Python

1. High Level Programming Language.
2. Created By Guido Van Rossum

1. 1989

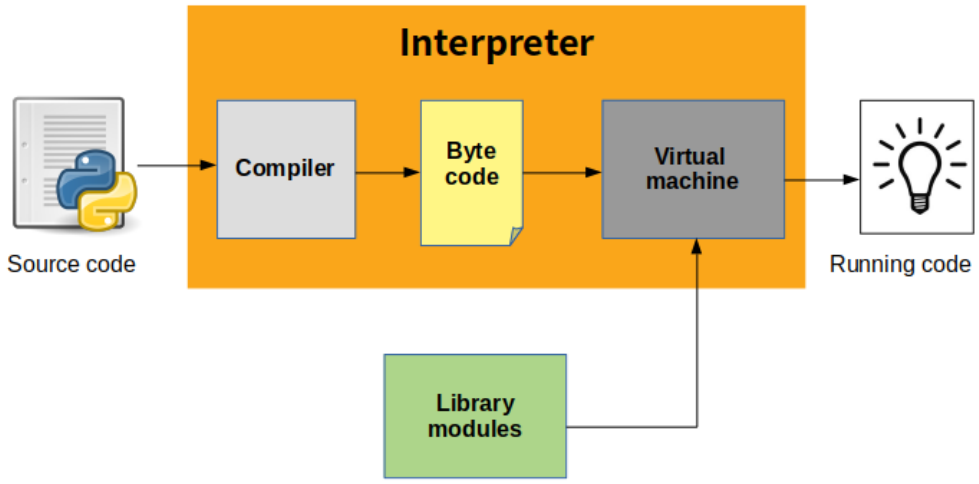
Python supports multiple programming paradigms

1. Object Oriented Programming
2. Imperative Programming
3. Interpreted Programming
4. Functional Programming
5. Procedural Programming

Features

1. Easy to learn
2. Has a broad standard library
3. Portable
4. Extendable
5. Databases
6. GUI programming and Desktop Application
7. Supports Dynamic Data Types
8. Automatic Garbage Collection
9. Easy integration with C, C++, JAVA and more.

Source Code to Byte Code



Compilation

It is simply a translation state.

Byte Code

It is low level plateform independent representation of source code.

Who uses python today ?

Industrial Light and Magic

ILM chose Python 1.4 over Perl and Tcl, opting to use Python because it was a much faster to integrate into their existing infrastructure. Because of Python's easy interoperability with C and C++, it was simple for ILM to import Python into their proprietary lighting software.

This let them put Python in more places, using it for wrapping software components and extend their standard graphics applications.

The studio has used Python in multiple other facets of their work. Developers use Python to track and audit pipeline functionality, maintaining a database of every image produced for each film. As more and more of ILM's programs were controlled by Python, it created a simpler unified toolset that allowed for a more effective production pipeline.

For a real world example, look no farther than OpenEXR, an HD file format used by ILM. As part of the package, PyIlmBase is included (although it does have a Boost dependency).

Google

Google has been a supporter of Python from nearly the very beginning. In the beginning, the founders of Google made the decision of “Python where we can, C++ where we must.” This meant that C++ was used where memory control was imperative and low latency was desired. In the other facets, Python enabled for ease of maintenance and relatively fast delivery.

Even when other scripts were written for Google in Perl or Bash, these were often recoded into Python. The reason was because of the ease of deployment and how simple Python is to maintain. In fact, according to Steven Levy – author of “In the Plex,” Google’s very first web-crawling spider was first written in Java 1.0 and was so difficult that they rewrote it into Python.

Facebook

According to a 2016 post by Facebook, Python is currently responsible for multiple services in infrastructure management.

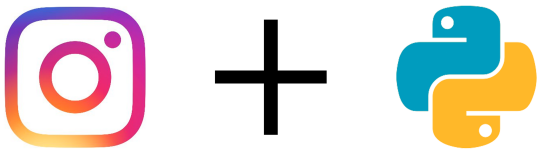
These include using TORconfig to handle network switch setup and imaging, FBOSS for whitebox switch CLIs, and using Dapper for scheduling and execution of maintenance work.

Facebook has published numerous open-source Python projects written for Py3 including a Facebook Ads API and a Python Async IRCbot framework.

Instagram

In 2016, the Instagram engineering team boasted that they were running the world's largest deployment of the Django web framework, which is written entirely in Python.

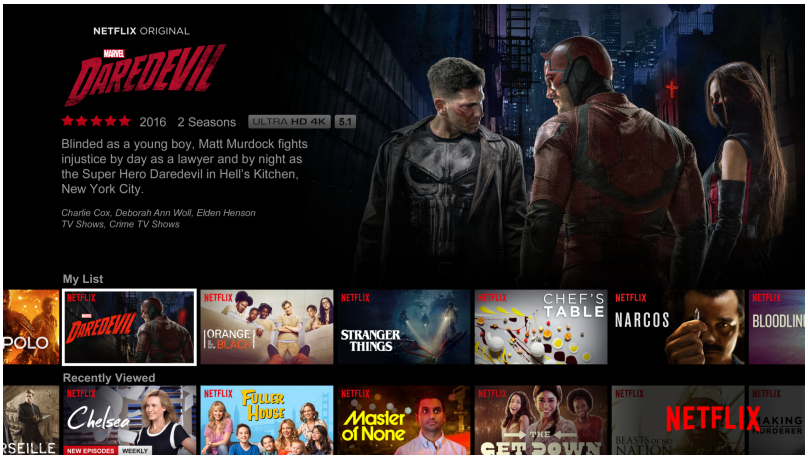
In 2017, Instagram migrated most of their Python code base from Python 2.7 to Python 3.



Netflix

One of the primary places that Python is used is in the Central Alert Gateway. This RESTful web app processes alerts from anywhere, and then route them to people or groups that would need to see them.

Additionally, the app has the power to suppress duplicate alerts that have already been handled and in some cases, perform automated solutions such as rebooting a process or terminating something that is starting to look shaky. This app is a huge win for Netflix considering the sheer volume of alerts. Handling them intelligently means that developers and engineers aren't flooded with redundant calls.



Another area that Python is used at Netflix is with monkey apps used to track security changes and history. These monkeys are used to track and alert any changes in EC2 security related policies in any groups, tracking any changes in these environments. They are also used to ensure to track the dozens of SSL certificates attached to Netflix's multiple domains. In tracking, Netflix has seen a reduction in unexpected expirations drop from one very quarter to none since 2012.

Goverment Agencies

In [1]:

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

```
Hello
```

In [4]:

```
print("Hello Indore")
```

```
Hello Indore
```

In [5]:

```
print("Hello")
print("Indore")
```

```
Hello
Indore
```

In [6]:

```
print("Hello\nIndore")  #\n - > new line character
```

```
Hello
Indore
```

In [7]:

```
print("Hello\tIndore")  #\t -> horizontal tab
```

```
Hello   Indore
```

In [ ]:

```
*
**
***
```

In [8]:

```
print("*\n**\n***")
```

```
*
**
***
```

Name College City

In [9]:

```
print("Himanshu\n\tData Science\n\t\tIndore")
```

```
Himanshu
        Data Science
                Indore
```

```
   *

  •
```

In [10]:

```
print("    *\n*\t*\n    *")
```

```
    *
*       *
    *
```

Variables

In [13]:

```
a = 10
print(a)
```

```
10
```

In [14]:

```
type(a)
```

Out[14]:

```
int
```

In [15]:

```
a = 10
b = 20
print(a)
print(b)
print( a+b )
```

```
10
20
30
```

In [16]:

```
x = 7
print(x)
print(x+3)
print(x)
```

```
7
10
7
```

In [17]:

```
x = x+3
print(x)
```

```
10
```

In [18]:

```
a = "Indore"
type(a)
```

Out[18]:

```
str
```

In [22]:

```
a = "Indore"
print(a)
t = 20
print(t)
del a
print(a)
```

```
Indore
20

---------------------------------------------------------------------
-------
NameError                                 Traceback (most recent cal
l last)
<ipython-input-22-6216a4cade47> in <module>
      4 print(t)
      5 del a
----> 6 print(a)

NameError: name 'a' is not defined
```

In [23]:

```
a  = 10.5
type(a)
```

Out[23]:

```
float
```

In [24]:

```
a = 10
b = "Hi"
c = 7.5
print(a)
print(b)
print(c)
```

```
10
Hi
7.5
```

User Input

In [25]:

```
x = input()
print(x)
```

```
India
India
```

In [26]:

```
x = input()
print("You have entered : ", x)
```

```
Google
You have entered :  Google
```

In [27]:

```
y = input("Enter your name : ")
print("Hello!! ", y)
```

```
Enter your name : Puneet Sir
Hello!!  Puneet Sir
```

Type conversion

In [29]:

```
a = "10"
print(type(a))
b = int(a)
print(type(b))
```

```
<class 'str'>
<class 'int'>
```

In [30]:

```
a = input("Enter 1st number : ")
b = input("Enter 2nd Number : ")
c = a+b
print(c)
```

```
Enter 1st number : 10
Enter 2nd Number : 20
1020
```

In [32]:

```
a = int(input("Enter 1st number : "))
b = int(input("Enter 2nd Number : "))
c = a+b
print(c)
```

```
Enter 1st number : 10
Enter 2nd Number : 20
30
```

In [33]:

```
a = int(input("Enter 1st number : "))
b = int(input("Enter 2nd Number : "))
c = a+b
print("Addition is = ",c)
```

```
Enter 1st number : 77
Enter 2nd Number : 33
Addition is =  110
```

Operators

1. Arithmetic Operators

+ , - , * , / , % , // , **

In [34]:

```
20 + 30
```

Out[34]:

```
50
```

In [35]:

```
40 - 50
```

Out[35]:

```
-10
```

In [36]:

```
3 * 4
```

Out[36]:

```
12
```

In [37]:

```
7 / 3
```

Out[37]:

```
2.3333333333333335
```

In [38]:

```
7 % 2
```

Out[38]:

```
1
```

In [39]:

```
10 % 4
```

Out[39]:

2

In [40]:

```
21 % 9
```

Out[40]:

3

In [41]:

```
7 // 2
```

Out[41]:

3

In [42]:

```
2 ** 3
```

Out[42]:

8

In [43]:

```
2 ** 3 ** 2
```

Out[43]:

512

1. Assignment Operators

= , += , -= , *= , /= , %= , //=

In [44]:

```
a = 10
a += 5   # a = a + 5
print(a)
```

15

In [45]:

```
a = 10
a *= 5   # a = a * 5
print(a)
```

50

In [46]:

```
a = 10
a %= 3   # a = a % 3
print(a)
```

1

1. Relational Operators

> , < , >= , <= , == , !=

In [47]:

```
10 > 3
```

Out[47]:

True

In [48]:

```
10 < 3
```

Out[48]:

False

In [49]:

```
5 >= 5
```

Out[49]:

True

In [50]:

```
10 == 10
```

Out[50]:

True

In [51]:

```
10 == 4
```

Out[51]:

False

In [52]:

```
7 != 7
```

Out[52]:

False

In [53]:

```
7 != 4
```

Out[53]:

True

1. Boolean Operators

    not
    and
    or

1. Bitwise Operators

    &
    |
    ^
    <<
    >>

0 & 0 - 0 0 & 1 - 0 1 & 0 - 0 1 & 1 - 1

In [2]:

```
1 & 3   # 0001
        # 0011  &
        # 0001
```

Out[2]:

1

In [3]:

```
7 & 4   # 0111
        # 0100  &
        # 0100
```

Out[3]:

4

In [4]:

```
11 & 6  # 1011
        # 0110   &
        # 0010
```

Out[4]:

2

0 | 0 - 0 0 | 1 - 1 1 | 0 - 1 1 | 1 - 1

In [5]:

```
5 | 4   # 0101
        # 0100  |
        # 0101
```

Out[5]:

5

0 ^ 0 - 0 0 ^ 1 - 1 1 ^ 0 - 1 1 ^ 1 - 0

In [6]:

```
7 ^ 3   # 0111
        # 0011   ^
        # 0100
```

Out[6]:

4

In [7]:

```
8 << 2    #  00100000
```

Out[7]:

32

In [8]:

```
11 << 3  # 00001011
         # 01011000
```

Out[8]:

88

In [9]:

```
7 >> 2   # 00000111
         # 00000001
```

Out[9]:

1

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```
```

In [ ]:

```
```

In [54]:

```
not True
```

Out[54]:

False

In [55]:

```
not False
```

Out[55]:

True

0 and 0 - 0 0 and 1 - 0 1 and 0 - 0 1 and 1 - 1

In [56]:

```
True and True
```

Out[56]:

True

0 or 0 - 0 0 or 1 - 1 1 or 0 - 1 1 or 1 - 1

In [57]:

```
True or True
```

Out[57]:

True

In [58]:

```
True or False
```

Out[58]:

True

In [59]:

```
a = 4
b = 4
print(a == b)
```

True

In [60]:

```
s = {1,2,3}
```

In [61]:

```
s
```

Out[61]:

```
{1, 2, 3}
```

In [62]:

```
a = 10.567
```

In [63]:

```
b = "{:.2f}".format(a)
```

In [64]:

```
b
```

Out[64]:

```
'10.57'
```

In [65]:

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

In [66]:

```
id(a)
```

Out[66]:

```
94658289652768
```

In [67]:

```
id(b)
```

Out[67]:

```
94658289652768
```

In [70]:

```
a = [1, 2, 3]
b = [1, 2, 3]
a is b
```

Out[70]:

```
False
```

In [69]:

```
print(" hello " + x)
```

```
 hello Google
```

In [78]:

```
a = [1, 2, 3]
b = [1, 2, 3]
```

In [79]:

```
a.append(5)
```

In [ ]:

```

```

In [80]:

```
id(a)
```

Out[80]:

```
139773608353296
```

In [81]:

```
id(b)
```

Out[81]:

```
139773609320608
```

In [74]:

```
a = 1000000
b = 1000000
```

In [75]:

```
id(a)
```

Out[75]:

```
139773608758640
```

In [76]:

```
id(b)
```

Out[76]:

```
139773608758064
```

In [83]:

```
x =10
```

In [85]:

```
print("Hello " + str(x))
```

```
Hello 10
```

In [86]:

```
10 -- 3  #10 - (-3)
```

Out[86]:

```
13
```

In [ ]:

```
a +=1
```