Gmail

Константин Городецкий <gorodetskiykp@gmail.com>

[PyNet Learning Python 2018] - Lesson1 / Why Python, the Shell, and Strings

1 письмо

ktbyers@twb-tech.com <ktbyers@twb-tech.com> 2 февраля 2018 г., 2:00
Кому: gorodetskiykp@gmail.com

Konstantin

Note: There is a table of contents for each video at the bottom of this email including timestamps to where various content is located. This should be helpful in navigating the videos.

In this email of Learning Python we are going to cover the following:

1. **Introduction**
   Video **https://vimeo.com/243034300**
   Length is 7 minutes

2. **Why Learn Programing?**
   Video **https://vimeo.com/243905715**
   Length is 1 minute

3. **Why Python?**
   Video **https://vimeo.com/243909371**
   Length is 3 minutes

4. **Python2 versus Python3**
   Video **https://vimeo.com/243912631**
   Length is 2 minutes

5. **Characteristics of Python**
   Video **https://vimeo.com/243918300**
   Length is 5 minutes

6. **The Python Interpreter Shell**
   Video **https://vimeo.com/242411259**
   Length is 9 minutes

7. **IPython**
   Video **https://vimeo.com/242460561**
   Length is 4 minutes

8. **Printing to stdout and Reading from stdin**
   Video **https://vimeo.com/243028886**
   Length is 6 minutes

9. **Dir, Help, and Variables**
   Video **https://vimeo.com/243480156**
   Length is 10 minutes

10. **Python Strings (Part 1)**
    Video **https://vimeo.com/243481392**
    Length is 6 minutes

11. **Python Strings (Part 2)**
    Video **https://vimeo.com/243482081**
    Length is 8 minutes

12. **Python Strings (Part 3)**
    Video **https://vimeo.com/243482871**
    Length is 10 minutes

13. **Python String Formatting (Part 1)**
    Video **https://vimeo.com/243936489**
    Length is 12 minutes

14. **Python String Formatting (Part 2)**
    Video **https://vimeo.com/243956669**
    Length is 4 minutes

Additional Content:

Google Python Course on Strings

Automate the Boring Stuff with Python (Chapter 6 on Strings)
*Read up through the section on .join() and .split() string methods.

1. Python naming conventions:

a. For variable names, function names, object names, and module names use lower case separated by underscore, for example:

```
my_router
find_set_of_devices
convert_id_string_to_list
```

b. For class names, capitalize the first letter of each word. Do not use any underscores. For example:

```
ManyToManyField
ClientHistory
UserProfile
```

c. For constants, use all upper case; use underscores for word separation.

```
PI = 3.14
EMAIL_MODE
EMAIL_FROM_ADDRESS
```

Exercises

Reference code for these exercises is posted on GitHub at:

https://github.com/ktbyers/pynet/tree/master/learning_python/lesson1

1. Create a Python script that has three variables: ip_addr1, ip_addr2, ip_addr3 (representing three corresponding IP addresses). Print these three variables to standard output using a single print statement.

Make your print statement compatible with both Python2 and Python3.

If you are using either Linux or MacOS make your program executable by adding a shebang line and by changing the files permissions (i.e. chmod 755 exercise1.py).

2. Prompt a user to enter in an IP address from standard input. Read the IP address in and break it up into its octets. Print out the octets in decimal, binary, and hex.

Your program output should look like the following:

```
 $ python exercise2.py
Please enter an IP address: 80.98.100.240

    Octet1         Octet2         Octet3         Octet4
-------------------------------------------------------------
      80             98             100            240
   0b1010000      0b1100010      0b1100100     0b11110000
     0x50           0x62           0x64           0xf0
-------------------------------------------------------------
```

Four columns, fifteen characters wide, a header column, data centered in the column.

3. Create three different variables the first variable should use all lower case characters with underscore (_) as the word separator. The second variable should use all upper case characters with underscore as the word separator. The third variable should use numbers, letters, and underscore, but still be a valid variable Python variable name.

Make all three variables be strings that refer to IPv6 addresses.

Use the from future technique so that any string literals in Python2 are unicode.

compare if variable1 equals variable2
compare if variable1 is not equal to variable3

4. Create a show_version variable that contains the following

```
show_version = "*0        CISCO881-SEC-K9       FTX0000038X    "
```

Remove all leading and trailing whitespace from the string.

Split the string and extract the model and serial_number from it.

Check if 'Cisco' is contained in the model string (ignore capitalization).

Check if '881' is in the model string.

Print out both the serial number and the model.

5. You have the following three variables from the arp table of a router:

```
mac1 = "Internet  10.220.88.29         94   5254.abbe.5b7b  ARPA   FastEthernet4"
mac2 = "Internet  10.220.88.30          3   5254.ab71.e119  ARPA   FastEthernet4"
```

mac3 = "Internet 10.220.88.32   231   5254.abc7.26aa ARPA   FastEthernet4"

Process these ARP entries and print out a table of "IP ADDR" to "MAC ADDRESS" mappings. The output should look like following:

```
             IP ADDR          MAC ADDRESS
------------------- --------------------
        10.220.88.29       5254.abbe.5b7b
        10.220.88.30       5254.ab71.e119
        10.220.88.32       5254.abc7.26aa
```

Two columns, 20 characters wide, data right aligned, a header column.

CLASS OUTLINE

```
1. Introduction (VIDEO1)
   A. Purpose of the course  [0:11]
   B. Who is this video series for?  [0:33]
   C. About me  [2:51]
   D. Course logistics  [3:29]
   E. Apply what you learn  [4:23]
   F. Context for the exercises  [4:49]

2. Why Learn Programming? (VIDEO2)
   A. Programming as a power skill  [0:10]
   B. Still need to retain core engineering skills  [0:20]
   C. Automation is major trend in our industry  [0:30]
   D. Range of potential programming skills  [0:54]

3. Why Python? (VIDEO3)
   A. Large, active community  [0:20]
      1. Libraries
      2. Learning resources
      3. People you can ask questions to
   B. High-level language  [1:29]
   C. Widely available on systems  [1:53]
   D. Readability  [2:05]
   E. Supports beginners through advanced programmers  [2:33]

4. Python2 versus Python3? (VIDEO4)
   A. What should network engineers do (as of today)?  [1:20]

5. Characteristics of Python (VIDEO5)
   A. Blocks of code are indicated by indentation  [0:13]
   B. Python conventions  [2:02]
      1. Follow the Python conventions (PEP8)   [2:12]
```

C. What is Python like as a language? [3:20]
D. High-level language [3:58]
1. Dynamically typed variables [4:12]

6. The Python Interpreter Shell VIDEO6)
A. Launch the interpreter shell [0:05]
B. Running python 3.6.2 [0:33]
C. Creating a variable [1:00]
1. Variable naming conventions [1:17]
D. Using type [1:43]
E. Characters allowed in variable names [2:08]
1. Upper case / lower case / numbers [2:23]
2. Can't start with a number [2:36]
3. Can start with an underscore [2:47]
F. Automatic evaluation of expressions [3:42]
G. First Python program [4:15]
H. Printing to standard output [5:21]
I. Making the program executable at the system level [6:22]
J. Editing and Editors (to create Python programs) [8:30]
K. IDEs [8:53]

7. IPython (VIDEO7)
A. Launching IPython [0:13]
1. Better formatting [0:47]
2. History buffer / improved history [1:13]
3. Command completion [1:40]
B. Installing IPython [2:30]
C. More on variable names [3:00]
D. Comments [3:28]

8. Printing to stdout / Reading from stdin (VIDEO8)
A. print() in Python3 [0:14]
1. Differences in Python2 [1:00]
2. Writing PY2 and PY3 compatible code [1:41]
B. Python3 reading from stdin (input) [2:43]
1. Differences in Python2 [3:31]
2. Writing PY2 and PY3 compatible code [4:18]

9. Dir, Help, and Variables (VIDEO9)
A. Using dir() [0:24]
B. Calling methods versus referring to methods [0:55]
1. Examples calling various string methods [2:16]
2. Chaining methods [2:39]
C. Using help() [3:34]
D. Assignment operator [5:44]
E. Variables as references to objects [6:10]
1. Using id() [6:16]
2. Two names that refer to the same thing [7:17]
3. Reassigning a name to a different object [8:40]

10. Strings (Part1) (VIDEO10)
    A. Fundamental Difference between Python2 and Python3 [0:30]
    B. String in Python2 is an ASCII string [1:10]
    C. String in Python3 is a Unicode string [1:59]
    D. How to write Unicode strings in Python2 [2:08]
    E. How to write byte strings in Python3 [3:23]
    F. Technique for making all string literals unicode [3:46]

11. Strings (Part2) (VIDEO11)
    A. Comparison Operators [0:30]
    B. Substring in broader string [2:16]
    C. String indexes [3:51]
    D. len() function [4:50]
    E. String concatenation [5:14]
    F. Binary and hex representations of strings [6:43]

12. Strings (Part3) (VIDEO12)
    A. Raw Strings [1:04]
    B. Assigning strings (single, double, triple quotes) [1:44]
    C. repr() of a string [3:32]
    D. String methods [4:21]
       1. strip() [4:34]
       2. split() [6:04]
       3. splitlines() [9:15]

13. String Formatting (Part1) (VIDEO13)
    A. Replicating a string [1:12]
    B. Format Method [1:35]
       1. Calling with positional arguments [2:00]
       2. Specifying field width [2:59]
       3. Right aligned, centered [3:21]
       4. Using named arguments [6:20]
       5. Using *args to pass in a list [10:50]

14. String Formatting (Part2) (VIDEO14)
    A. Using the format operator (%) [0:25]
    B. F-strings [1:41]

Kirk Byers