python

Chapter 1

Introduction

Outline

- History
- Features
- Environment Setup
- Working with Python
- Variables and Data Types
- Operators

Python : History

1980

- Python was founded by Guido van Rossum.
- As a successor of ABC programming language

2000

- Python 2.0 was released.
- With many major new features such as garbage collector.

2008

- Python 3.0 was released.
- At that time, it was not back-compatible.
- However, many of its major features have been backported to Python 2.x

2017

- The EOL of Python was announced as 2015, but postponed to 2020 due to large existing code base.
- Google in January announced the *transcompiler* for Python 2.x.

Python : Features

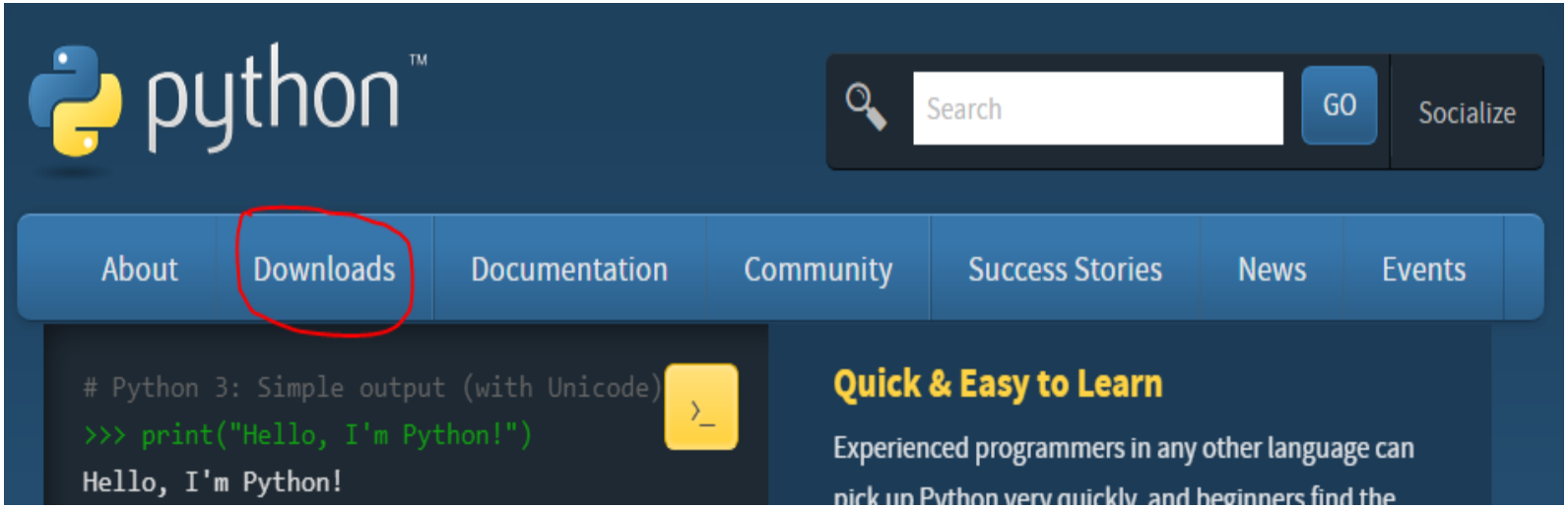
- **Easy-to-learn**
- **Easy-to-read**
- **Easy-to-maintain**
- **A broad standard library**
- **Interactive Mode**
- **Portable**
- **Supports all major databases**
- **Supports automatic garbage collection**

Environment Setup

- What tools we need :
  - Python Interpreter
    - Python 2.x
    - Python 3.x
  - Code Editor
    - PyCharm IDE
    - PyDev IDE
    - Vim
    - Notepad ++
    - Gedit

How to Installing a Python Interpreter

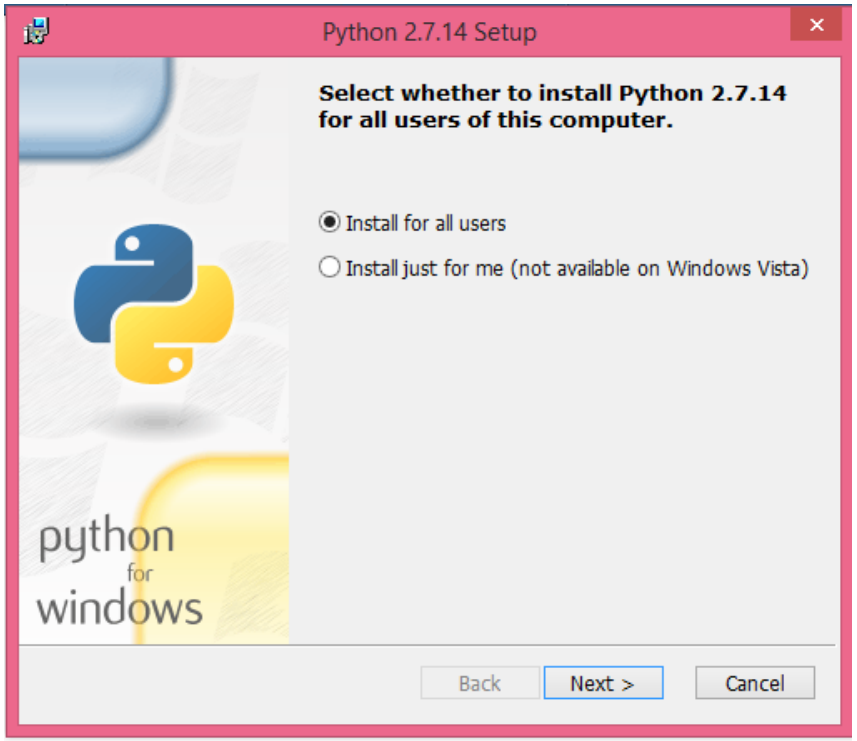
- Goto python.org website and click on downloads



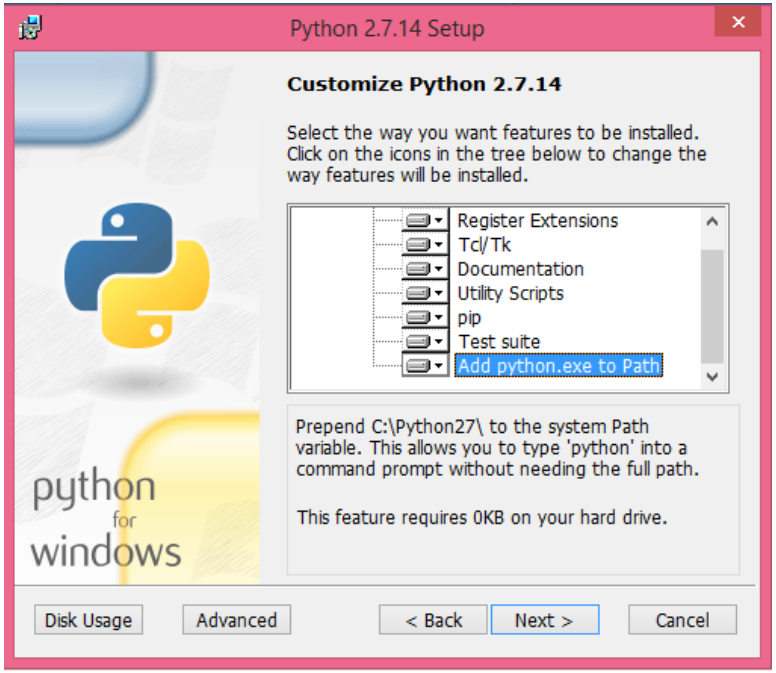
- Then from the drop down menu, select your platform i,e.
  - Windows
  - Linux
  - Mac OS X

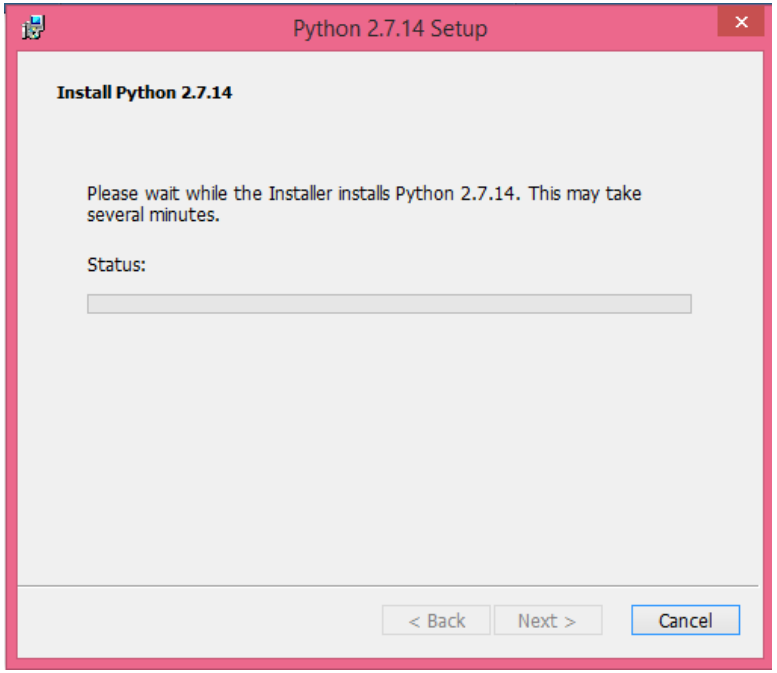
How to Installing a Python Interpreter

- Then you select your preferred python flavor (2.x or 3.x)
- A setup file will be downloaded.
- After download is complete, double click on the setup file.



How to Installing a Python Interpreter





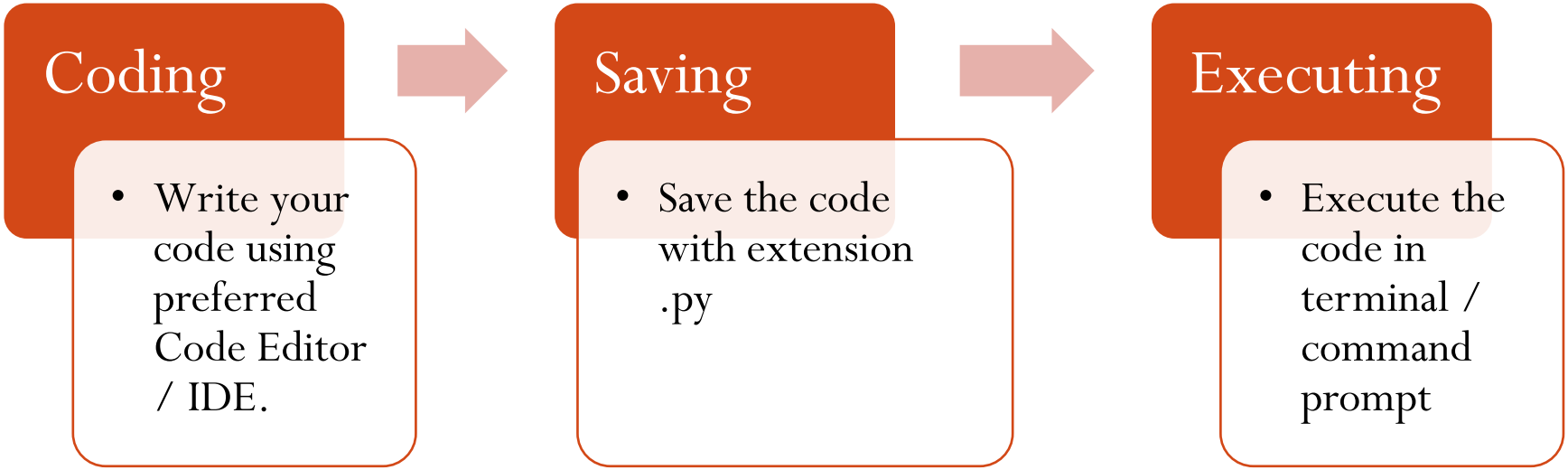
- Once Installation is complete, you can write your first program.

How to Installing a Code Editor

- Core Python
  - In Windows, Notepad++ : notepad-plus-plus.org
  - Ubuntu: Gedit / vim (Installed by default)

- Advanced Python
  - PyCharm IDE
  - PyDev IDE

Working with Python



Writing your first program.

- Demo of a simple Hello World program.
- Demo two different ways to execute a python script.

Variables and Data Types

- **Python Identifiers :** An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).

Identifiers	Validity
abc	Valid
_abc	Valid
Abc	Valid
8abc	Invalid
%abc	Invalid
A	Valid
abc678#1	Invalid

Variables and Data Types

- **Reserved Words :**

and	exec	not
assert	finally	or
break	for	pass
class	from	print
continue	global	raise
def	if	return
del	import	try
elif	in	while
else	is	with
except	lambda	yield

Check your understanding

- State which of then are Valid Python identifiers.
  a) RollNo
  b) _name
  c) ^class
  d) class
  e) Class
  f) date-of-birth
  g) car's_plate_no
  h) var56
  i) 63mark
  j) lambda

Variables and Data Types

- Standard data types in Python:
  - Number
    - int
    - float
    - long
    - complex
  - String
  - Boolean
  - List
  - Tuple
  - Dictionary

Note : List, Tuple and Dictionary data types will be discussed separately.

Variables and Data Types

- Assigning a value to an identifier / variable.
  - Single Assignment

```
City = "Bangalore"
age = 31
Value = 2+3j
```

  - Multiple Assignment

```
a = b = c = 10
```

Check your understanding

- Indentify the data types of the following constants
  - 1.25
  - 1
  - abc
  - 2+3j
  - 99983838338383
  - True
  - False
  - “Bangalore”

Operators in Python

- Based on type of operation
  - Arithmetic Operators
  - Comparison (Relational) Operators
  - Logical Operators
  - Bitwise operator
  - Assignment operators
  - Special Operators
- Based on number of operands
  - Unary
  - Binary
  - Ternary

Arithmetic Operators

Operator	Meaning	Example
+	Add two operands or unary plus	`x + y` `+2`
-	Subtract right operand from the left or unary minus	`x - y` `-2`
*	Multiply two operands	`x * y`
/	Divide left operand by the right one (always results into float)	`x / y`
%	Modulus - remainder of the division of left operand by the right	`x % y (remainder of x/y)`
//	Floor division - division that results into whole number adjusted to the left in the number line	`x // y`
**	Exponent - left operand raised to the power of right	`x**y (x to the power y)`

Comparison Operators

Operator	Meaning	Example
>	Greater that - True if left operand is greater than the right	`x > y`
<	Less that - True if left operand is less than the right	`x < y`
==	Equal to - True if both operands are equal	`x == y`
!=	Not equal to - True if operands are not equal	`x != y`
>=	Greater than or equal to - True if left operand is greater than or equal to the right	`x >= y`
<=	Less than or equal to - True if left operand is less than or equal to the right	`x <= y`

Logical Operators

Operator	Meaning	Example
`and`	True if both the operands are true	`x and y`
`or`	True if either of the operands is true	`x or y`
`not`	True if operand is false (complements the operand)	`not x`

Bitwise Operators

Operator	Meaning	Example
&	Bitwise AND	x & y = 0
|	Bitwise OR	x | y
~	Bitwise NOT	~x
^	Bitwise XOR	x ^ y
>>	Bitwise right shift	x>>2
<<	Bitwise left shift	x<<2

Assignment operators

Operator	Meaning	Example
=	Equal to	`x = 5`
`{ArithmeticOperator}=`	Calculate then assign	`X += 5` `x = x + 5`
`{BitwiseOperator}=`	Calculate then assign	`x &= 2` `x = x & 2`

Note : Assignemnt operator (~=) is not suppoerted.

Special Operator

- Identity Operator

Operator	Meaning	Example
`is`	True if the operands are identical	`x is y`
`is not`	True if the operands are not identical	`x is not y`

Note : Will be explained more in Object Oriented programming

- Membership Operator

Operator	Meaning	Example
`in`	True if value/variable is found in the sequence	`x in y`
`not in`	True if value/variable is not found in the sequence	`x not in y`

Note : Will be explained more in List, Tuple, Map

Some more rules

- Lines and Indentation
  - Python provides no braces to indicate blocks of code for class and function definitions or flow control. Blocks of code are denoted by line indentation, which is rigidly enforced.
- Multi-Line Statements
  - Statements in Python typically end with a new line. Python does, however, allow the use of the line continuation character (\) to denote that the line should continue.
- Quotation in Python
  - Python accepts single ('), double (") and triple ('" or """) quotes to denote string literals, as long as the same type of quote starts and ends the string.
- Comments in Python
  - A hash sign (#) that is not inside a string literal begins a comment. All characters after the # and up to the end of the physical line are part of the comment and the Python interpreter ignores them.

Check your understanding.

- Write a program to find simple interest.
- Write a program to find area of triangle.
- Write a program to join two strings.