Learning Python

@kabirbaidhya

Introduction

What is Python?

> Python is a programming language that lets you work quickly and integrate systems more effectively.
>
> www.python.org

Why Python?

1. Easy to learn yet very powerful
2. General purpose language that could be used almost everywhere
3. Flexibility and scalability
4. Multi-paradigm: object oriented, functional, procedural etc.
5. Popular language, big community
6. Fun to work with
7. Career Opportunities

Python is everywhere

- CLI applications
- Web Apps backened / APIs
- GUI applications
- Network applications
- System Administration / DevOps
- Scientific Applications
- Data Science Projects
- Machine Learning / Deep learning
- and much more

Getting Started

Python 2 or 3?

There are two versions of python currently in use python 2.7 & python 3.

So, begineers often come up with question **“Should I choose Python 2 or Python 3”**?

Current State

1. Most production applications today use Python 2.7.
2. Python 3 is ready for the production deployment of applications today.
3. Python 2.7 will only receive necessary security updates until 2020.
4. The brand name “Python” encapsulates both Python 3 and Python 2.

So... 3?

Yes, start with python 3.x but familiarizing yourself with Python 2.7 will be very useful.

In case if you have to work with legacy projects in the future you might need to know 2.x too.

Installation

Linux

Python comes out of the box in most modern linux distros like Debian, Fedora, Centos etc. In case it's not preinstalled you can always download it from the official website.

Windows

You can download the installer from the official website.
https://www.python.org/downloads/

Code Editors

You could use any code editors out there to write code in python as long as it supports syntax highlighting.

Here are some recommended editors/IDEs:

- Microsoft Vscode
- Atom
- Sublime
- Vim
- Jetbrains PyCharm IDE

Python Basics

Hello World

Hello World in python is just a one liner.

```
print "Hello, World!"
```

Pretty neat. Isn't it?

Hello World: Python 3

In `python3` it's

```
print("Hello, World!")
```

Mind the parenthesis. In python3 `print` is a function.

How'd you run it?

1. Directly in the Python shell.

Run python shell first.

```
 ➜ python
Python 2.7.12 (default, Nov 19 2016, 06:48:10)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more i
>>>
```

Then

```
>>> print "Hello World!"
Hello World!
```

2. Standard approach

Create a file `hello_world.py` with the code.

```
print "Hello World!"
```

Now run it with python.

```
➜ python hello_world.py
```

```
Hello World!
```

How it works?

Compilation & Interpretation

Python is a dynamic language, and running it from the command line essentially triggers the following steps:

1. The source is compiled the first time it is encountered (e.g., imported as a module or directly executed). This step generates the binary file, with a pyc or pyo extension depending on your system.

2. The interpreter reads the binary file and executes the instructions (opcodes) one at a time.

From http://security.coverity.com/blog/2014/Nov/understanding-python-bytecode.html

Syntax and semantics

Incase you're from a C-style language

You need to be aware of few things in python.

1. No semicolon
2. No curly braces
3. Indentation is a part of syntax
4. Dynamically-typed language

Pythonic code

Looks something like this

```
"""
Generates Fibonacci series upto n
"""
def fib(n):
    a, b = 0, 1

    while a < n:
        print a
        a, b = b, a+b


fib(1000)
```

Let's dive into the code

Example 1: Printing and Comments

```
# This prints Hello World!
print "Hello World!"

# This is a comment; single line comment

"""
And this is a multiline comment.
As this could take more than one line of comments.
Like this.
"""
print "Hello Again."
```

Output:

```
Hello World!
Hello Again.
```

Example 2: Printing More

```
# Printing more texts
print "Hello World!"
print "Hello Again."
print "We enjoy typing."
print "Learning python is easy."
print "And fun."
print "Printing text is way easier."
```

Output:

```
Hello World!
Hello Again.
We enjoy typing.
Learning python is easy.
And fun.
Printing text is way easier.
```

Example 3: Printing more than just texts

```
print "One + One =", 1 + 1
print "One - Two =", 1 - 2
print "Two * Five =", 1 * 5
print "Four / Two =", 4 / 2
print "Expression (12 * 5) - (2 ^ 3) + (1 / 2) =", ((12 * 5
print "Seven is less than or equal to Six is", 7 <= 6
```

Output:

```
One + One = 2
One - Two = -1
Two * Five = 5
Four / Two = 2
Expression (12 * 5) - (2 ^ 3) + (1 / 2) = 52
Seven is less than or equal to Six is False
```

Example 4: Variables and Printing

```
first_name = "Kabir"
last_name = "Baidhya"
dob = "July 30, 1992"
home_town = "Kathmandu, Nepal"

print "Hi! I am", first_name, last_name, "."
print "I was born on", dob, "."
print "I'm from", home_town, "."
```

Output:

```
Hi! I am Kabir Baidhya .
I was born on July 30, 1992 .
I'm from Kathmandu, Nepal .
```

Example 5: Proper Formatting

```
first_name = "Kabir"
last_name = "Baidhya"
dob_month = "July"
dob_day = 30
dob_year = 1992
difficulty = "easy"

print "Hi! I am %s %s." % (first_name, last_name)
print "I was born on %s %d, %d." % (dob_month, dob_day, dob
print "Python is %s to learn." % difficulty
```

Output:

```
Hi! I am Kabir Baidhya.
I was born on July 30, 1992.
Python is easy to learn.
```

Example 6: User Input

```
print "What is your name?",

# This would take input from the user
# and store it in a variable.
name = raw_input()

print "Hi! %s. \nIt's nice to meet you." % (name)
print "Hope you're doing good."
```

Output:

```
What is your name? Kabir
Hi! Kabir.
It's nice to meet you.
Hope you're doing good.
```

Data Types

Buit-in Data types

- **Numeric**: int, float, long
- **Boolean**: bool
- **Sequences**: str, list, tuple, bytes
- **Mappings**: dict
- **Sets**: set, frozen set

Read more
https://en.wikibooks.org/wiki/Python_Programming/Data_Types

Immutable & Mutable types

1. Immutable types
    - int, float, long, tuple, byetes, frozen set, etc.
2. Mutable types
    - list, dict, set, etc.

Example 7: Basic data types

```
an_integer = 6
a_floating_point = 17.60
a_boolean = True
a_string = "Foo"

print "Integer value = %d" % an_integer
print "Float value = %f" % a_floating_point
print "Boolean value = %r" % a_boolean
print "String value = %s" % a_string
```

Output:

```
Integer value = 6
Float value = 17.600000
Boolean value = True
String value = Foo
```

Example 8: Lists

```
fruits = ['Banana', 'Apple', 'Lime']
numbers = [1, 2, 3, 4, 5, 6, 7, 8]

# This is called list comprehension
even_numbers = [x for x in numbers if x % 2 == 0]

print "Numbers: %s" % numbers
print "Even Numbers: %s" % even_numbers
print "Fruits: %s, %s and %s" % (fruits[0], fruits[1], fru
```

Output:

```
Numbers: [1, 2, 3, 4, 5, 6, 7, 8]
Even Numbers: [2, 4, 6, 8]
Fruits: Banana, Apple and Lime
```

Example 9: Dictionaries

```
data = {
    "name": "Kabir Baidhya",
    "dob": "July 30, 1992",
    "home_town": "Kathmandu, Nepal"
}

print "Hi! I am %s." % data["name"]
print "I was born on %s." % data["dob"]
print "I'm from %s." % data["home_town"]
```

Output:

```
Hi! I am Kabir Baidhya.
I was born on July 30, 1992.
I'm from Kathmandu, Nepal.
```

Read More?

1. http://docs.python-guide.org/en/latest
2. https://www.python.org
3. https://iluxonchik.github.io/why-you-should-learn-python/
4. https://learnpythonthehardway.org/book/

Any Questions?

Thank You

@kabirbaidhya

kabirbaidhya@gmail.com