



Scripting

Agenda

1. Python Introduction
2. Installation and usage of Python
3. Python2 vs Python3
4. First python Script
5. Basics of Python
6. If condition and Loops
7. Important libraries in Python Scripting

Python Introduction

1. General-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990
2. Python is Interpreted , Easy-to-learn , Easy-to-read , Easy-to-maintain , A broad standard library , Portable and Scalable
3. Supports functional and structured programming methods as well as OOP and can be used as a scripting language
4. Supports automatic garbage collection and can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Installation and Usage of Python

1. Download python

<https://www.python.org/downloads/release>

2. Extract and run the package

Usage : Python is supported on many IDEs such as Eclipse, Visual Studio Code , IDLE , Sublime Text 3., Atom.Thonny , PyCharm (Paid). We can use notepad ++, Vim ,nano etc., editors to use python programming

Comparison Python2

Python3

Function print	<code>print ("hello")</code>	<code>print "hello"</code>
Division of Integers	Whenever two integers are divided, you get a float value	When two integers are divided, you always provide integer value.
Syntax	The syntax is simpler and easily understandable.	The syntax of Python 2 was comparatively difficult to understand.
Exceptions	It should be enclosed in parenthesis.	It should be enclosed in notations.
Library	Many recent developers are creating libraries which you can only use with Python 3.	Many older libraries created for Python 2 is not forward-compatible.
Leak of variables	The value of variables never changes.	The value of the global variable will change while using it inside for-loop.

First Python Script

1. Write first script which is main.py

```
1  #!/usr/bin/python
2  print "Hello, DevOps Engineer !"
```

2. Execute the script after the python is installed as below

```
$python main.py
Hello, DevOps Engineer !
```

Basics of Python

Indentation and Quotation

1. Check the Indentation as below

```
1  #!/usr/bin/python
2  print "Hello, DevOps Engineer !"
3  a=1
4  if a==1 :
5      print "Welcome to Python Scripting"
6  else:
7      print "Sorry, you should learn Python to be highly paid DevOps Engineer"
```

2. Types of Quotations

```
1  #!/usr/bin/python
2  print "Hello, DevOps Engineer !"
3  word = 'word'
4  sentence = "This is a sentence."
5  paragraph = """This is a paragraph. It is
6  made up of multiple lines and sentences."""
```


Comments and input

Single line and Multi line Comments

```
1  #!/usr/bin/python
2  # single line comment
3  print "Hello, DevOps Engineer !"
4  '''
5  This is a multiline
6  comment.
7  '''
```

Input a value

```
1  #!/usr/bin/python
2  x = raw_input("Enter a value : ")
3  print("the value entered is : "+x)
```

Command Line Arguments and Multi statements in single line

Command Line Arguments

```
1  #!/usr/bin/python
2  import sys
3
4  print("Number of arguments: " + len(sys.argv) + " arguments.")
5  print("Argument List:" + str(sys.argv))
6  print("First argument is the name of document which is " + sys.argv[0])
7  print("Second argument is " + sys.argv[1])
8  ..
9  print("nth argument is " + sys.argv[n])
```

Execution of arguments list

```
1  python test.py arg1 arg2 arg3 ...|
```

Multiple statements in single line

```
1  import sys; x = 'foo'; sys.stdout.write(x + '\n')|
```

Data types and Numerical types

Data types :

1. Numbers
2. String
3. List
4. Tuple
5. Dictionary

Numerical types :

1. int (signed integers)
2. long (long integers, they can also be represented in octal and hexadecimal)
3. float (floating point real values)
4. complex (complex numbers)

What is List

List : stores a sequence of objects in a defined order

```
1  #!/usr/bin/python
2
3  first_list = [ 'I am moving to ', 2019 , 2020, 'learn DevOps', 1234 ]
4  second_list = [2020, 'Revolution of DevOps']
5
6  print first_list           # Prints complete list
7  print first_list[0]       # Prints first element of the list
8  print first_list[0:3]     # Prints elements starting from 2nd till 3rd
9  print first_list[2:]     # Prints elements starting from 3rd element
10 print second_list * 2    # Prints list two times
11 print first_list + second_list # Prints concatenated lists
```

```
$python main.py
['I am moving to ', 2019, 2020, 'learn DevOps', 1234]
I am moving to
['I am moving to ', 2019, 2020]
[2020, 'learn DevOps', 1234]
[2020, 'Revolution of DevOps', 2020, 'Revolution of DevOps']
['I am moving to ', 2019, 2020, 'learn DevOps', 1234, 2020, 'Revolution of DevOps']
```

What is Tuple

Tuple: A tuple is another sequence data type that is similar to the list but list enclosed in brackets (`[]`) and their elements and size can be changed, while tuples are enclosed in parentheses (`()`) and cannot be updated.

```
1  #!/usr/bin/python
2
3  tuple = ( 'I am moving to', 2019 , 2020, 'learn DevOps', 1234 )
4  list = [ 'I am moving to', 2019 , 2020, 'learn DevOps', 1234 ]
5  tuple[2] = 2021      # Invalid syntax with tuple
6  list[2] = 2021      # Valid syntax with list
```

```
$python main.py
```

```
Traceback (most recent call last):
```

```
  File "main.py", line 5, in <module>
```

```
    tuple[2] = 2021      # Invalid syntax with tuple
```

```
TypeError: 'tuple' object does not support item assignment
```

Dictionary

Dictionary: Consist of key-value pairs and are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([])

```
1  #!/usr/bin/python
2
3  dict = {}
4  dict['institute'] = "Qshore"
5  dict[2020] = "This is 2020 year"
6
7  tinydict = {'name': 'DevOps', 'batch': 'January', 'Year': 2020 }
8
9
10 print dict['institute']      # Prints value for 'institute' key
11 print dict[2020]             # Prints value for 2020 key
12 print tinydict               # Prints complete dictionary
13 print tinydict.keys()        # Prints all the keys
14 print tinydict.values()      # Prints all the values
```

```
$python main.py
Qshore
This is 2020 year
{'Year': 2020, 'name': 'DevOps', 'batch': 'January'}
['Year', 'name', 'batch']
[2020, 'DevOps', 'January']
```

IF condition and Loops

IF Condition

```
1  #!/usr/bin/python
2
3  var = 100
4  if var < 200:
5      print "Expression value is less than 200"
6      if var == 150:
7          print "Which is 150"
8      elif var == 100:
9          print "Which is 100"
10     elif var == 50:
11         print "Which is 50"
12     elif var < 50:
13         print "Expression value is less than 50"
14 else:
15     print "Could not find true expression"
16
17 print "Good bye!"
```


While Loop

```
1  #!/usr/bin/python
2
3  count = 0
4  while (count < 9):
5      print 'The count is:', count
6      count = count + 1
7
8  print "Good bye!"
```

For Loop


```
1  #!/usr/bin/python
2
3  for letter in 'Python':      # First Example
4      print 'Current Letter :', letter
5
6  fruits = ['banana', 'apple', 'mango']
7  for fruit in fruits:          # Second Example
8      print 'Current fruit :', fruit
9
10 print "Good bye!"
```

Important Libraries in Python scripting

Some Important Modules in Python Scripting

1. Sys module
2. Os module
3. Subprocess
4. Math
5. Random
6. Date and time
7. json

Example for Sys module

```
import sys  
  
print(sys.version)  
  
print(sys.argv)
```

```
import sys  
print("Hello {}. Welcome to {} tutorial".format(sys.argv[1], sys.argv[2]))
```

```
>>> import sys  
>>> sys.maxsize  
9223372036854775807
```

```
>>> sys.path
```

Example for Os module

```
>>>os.chdir("tempdir")
>>>os.getcwd()
'd:\\tempdir'
>>>os.rmdir("d:\\tempdir")
PermissionError: [WinError 32] The process cannot access the file because it
is being used by another process: 'd:\\tempdir'
>>>os.chdir("..")
>>>os.rmdir("tempdir")
```

```
>>>os.listdir("c:\\python37")
['DLLs', 'Doc', 'fantasy-1.py', 'fantasy.db', 'fantasy.py', 'frame.py',
'gridexample.py', 'include', 'Lib', 'libs', 'LICENSE.txt', 'listbox.py',
'NEWS.txt', 'place.py', 'players.db', 'python.exe', 'python3.dll',
'python36.dll', 'pythonw.exe', 'sclst.py', 'Scripts', 'tcl', 'test.py',
'Tools', 'tooltip.py', 'vcruntime140.dll', 'virat.jpg', 'virat.py']
```

Thank you
