 2001
Python 2.3	July 29, 2003
Python 2.4	November 30, 2004
Python 2.5	September 19, 2006
Python 2.6	October 1, 2008
Python 2.7	July 3, 2010
Python 3.0	December 3, 2008
Python 3.1	June 27, 2009
Python 3.2	February 20, 2011
Python 3.3	September 29, 2012
Python 3.4	March 16, 2014
Python 3.5	September 13, 2015
Python 3.6	December 23, 2016
Python 3.7	June 27, 2018
Python 3.8	October 14, 2019

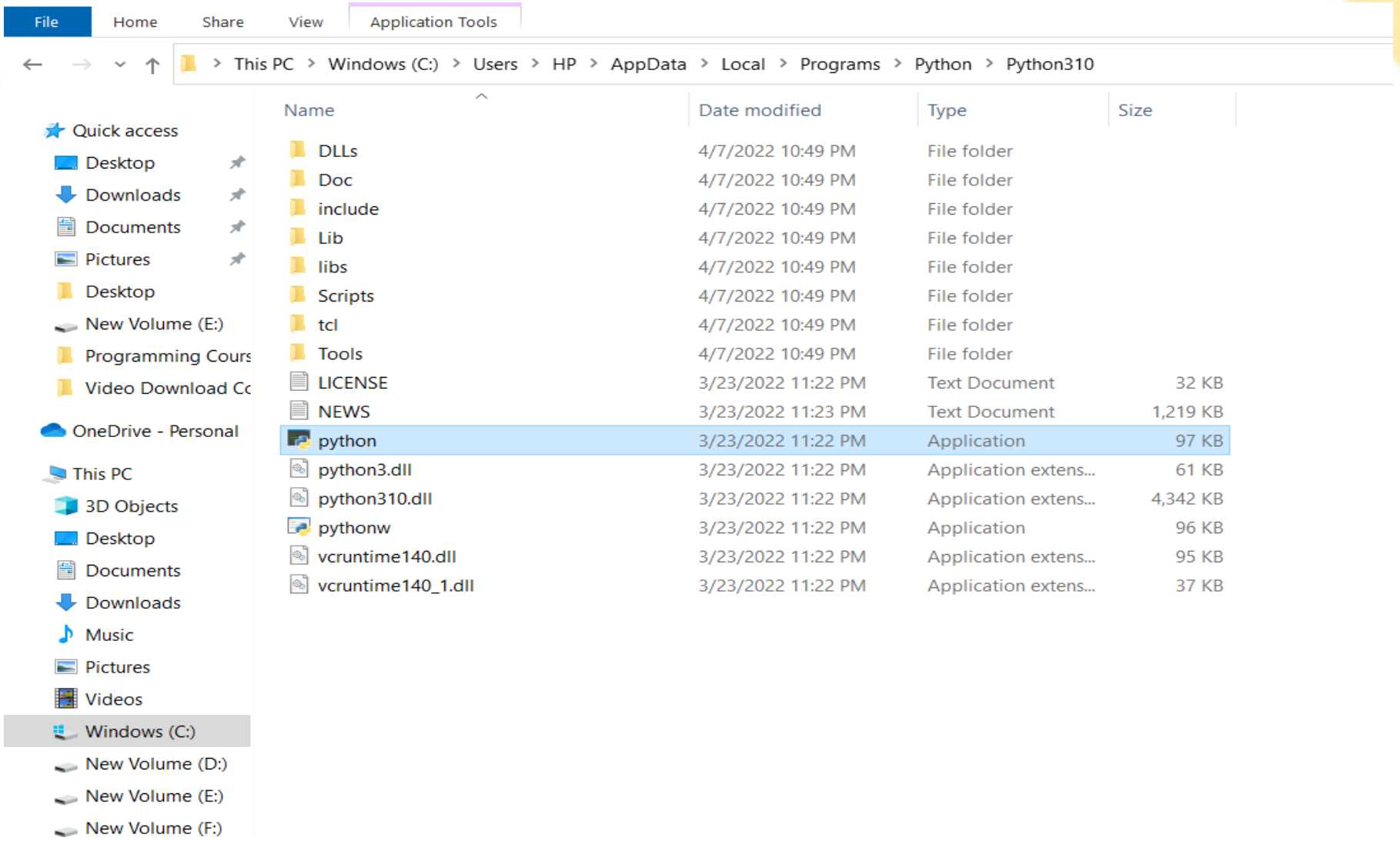
Python Software Foundation

First of all, there are the Pythons which are maintained by the people gathered around the PSF (Python Software Foundation), a community that aims to develop, improve, expand, and popularize Python and its environment. The PSF's president is Guido von Rossum himself

Installing Python

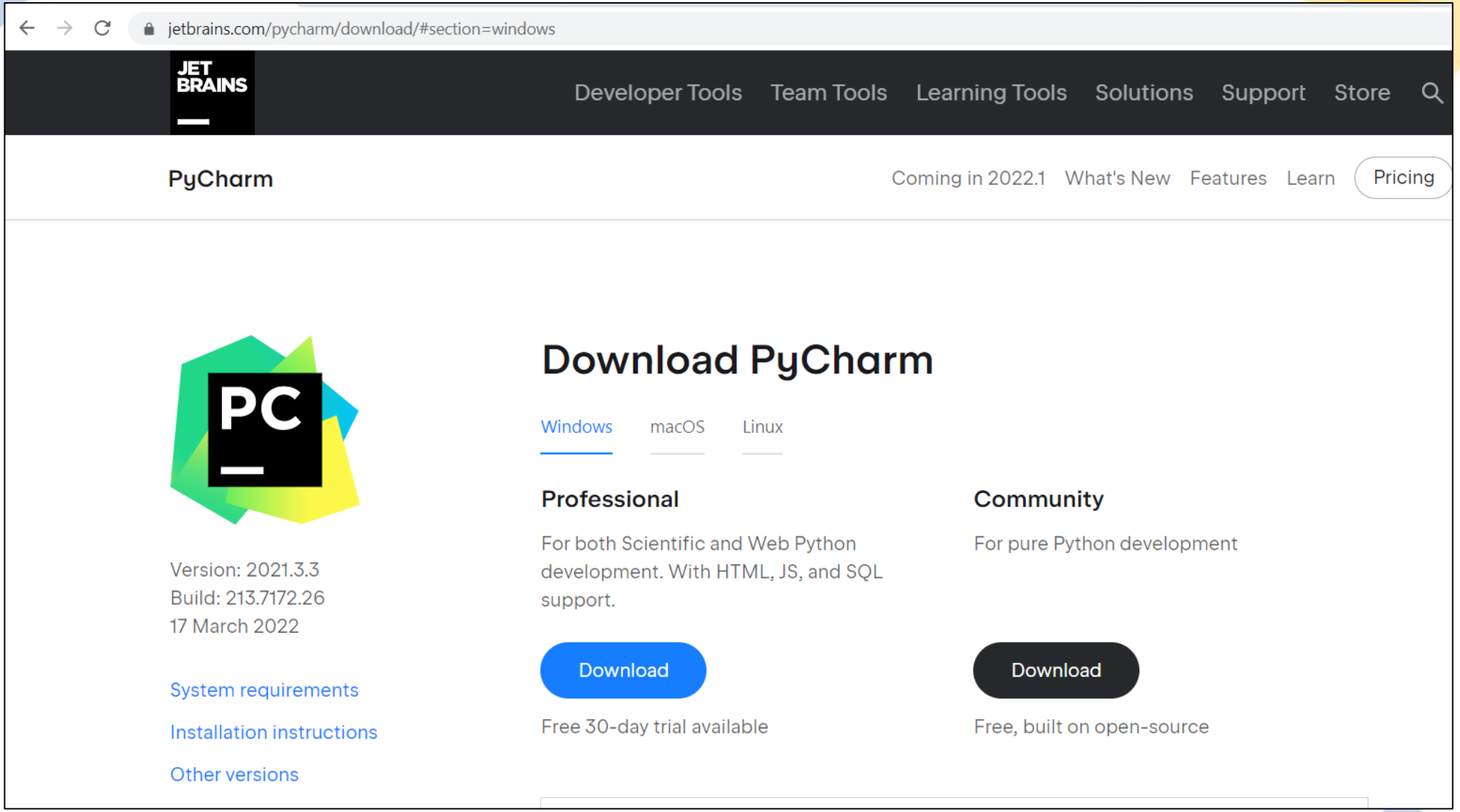


Installing Python

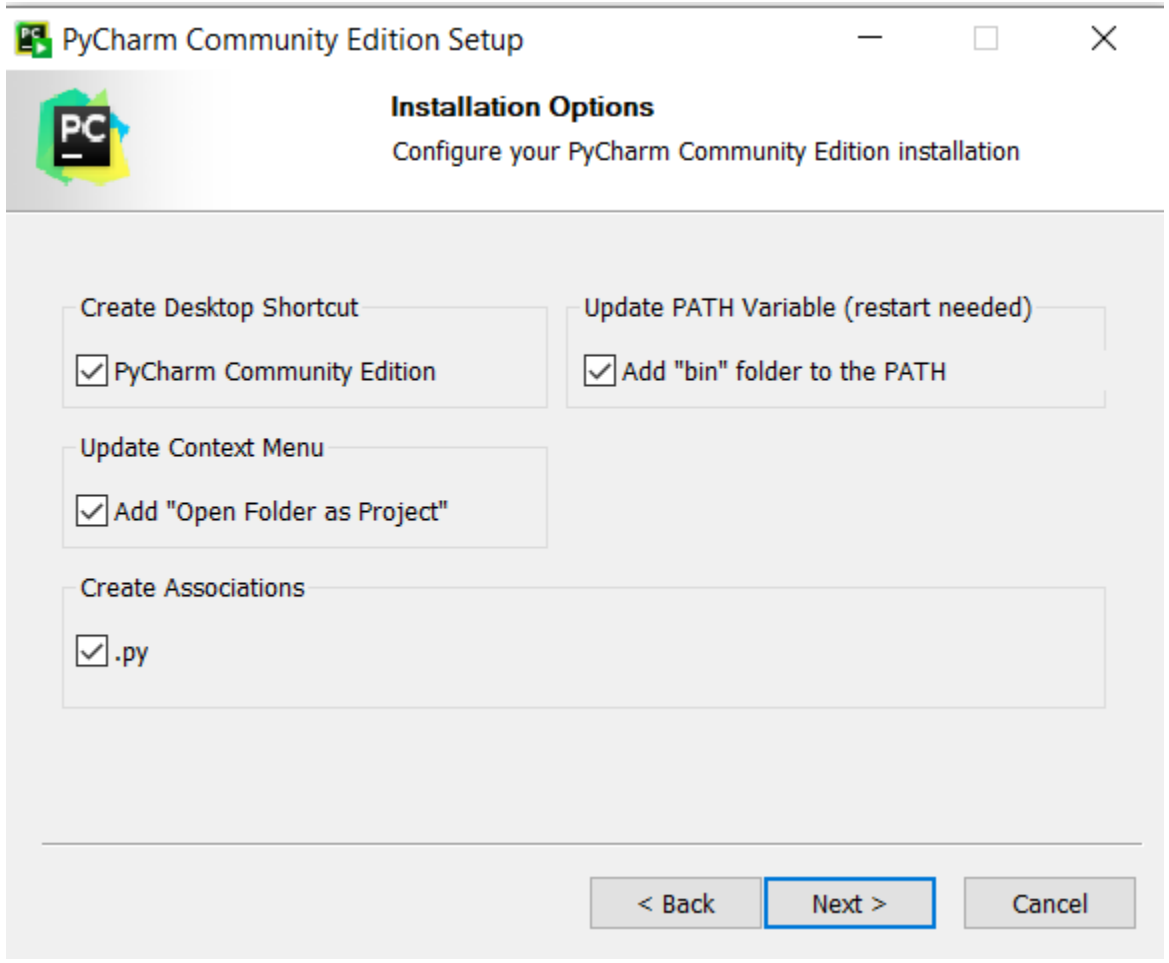


Installing Python

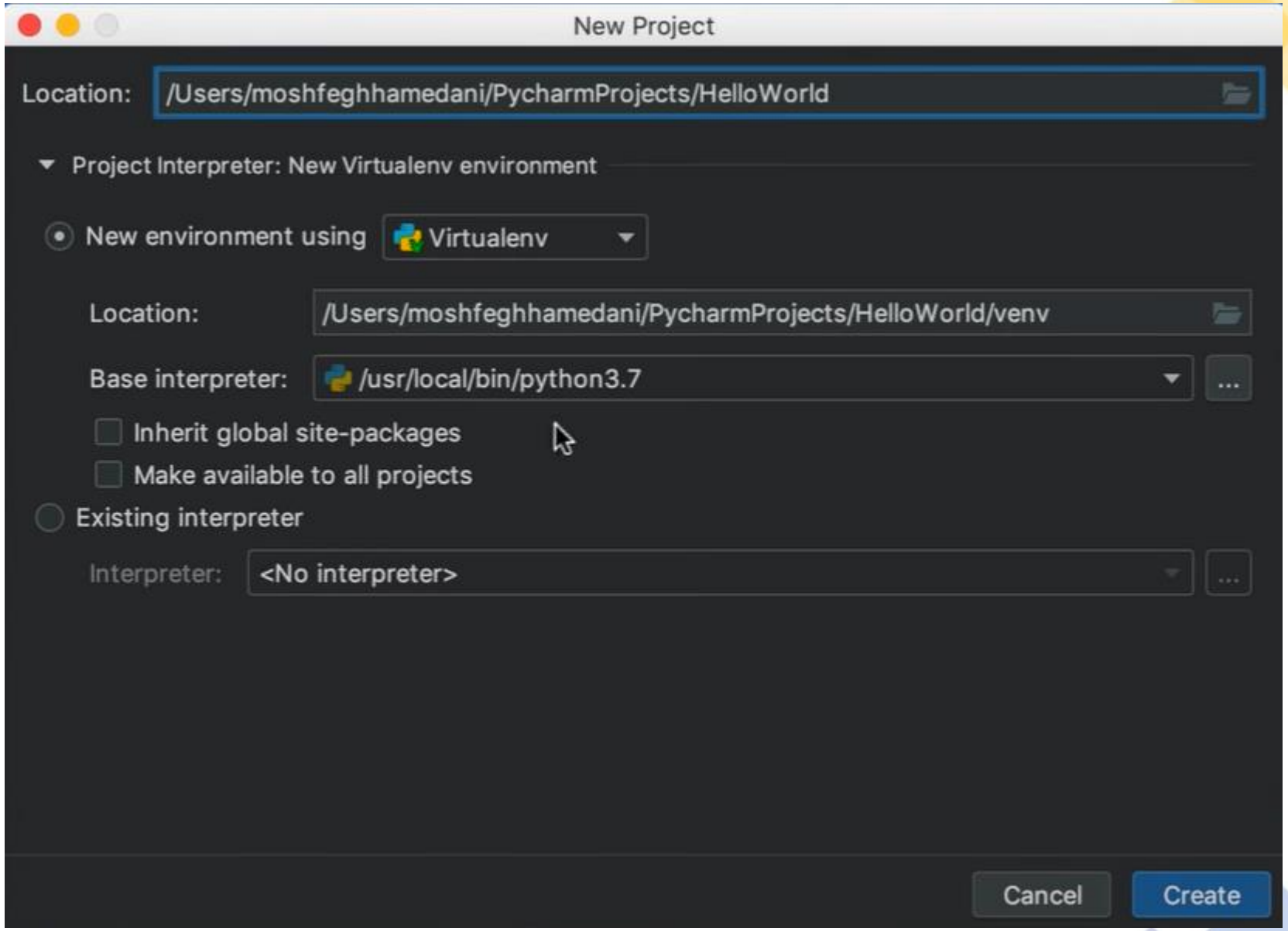
Installing PyCharm



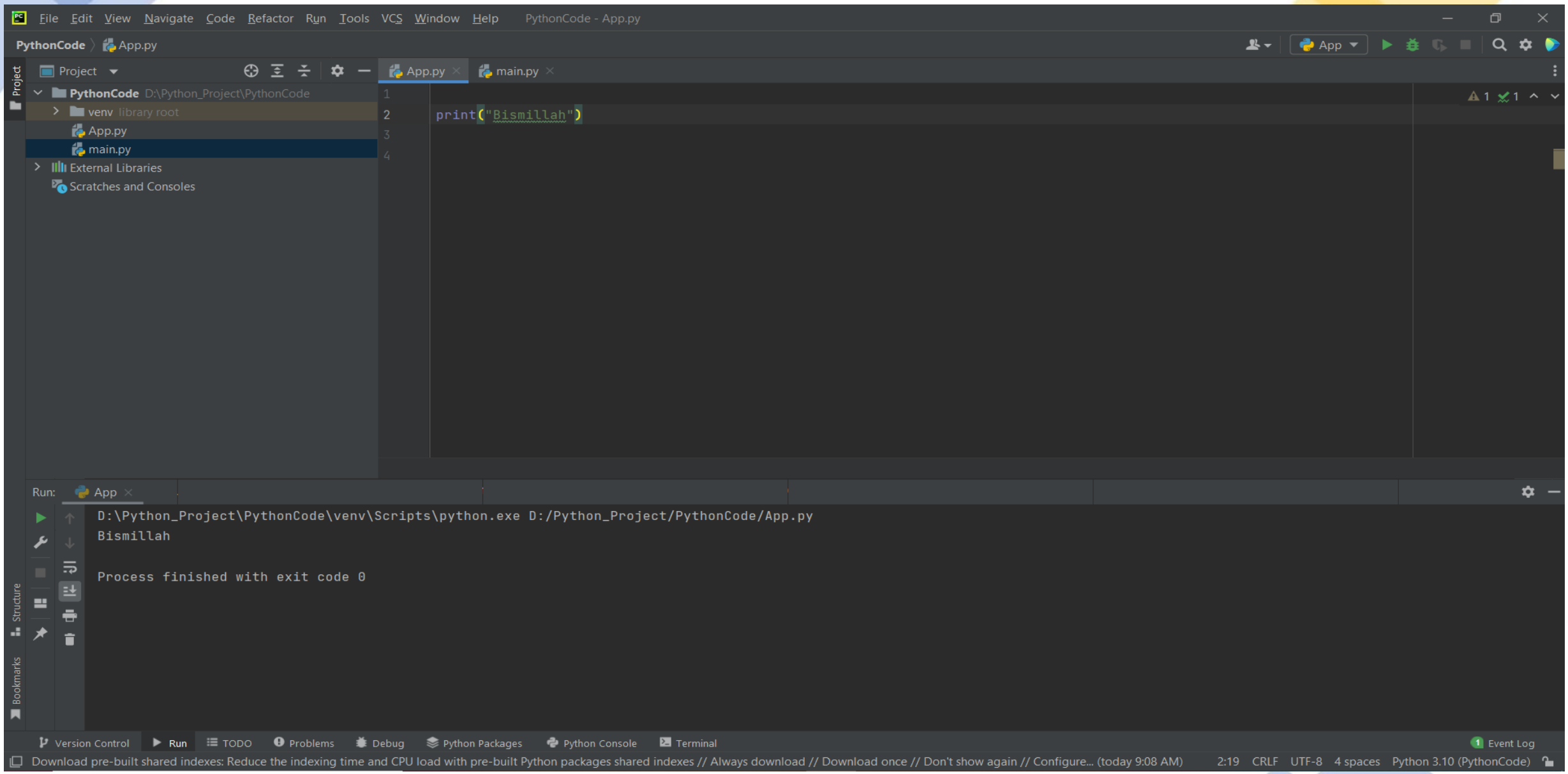
Installing PyCharm



Your First Python Program



Your First Python Program



Variables

Variable is a name that is used to refer to memory location. It stores and manipulates data.

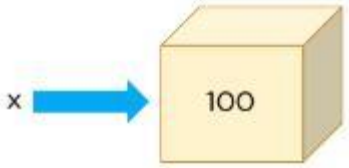
Variables are entities of a program that holds a value. Here is an example of a variable:

x=100

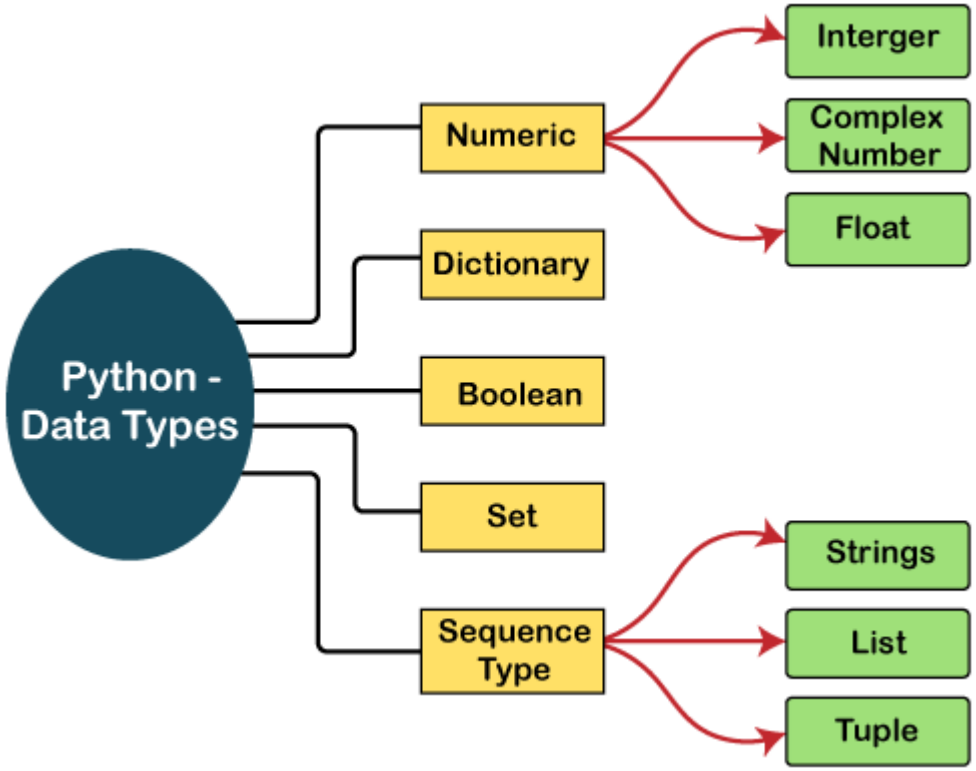
```
x=100

type(x)
int
```

In the below diagram, the box holds a value of 100 and is named as x. Therefore, the variable is x, and the data it holds is the value.



Data Types


Python - Data Types
Numeric
Interger
Complex Number
Float
Dictionary
Boolean
Set
Sequence Type
Strings
List
Tuple

Dynamic Types

Python is a **dynamically typed** language. It doesn't know about the type of the variable until the code is run.

```
x = 6

print(type(x))

x = 'hello'

print(type(x))
```

Naming Conventions

Variable Names

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume).

Rules for Python variables:

- **Rule-1**: You should start variable name with an alphabet or underscore(_) character.
- **Rule-2**: A variable name can only contain **A-Z,a-z,0-9** and **underscore(_).**
- **Rule-3**: You cannot start the variable name with a number.
- **Rule-4**: You cannot use special characters with the variable name such as such as **$,%,#,&,@.-,^** etc.
- **Rule-5**: Variable names are case sensitive. For example **str and Str** are two different variables.
- **Rule-6**: Do not use reserve keyword as a variable name for example keywords like **class, for, def, del, is, else, try, from,** etc**.**

Python Reserved Keywords

Python Keywords			
False	def	if	raise
None	del	import	return
True	elif	in	try
and	else	is	while
as	except	lambda	with
assert	finally	nonlocal	yield
break	for	not	
class	from	or	
continue	global	pass	

Instruction/Statement

A **statement is an instruction that a Python interpreter can execute**. So, in simple words, we can say anything written in Python is a statement.

```
# Statement 1
print('Hello')

# Statement 2
x = 20

# Statement 3
print(x)
```

Basic Syntax Comments

```
print('Bismillah')

# print('Bismillah')

'''
print('Bismillah')
print('Bismillah')
'''


"""
print('Bismillah')
print('Bismillah')
"""
print('Alhamdulillah')
```

Receiving Input

Get user input with Python using the input() function. The user can enter keyboard input in the console

```
print("Enter your first name: ")

name = input()

print("Nice to meet you", name)
```

```
name = input("Enter your first name: ")

print("Nice to meet you", name)
```

Type Conversion/Casting

Type Conversion

The process of converting the value of one data type (integer, string, float, etc.) to another data type is called type conversion. Python has two types of type conversion.

- **Implicit Type Conversion**

  In Implicit type conversion, Python automatically converts one data type to another data type. This process doesn't need any user involvement.

```
num_int = 123
num_flo = 1.23

num_new = num_int + num_flo

print("datatype of num_int:",type(num_int))
print("datatype of num_flo:",type(num_flo))

print("Value of num_new:",num_new)
print("datatype of num_new:",type(num_new))
```

Type Conversion/Casting

❑ **Explicit Type Conversion**

In Explicit Type Conversion, users convert the data type of an object to required data type. We use the predefined functions like int(), float(), str(), etc to perform explicit type conversion.

```
num_int = 123
num_str = "456"

print("Data type of num_int:",type(num_int))
print("Data type of num_str:",type(num_str))

print(num_int+num_str)
```

```
num_int = 123
num_str = "456"

print("Data type of num_int:",type(num_int))
print("Data type of num_str before Type
Casting:",type(num_str))

num_str = int(num_str)
print("Data type of num_str after Type Casting:",type(num_str))

num_sum = num_int + num_str

print("Sum of num_int and num_str:",num_sum)
print("Data type of the sum:",type(num_sum))
```

Numeric Data Types

Python Number Types: **int, float**

Int

In Python, integers are zero, positive or negative whole numbers without a fractional part

```
num=100
print(num)
```

Float

In Python, floating point numbers (float) are positive and negative real numbers with a fractional part denoted by the decimal symbol.

```
num=100.25
print(num)
```

Boolean Data Types

boolean

The boolean value can be of two types only i.e. either True or False. The output <class ‘bool’> indicates the variable is a boolean data type.

```
a = True
print(a)
print(type(a))

b = False
print(b)
print(type(b))
```

Swapping

```
a = 10
b = 20

temp = a
a = b
b = temp

print(a, b)
```

```
a = 10
b = 20

a = a+b
b = a - b
a = a -b

print(a, b)
```

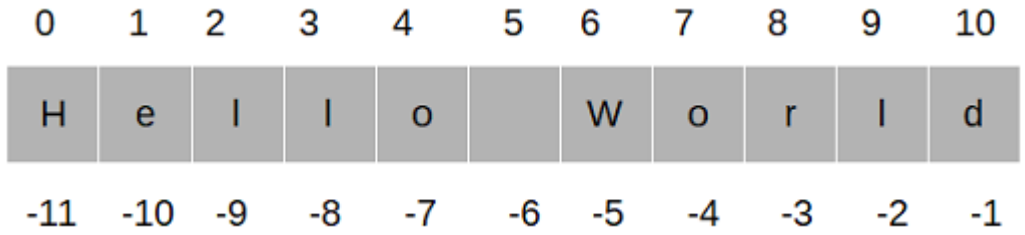
```
a = 10
b = 20

a, b = b, a

print(a, b)
```

Strings

Strings are List like many other popular programming languages. Python does not have a character data type, a single character is simply a string with a length of 1. Square brackets can be used to access elements of the string.




```
a = "Hello World"
print(a)
print(a[0])
print(a[-1])
print(a[0:3])
print(a[0:])
print(a[1:])
print(a[:4])
print(a[0:-1])
```