Python basics

22 February 2018 14:09

name = raw_input("What is your name? ")
# Out: What is your name? _

Note: We need to use raw_input in python 2 version

a=input("enter")

raw_input -->To input Strings
input --> To input values (integer,float)

https://stackoverflow.com/questions/4960208/python-2-7-getting-user-input-and-manipulating-as-string-without-quotations

number=int(raw_input("enter"))

string=str(raw_input("enter"))

type(string) ---> **<type 'str'>**
Module:(Python File)

A module can be created by creating a **.py** file.

hello.py

Functions in a module can be used by **importing the module.**

Modules can be imported by other modules.

```
import hello
hello.say_hello()
hello.say_welcome()
```

Specific functions of a module can be imported.

```
from hello import say_welcome
say_welcome()
```

Modules can be aliased.

```
import hello as ai
ai.say_hello()
```

A module can be stand-alone runnable script.

Main function:

```
if __name__== '__main__':
    hello()
```

*Python executes each lines in order.

In python by default name=main

__name__= ='__main__'

While importing other modules, __name__= imported module
import hello

__name__= = hello #Python file name

https://stackoverflow.com/questions/419163/what-does-if-name-main-do

You Tube video:

Concept behind : if __name__ == "__main__"

Learn How To Automate Work with Python Programming Course Complete

https://www.youtube.com/watch?v=XWkLyn0Fct4

Python Tutorial for Beginners From the Basics to Advanced 1/2

Pycharm

Configure Pycharm for GIT

To clone GIT projects

1)Close existing project
2)Version control-->GIT

https://www.jetbrains.com/help/pycharm/set-up-a-git-repository.html

Python with EXCEL
Become an Excel Wizard With Python

Hints:

String Formatting:
==========

Curly braces are replaced with the values passed.

```
foo = 1
bar = 'bar'
baz = 3.14

print('{}, {} and {}'.format(foo, bar, baz))

# Out: "1, bar and 3.14"


name="Suresh"
print("hello  {}".format(name))
#Out:  hello  Suresh


my_list = ['zero', 'one', 'two']
print("2nd element is: {0[2]}".format(my_list))     # "0" is optional
# Out: "2nd element is: two"
```

You Tube video:

Concept behind : if __name__ == "__main__"

Python Tutorial: if __name__ == '__main__'

If the module is inside a directory and needs to be detected by python, then the directory should contain a file named
__init__.py.

[__init__.py]

str()

For strings, this returns the string itself.

str(x) x is an arguments to print

Print the string.

str is equivalent to Print function.

```
s="SURESH"
str(s)
```

Op: SURESH

```
name=5
>>> str(name)
#  '5'
```

Converts the integer value to String value

Comments:

Comments spanning multiple lines have """ or ''' on either end. This is the same as a multiline string, but they can be used as comments:

```
"""
Suresh
"""
```

==========

FOR LOOP:

```
for i in range(stop):
```

for i in range(5): **for i in range(1,5) : start,stop** **Range 5--> 0-4**
print i **(1,5)--> 1-4**

OP:

0
1
2
3
4

For iterating over LIST

```
for x in ['one','two','three']:
    print x
```

OP:

one
two
three

Lists allows to use slice notation as **list[start:end:step]**

```
print("2nd element is: {0[2]}".format(my_list))    # "0" is optional
# Out: "2nd element is: two"
```

% type:

```
a=10
print("value of a is %i" %a)
# value of a is 10
a=10
b=20
print("value of a and b %i  %i" %(a,b))
# value of a and b is 10 20

string="Suresh"
print("String value is %s" %string)
```

Hints:
=====

```
name="Suresh"
>>> print("Name is " +name)
#Name is Suresh

>>print("his name"  +name  +"was good boy")
#his name Suresh was good boy
```

+name string concatenation

Variables:

https://stackoverflow.com/questions/17153779/how-can-i-print-variable-and-string-on-same-line-in-python

---

Since Python 3.x the print is actually a function, so it now takes arguments like any normal function.

The end=' ' is just to say that you want a space after the end of the statement instead of a new line character. In Python 2.x you would have to do this by placing a comma at the end of the print statement.

Python 3+

```
while i<5:
   print(i, end = ' ')
   i=i+1
```

Will give as output:

0 1 2 3 4

Python 2+

```
while i<5:
   print(i),
   i=i+1
```

Will give as output:

0 1 2 3 4

In Python 2.x, to continue a line with print, end the print statement with a comma. It will automatically add a space.

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

three

Lists allows to use slice notation as **list[start:end:step]**

---

break and continue Statements:

- ***break*** terminates the loop completely and proceeds to the first statement following the loop
- ***continue*** terminates the current iteration and proceeds to the next iteration

==========

How to pass a file as an input in python?

Input from a File:

```
f=open("test.txt","r")          --->mode='r'  reading the file
print(f.name())
f.close()
```

We need to close the file after done. In open function Default mode is **'READ'**

In order to overcome this issues like closing every time after finished. We are going to use context manager (i.e) with because of this so after it executed it will be closed automatically.

r+ mode both read and write the files
r read operation
w write operation

For images we need to use bytes (b) to read and write,
To read the images from a file -- **rb**
To write the images to a file-- **wb**

https://www.programiz.com/python-programming/file-operation

Syntax:

with <command> **as** <name> :

```
with open('somefile.txt', 'r') as fileobj:
   # write code here using fileobj
```

This ensures that when code execution leaves the block the file is automatically closed.

```
with open("D:\Serial.txt",'r') as file:
      print(file.readlines())
```

Useful links for reading and writing the text files:

https://stackoverflow.com/questions/4710067/
eting-a-specific-line-in-a-file-python

```
#    dir(f)    # it returns all available functions.
```

['__class__', '__delattr__', '__doc__', '__enter__', '__exit__', '__format__', '__getattribute__', '__hash__', '__init__', '__iter__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', 'close', 'closed', 'encoding', 'errors', 'fileno', 'flush', 'isatty', 'mode', 'name', 'newlines', 'next', 'read', 'readinto', 'readline', 'readlines', 'seek', 'softspace', 'tell', 'truncate', 'write', 'writelines', 'xreadlines']

```
space.
print "Hello,",
print "World!"
# Hello, World!
```

```
#A demo of Python 'not' with 'in' operator

List = [5, 10, 15, 20, 25, 30]

for a in List:

   if not a in (10,25):     checking a is present in a list or not.(10,25) is a tuples(List)

      print ("List Item: " ,a)
```

Try and except:

```
  --> Try means it will display the results once it is true
  ---> except part will be displayed if the output is False.

   num=2
>>> try:
      if num>3:
            print ("hello")
except:
      print("Please check")

      Output:
>>> Please check
```

How to PASS variable as an Arguments in Python:

Command: **python test.py Suresh**

Code:

```
import sys
name=sys.argv[1]
print name
```

Output: **Suresh**

System argument variable:

```
sys.argv[0]=test.py
sys.argv[1]=Suresh
```

```
+-----------------------------------------------------------------+

 message="suresh-kumar-arul"

 message.split('-')
 ['suresh', 'kumar', 'arul']

 a,b,c=message.split('-')   # splits the string using '-' as separator
print a,b,c
suresh  kumar  arul

==================================
a,b,c=['suresh', 'kumar', 'arul']

a='suresh'
b='kumar'
c='arul'
```

'mode', 'name', 'newlines', 'next', 'read', 'readinto', 'readline', 'readlines', 'seek', 'softspace', 'tell', 'truncate', 'write', 'writelines', 'xreadlines']

```
>>> f=open("D:\harish.txt",'w')
>>> f.write("harish")
>>> f.write("\n suresh")
>>> f.close()                              #close function is used to close the file
>>> f=open("D:\harish.txt",'r+')
>>> f.readlines()                      # Reads each lines in the form of a LIST
['harish\\m suresh']
>>> f.write("\n this is line")
>>> f.close()                      #save And close the file
```

```
>>> f.tell()      # gets the current file position
56

>>> f.seek(0)   # bring file cursor to initial position
0

>>> f.read()    # read the entire file

>> f.read(21)   #Reads the file after the cursor 21

f.truncate()    #removes values after the indexes.
```

Reading a file line-by-line: **using for loop**

```
with open('myfile.txt', 'r') as f:
    for line in f:
        print(line)
```

==========

Shutil Module:

How to copy files and Move files? **(Shutil module)**

shutil.copy(Source_path,destination_path)

```
shutil.copy("D:\harish.txt","D:\userdata\suvs\Desktop\Destination")
shutil.copy("D:\harish.txt","D:\userdata\suvs\Desktop\Destination\suresh.txt")
```

shutil.copytree(source_dir_path,destination_path_dir)

Recursively copy the entire directory tree rooted at src to dest.

shutil.move(Source_path,detination_path)

shutil.rmtree(path)

Removes the directory.

Useful links to learn:

https://www.pythonforbeginners.com/os/python-the-shutil-module

==========

Subprocess Module:

If you need to read also the standard error, into the Popen initialization, you can set stderr to subprocess.PIPE or to subprocess.STDOUT:

b= kumar
c='arul'

==========

To delete Python Variable:

```
del your_variable
```

Attributes is also called as Variables.

stdout=subprocess.PIPE, stderr=subprocess.STDOUT

```
import subprocess
p=subprocess.Popen("pwd", stdout=subprocess.PIPE)
result=p.communicate()
print result

# ('/var/robot','None')

result=p.Communicate()[0]

#/var/robot
```

==========

Functions and Classes:

Functions:

```
 def func(list):
     for x in list:
          print x

Num=[1,2,3,4,5]
func(Num)            # Passing list variable to function
```

```
def sum(a,b):
    print a+b

sum(5,2)
```

Classes:

Basic Syntax:

```
class student:
      ------

 instance=student()
```

Class Example:

In class, all the functions are specified with **self** instance as a first parameter.

https://micropyramid.com/blog/understand-self-and-__init__-method-in-python-class/

```
class Students:
     def __init__(self,name,age,grade):    #Self is an instance of the class
          self.name=name
          self.age=age
          self.grade=grade

    #name,age,grade are  attributes(Variables)

student1=Students("Suresh",23,"12th")        #class instance
```

Note:

4 arguments are passed, self is an implicit argument. Here Students consists of self and 3 arguments.

- Self arguments are implicitly(automatically) added in the functions call.

  Self are explicitly(manually) added in the function definition

---

>>>student1.name
>>>'Suresh'
>>>student1.age
>>>23
>>>student1.grade
>>'12th'

__init__ function executes during the intialization of object.

https://stackoverflow.com/questions/46448875/class-takes-no-arguments-1-given/46448888

==========

How to install Python libraries?

06 June 2019 09:17

To install python libraries we have to use pip

--> **pip install numpy**

To install the all the libraries using requirements.txt file:

---> pip install --help

-r, --requirement --- Install from the given requirements file. This option can be used multiple times.

requirements.txt

#This file contains the list of libraries to be installed

numpy
pandas

---> **pip install -r requirements.txt**

To disable cache when installing python libraries: **[--no-cache-dir]**

-----> pip install numpy --no-cache-dir

requirements.txt

Python Libraries

numpy
pandas

Python Cheatsheet

29 April 2019 00:39

Comprehensive Python Cheatsheet

https://github.com/gto76/python-cheatsheet#dictionary

Python Regular Expression

23 July 2018 12:17

- A Regular Expression is a Special text String for describing a search pattern

Pycharm import regular expressions.

Multiple lines String we have to enclosed with ''' …..'''

import re

Nameage='''
Suresh is 23 and Akhil is 22
Lukas is 21 and kumar is 20
'''
ages=re.findall(r'\d{1-3}',Nameage)
Name of the string

\d--> It will take only one digit eg: [0-9]--->5
\d+ --> More digits combined --->45
\+ --> 1 or more (append)
\w--> Single character --->a
\w+ --->Entire String --->apple
\s Space
. Matches Any string
* 0 or more (append)
^ string starts with
$ string ends with

out=re.findall(**"\d+/\d+/\d+\s\d+:\d+:\d+.*\[\w+]\s\s\w+\s+\w+\s::\s\w+\s\w+\s\w+:"**,string)

(r".....")

You may notice that the pattern variable is a string prefixed with r, which indicates that the string is a **raw string literal.**

Escaping Special Characters:

Special characters (like the character class brackets [and] below) are not matched literally:

match = re.search(r'[b]', 'a[b]c')
match.group()
# Out: 'b'

By escaping the special characters, they can be matched literally:
match = re.search(r'**\[b\]**', 'a[b]c')
match.group()
# Out: '[b]'

re.findall()

Regular Expression **findall** finds all the occurrences of the String.

Python regular Expression Cheat Sheet:

\d
Matches any decimal digit; this is equivalent to the class [0-9].
\D
Matches any non-digit character; this is equivalent to the class [^0-9].
\s
Matches any whitespace character; this is equivalent to the class [\t\n\r\f\v].
\S
Matches any non-whitespace character; this is equivalent to the class [^ \t\n\r\f\v].
\w
Matches any alphanumeric character; this is equivalent to the class [a-zA-Z0-9_].
\W
Matches any non-alphanumeric character; this is equivalent to the class [^a-zA-Z0-9_].

Python Regular expression:

Frame\s+[0-9]+:
Frame 162:

```
mem='call me 415-555-1011 tomorrow or at 415-555-9999'
>>> reg=r'\d\d\d-\d\d\d-\d\d\d\d'
>>>re.findall(reg,mem)
['415-555-1011', '415-555-9999']
```

Syntax:

re.compile(pattern)
re.search(string)

```
find=re.compile(r'\d\d\d-\d\d\d-\d\d\d\d')
>>> find.search('call me 415-555-1011 tomorrow or at 415-555-9999')
```

r'[a-zA-Z]'
r'[aeiouAEIOU]'
r'[aeiou]' -- Checking for the vowels
r'[^aeiou]' --- caret symbol (^) means negative, it will check strings without vowel characters
r"^\d$" --- Starts(^) with digits and ends($) with digit

```
text=re.complile(pattern)
 text.findall(string)    #calling with string variable name

 text=re.compile(r'.at')
 text.findall("Cat and Dog")
>> 'Cat'
```

(.) dot char looking for only single character Eg: .at so Cat

```
text=re.compile(r'Cat .* Dog')
 text.findall("Cat and Dog")
>> 'and'
```

.* looking for the characters after the string

Match regex with Group Pattern:

https://stackoverflow.com/questions/22989241/python-re-example

Ordinarily, parentheses create a "capturing" group inside your regex

```
 string="set var = 12"
 reg=r"(set|let) var = (\w+|\d+)"
 re.findall(reg,string)
```

Output: [('set', '12')]

As you see whatever is inside parentheses is captured in "**groups**." But you might not care about all those groups. Say you only want to find what's in the second group and not the first. You need the first set of parentheses in order to group "get" and "set" but you can turn off capturing by putting "**?:**" at the beginning:

```
string="set var = 12"
reg=r"(?:set|let) var = (\w+|\d+)"
re.findall(reg,string)
```

Output: ['12']

[0-9][0-9]\/[0-9][0-9]\/20[0-9][0-9].*\[SchedulerThread\]\s+INFO\s+PLXAbstractRAReportModule\s::\sGot\sBE\sgroup:.*

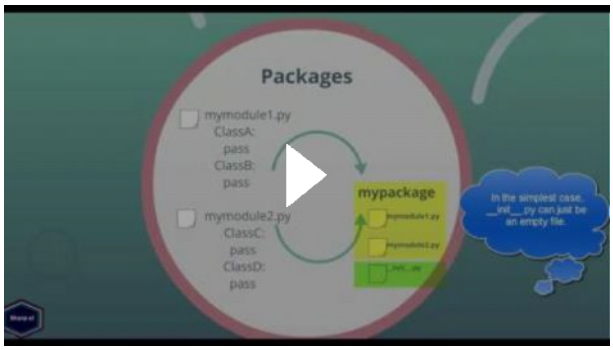
29/08/2018 20:35:00,046 [SchedulerThread] INFO PLXAbstractRAReportModule :: Got BE group:z1sdvctbe1401vm001

dot with asterik(**.***) means it will take all the values

Python Modules

30 August 2018 12:53

Python 3: Modules & Packages



Openpyxl Module for Excel Automation: (import openpyxl)

1) wb=openpyxl.load_workbook("GET_Registration_Sheet.xlsx")

2) wb.get_sheet_names()
   [u'GET Registration Sheet', u'Sheet1', u'Sheet2', u'Sheet3']

3) sheet=wb.get_sheet_by_name('GET Registration Sheet')

4) sheet['A1'].value
   u'Candidate Registration Sheet'

5) sheet['A2'].value
   u'Candidate Personal Data'

6) sheet['A3'].value
   u'S.No'

7) sheet['D4'].value
   8148300350L

   sheet["b4"].value

   Both it will accept capital and Small

8) **sheet['A3'] = 45** #Assigning Values to the cells
9) Once all the changes are done we have to the save the workbook
10) wb.save("New_name.xlsx")

wb.sheetnames ---> to Display sheet names

wb.save("example.xlsx")

sheet.cell(row=4,column=2).value

sheet.max_row
sheet.max_column

from openpyxl.styles import font

for i in range 10
sheet['A'+str(i)].value='suresh'

OS MODULE:

os.getcwd()
Returns the current working directory.

Eg: **os.getcwd()**
'C:\\python27'

It is displaying with double slash.

os.chdir('Path')
Change working directory

os.chdir('D:\\Excel Automate')
os.chdir("D:\\suresh")
os.chdir("D:\Suresh")

DOUBLE OR SINGLE quotes both are fine
Path we need to specify with double backslash (\\ )
Single backslash (\) also accepted.

os.listdir(path)
Displays all files and folders inside the directory.

os.unlink(path)
To delete a file

os.rmdir(path)
To delete a directory

os.path.join('a','b','c')

Output: a\\b\\c

os.path.join(os.getcwd(),"suresh.txt")

Output:
'C:\\python27\\suresh.txt'

os.path.split(os.getcwd())

Folder separator in Windows and Linux:

For Windows: (\) backslash

For Linux and mac : (/) forward slash

DATETIME MODULE:

import datetime
print datetime.datetime.now()

today = datetime.date.today()
print('Today:', today)
yesterday = today - datetime.timedelta(days=1)
print('Yesterday:', yesterday)

Output:

Today: 2016-04-15
Yesterday: 2016-04-14

for i in range 10
sheet['A'+str(i)].value='suresh'

****

>>> wb = load_workbook(filename = 'empty_book.xlsx')
>>> sheet_ranges = wb['sheet names'] #sheet name
>>> print(sheet_ranges['D18'].value)

To find max Rows and Max Column:

https://stackoverflow.com/questions/13377793/is-it-possible-to-get-an-excel-documents-row-count-without-loading-the-entire-d

To convert Excel(.xls) to (.xlsx)

import pyexcel as p

p.save_book_as(file_name='your-file-in.xls',
dest_file_name='your-new-file-out.xlsx')

https://stackoverflow.com/questions/9918646/how-to-convert-xls-to-xlsx

>>>import pyexcel
>>> excel_file="D:\userdata\suvs\Desktop\Destination/"+"suresh.xls"
>>> pyexcel.save_book_as(file_name=excel_file,dest_file_name=excel_file + "x")

suresh.xls+x suresh.xlsx

Excel using Dictionary format:

https://stackoverflow.com/questions/14196013/python-creating-dictionary-from-excel-data

Requests Module in Python:

Requests is a Python module that you can use to send all kinds of HTTP requests.
https://www.dataquest.io/blog/python-api-tutorial/

requests.get(URL)

It will returns the response code Eg: 200 (SUCCESS)

GITHUB API: https://api.github.com/

>>> r = requests.get('https://api.github.com/user', auth=('user', 'pass'))
>>> r.status_code
200
>>>r.text

>>>res.raise_for_status() # it raises error if any issue (Eg: 404 not Found)

>>> import requests
>>> r=requests.get("http://api.open-notify.org/iss-now.json")
>>> print(r.content)
{"iss_position": {"longitude": "30.4897", "latitude": "41.4381"}, "message": "success", "timestamp": 1538657876}
>>>

GET DATA:

```
import requests
url="https://api.github.com/"
response=requests.get(url)
data=respone.json()
print data
```

Selenium module:

For web automation

```
>>> from selenium import webdriver
>>> browser=webdriver.Chrome()
>>> browser.get("https://automatetheboringstuff.com/")
>>> elements=browser.find_element_by_css_selector("body > div.main > div:nth-child(1) > ul:nth-
child(18) > li:nth-child(1) > a")
>>> elements.text()

>>>elements.text
>>>u'Chapter 0 \u2013 Introduction'
```

From selenium module importing webdriver

```
>>> from selenium import webdriver
>>> browser=webdriver.Chrome()
>>> browser.get("https://www.google.com/")
>>> elements=browser.find_element_by_id("lst-ib")
>>> elements.send_keys("cribuzz")
>>> elements.submit()

>>>browser.back()
>>>browser.refresh()
>>>browser.forward()
```

SMTP Protocol: Module (smtplib)

Simple mail transfer protocol to send mails.

```
import smtplib
connection= smtplib.SMTP('smtp.gmail.com',587)      # Connection is an object
connection.ehlo()
connection.starttls()   #To start encryption before login
connection.login("karthickcj0083@gmail.com","8124386626")
connection.sendmail("karthickcj0083@gmail.com","karthickcj0083@gmail.com","Subject: Test mail")
connection.quit()
```

Provider	smtp server domain name
Gmail	smtp.gmail.com
Outlook	smtp-mail.outlook.com

Sending Email and Text Messages:

https://automatetheboringstuff.com/chapter16/

Paramiko Module-- SSH using Python

```
import paramiko
ssh=paramiko.SSHClient()
ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
ssh.connect('10.76.147.23',22,'centos','centos')
stdin, stdout, stderr =ssh.exec_command('ls')
stdout.read()
```

https://daanlenaerts.com/blog/2016/01/02/python-and-ssh-sending-commands-over-ssh-using-paramiko/

Python LIST and Dictionary Variables

04 October 2018 19:14

LIST:

Enclosed within square brackets. [ ]

List index starts with Zero

List = [1, 2, 3, 4]
List[0] # 1
List[1] # 2

Negative indices are interpreted as counting from the end of the list.

List[-1] #4 Last Value in the list

Important tips:

Python--Customized scripts

>>> **[i for i in range(5)]**
[0, 1, 2, 3, 4]

```
>>  a=[i for i in range(5)]
>>  print a
  [0, 1, 2, 3, 4]
>> type(a)
  <type 'list'>
```

To append the values to the list

```
list = []
for i in range(2, 11, 2):
   list.append(i)
```

The above for loop can be rewritten as a list comprehension:

list = [i for i in range(2, 11, 2)]

https://stackoverflow.com/questions/33553046/storing-values-from-loop-in-a-list-or-tuple-in-python

How to use shortcut for Tuples:

Hints:

tuple(i for i in range(1, 10, 2))

output: (1,3,5,7,9)

How to append values to the tuples:

```
tup1=()
for i in range(1,10,2):
  tup1+= (i,)
print tup1
```

This prints **(1, 3, 5, 7, 9)**

lst = range(1,10,2) or tup = tuple(range(1,10,2))

Dictionary variable:

Enclosed with curly braces { }
Key: Value pairs

How to Construct Dictionary?

Important points:

You can also construct a dictionary with the built-in dict() function. The argument to **dict()** should be a sequence of key-value pairs. A **list of tuples** works well for this:

https://realpython.com/python-dicts/

```
dictionary = {"Hello": 1234, "World": 5678}
print(dictionary["Hello"])
```

The above code will print 1234.

```
dictionary={'name':'suresh','age':23}

dictionary = {

'name' : 'suresh',
'age':23

}
```

+----------+

```
dict = {}
dict['one'] = "This is one"
dict[2]     = "This is two"

dict={'one': 'This is one', 2: 'This is two'}

tinydict = {'name':'john', code':6734, 'dept': 'sales'}

print (dict['one'])      # Prints value for 'one' key
print (dict[2])          # Prints value for 2 key
print (tinydict)         # Prints complete dictionary
print (tinydict.keys())  # Prints all the keys   ['dept', 'code', 'name']
print (tinydict.values()) # Prints all the values ['sales', 6734, 'john']
```

How to access lists of dictionary:

```
>>> dict_var={"controller": {"username": ["v962_vivekya"], "keystone_version": ["v2"]}}
>>> print dict_var['controller']
{'username': ['v962_vivekya'], 'keystone_version': ['v2']}
```

```
>>> print dict['controller']['username']
['v962_vivekya']

>>> print dict['controller']['username'][0]
v962_vivekya
>>>
```

How to add new items to dictionary? **Very important concept**

Adding items to dictionary

```
default_data = {
        'item1': 1,
        'item2': 2,
}
```

>> default_data['item3'] = 3

if you want to insert multiple items at once. you can use update to add more than one item. An example:

>> **default_data.update({'item3': 3})**

>> **default_data.update({'item4': 4, 'item5': 5})**

https://stackoverflow.com/questions/6416131/python-add-new-item-to-dictionary

==========

SELF Argument:

This is because methods in a class expect the first argument to be **self.** This self parameter is passed internally by python as it always sends a reference to itself when calling a method, even if it's unused within the method

```
class MyClass:
   def say(self):
      print("hello")
mc = MyClass()
mc.say()
```

>> hello

https://stackoverflow.com/questions/46448875/class-takes-no-arguments-1-given/46448888

Flask Framework

03 December 2018 21:48

Flask framework
api using python

https://carolinafernandez.github.io/development/2017/11/11/Upload-files-through-proxy-rest-api

import flask

Create an Flask object using the code

app=flask.Flask(__name__)

To run our application: (Last line of the code and important to be mentioned)

app.run()

To route the flask application:

```
@app.route('/')
def home():
    return 'Suresh'
```

home function is mapped to the path '/'
Flask executes this home function.

app.debug = True Debug mode

Passing as argument app.run

app.run(debug=True)

21 January 2019 22:39

Removes duplicates

https://stackoverflow.com/questions/7961363/removing-duplicates-in-lists

Reverse string
https://stackoverflow.com/questions/36949665/fastest-way-to-reverse-a-string-in-python

Hints:

num = 3
type(num) **# displays type of the variable**

<type 'int'>

import requests
req=requests.get("URL")

print(dir(req)) # displays all the methods and attributes corresponding to the variable

print(help(req)) #displays help section

Python xlrd and xlwt

05 February 2019 19:09

Python using xlrd:

https://stackoverflow.com/questions/23568409/xlrd-python-reading-excel-file-into-dict-with-for-loops

```
import xlrd
workbook=xlrd.open_workbook("excelsheet.xlsx")
sheet=workbook.sheet_by_name('sheet1')
read_cell_value=sheet.cell_value(row,column)          # To read cell value from the excel sheet
Print read_cell_value                                 # Hello
```

Python using xlwt

xlwt supports only xls format to write

https://stackoverflow.com/questions/25430110/detail-attributeerror-module-object-has-no-attribute-workbook

```
import xlwt
workbook = xlwt.Workbook()
sheet = workbook.add_sheet('Eswar')
sheet.write(0,0,'Test passed')                # write(row,column,value)  --> row-0 means 1st row and column 0 means 1st column
workbook.save("D:\\resultsLatest.xls")        #enter code here
```

Purpose of self

06 February 2019 19:17

Purpose of self in python:

https://stackoverflow.com/questions/2709821/what-is-the-purpose-of-self

Pandas

20 February 2019 19:06

https://stackoverflow.com/questions/17977540/pandas-looking-up-the-list-of-sheets-in-an-excel-file

Instance methods, Class methods, static methods

05 March 2019 11:13

https://realpython.com/instance-class-and-static-methods-demystified/

Python Hints for interview- ERRORS

07 March 2019 10:49

tuple' object does not support item assignment

```
>>> name="suresh"
>>> name[0]
's'
>>> name[0]='k'

Traceback (most recent call last):
  File "<pyshell#18>", line 1, in <module>
    name[0]='k'
TypeError: 'str' object does not support item assignment
```

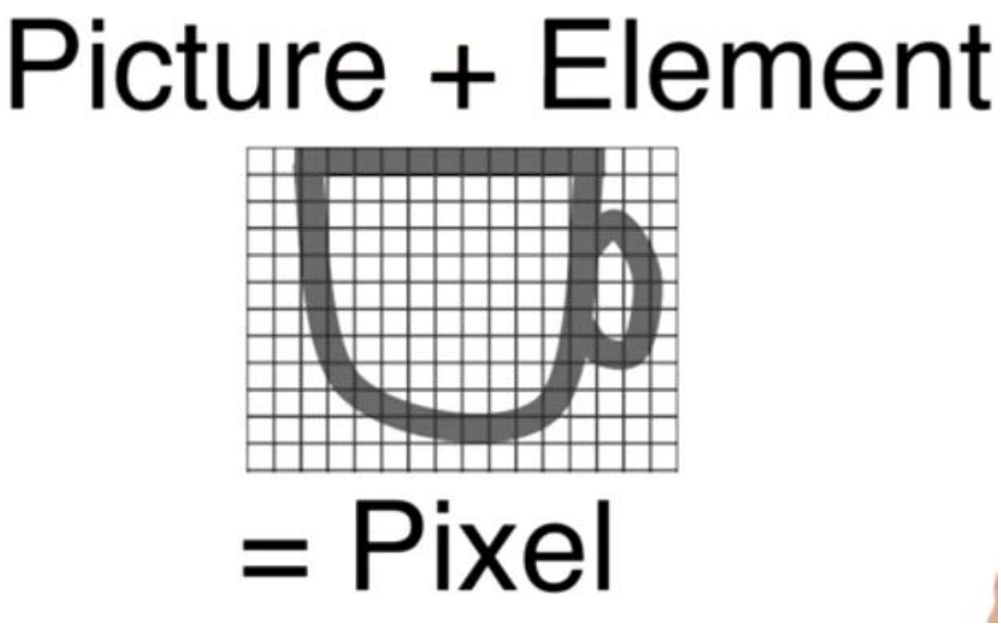
How Images works?

28 May 2019 16:08

Images are represented in Pixel

It is a square box

[How do computers store images?]



How to use Python2 and Python3 in same machine?

29 May 2019 19:26

https://stackoverflow.com/questions/341184/can-i-install-python-3-x-and-2-x-on-the-same-windows-computer

Here's my setup:

1. Install both Python 2.7 and 3.4 with the windows installers.
2. Go to `C:\Python34` (the default install path) and change python.exe to python3.exe
3. **Edit** your environment variables to include `C:\Python27\;C:\Python27\Scripts\;C:\Python34\;C:\Python34\Scripts\;`

Now in command line you can use `python` for 2.7 and `python3` for 3.4.

From <https://stackoverflow.com/questions/341184/can-i-install-python-3-x-and-2-x-on-the-same-windows-computer>

Python API Basics

17 June 2019 18:44

To learn how Python API works

API documentation:

https://2.python-requests.org//en/v0.10.6/api/

https://2.python-requests.org/en/master/api/

Quick start guides:

https://2.python-requests.org/en/master/user/quickstart/

To run simple HTTP server using python module?

python -m SimpleHTTPServer 8080

How to pass the arguments to the requests?

params--> this argument is used to add query string (which means filter) along with the request (eg: "name"="suresh" url=http://www.github.com?name=suresh)

data --> To add requests body parameters. This is used to send data to the request in POST method.

auth--> to add username and passwords

headers---> To add request headers like content-type

json---> To pass the json data to the server. This is used in the POST method. Json data is sent to the server to process the POST (creation).

200 -OK
201 - Created in POST request

202 -Accepted

1) **How to use python requests authentication token?**

- We have to add the authentication token in a form of json in headers section.
- We have to generate token and we need to use it in the api requests.

headers={'Authorization': 'access_token myToken'}

r=requests.get(url, headers=headers)

https://stackoverflow.com/questions/13825278/python-request-with-authentication-access-token

2) **How to add body parameters in the api requests?**

data – (optional) Dictionary or bytes to send in the body of the Request.

We have to pass all body parameters in data section.

https://stackoverflow.com/questions/11832639/how-to-specify-python-requests-http-put-body

3) **What is the meaning of content-type in api requests?**

To get json values in an response, we have to use application/json.

For JSON:

Content-Type: application/json

For JSON-P:

Content-Type: application/javascript

https://stackoverflow.com/questions/477816/what-is-the-correct-json-content-type

Application/x-www-form-urlencoded:

For **application/x-www-form-urlencoded**, the body of the HTTP message sent to the server is essentially one giant query string -- name/value pairs are separated by the ampersand (&), and names are separated from values by the equals symbol (=). An example of this would be:

MyVariableOne=ValueOne&MyVariableTwo=ValueTwo

4) **How to setup API requests?**

I faced some of the errors below and found the proper solution for that.

https://stackoverflow.com/questions/39737820/requests-exceptions-sslerror-ssl-certificate-verify-

failed-certificate-verif

ERROR: ssl.SSLError: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed (_ssl.c:600)

Solution:

API request verifies SSL certification when sending the request. So we have to disable the verify option to perform API operations

verify= False

The **Content-Type** entity header is used to indicate the media type of the resource.

In responses, a Content-Type header tells the client what the content type of the returned content actually is. Browsers will do MIME sniffing in some cases and will not necessarily follow the value of this header; to prevent this behavior, the header X-Content-Type-Options can be set to nosniff.

In requests, (such as POST or PUT), the client tells the server what type of data is actually sent.

From <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Type>

Difference between POST,PUT,PATCH:

https://stackoverflow.com/questions/31089221/what-is-the-difference-between-put-post-and-patch/31698882

Safari download py scripts

09 July 2019 19:34

https://github.com/yshean/safari-video-downloader/blob/master/safari_downloader.py

Python Selenium

19 July 2019 16:33

Python API

https://selenium-python.readthedocs.io/api.html#module-selenium.webdriver.chrome.options

https://stackoverflow.com/questions/18557275/how-to-locate-and-insert-a-value-in-a-text-box-input-using-python-selenium

```
from selenium import webdriver

### CBAM ADMIN GUI #####

chrome_options = webdriver.ChromeOptions()
chrome_options.add_argument('--headless')
chrome_options.add_argument('--no-sandbox')
chrome_options.add_argument('--disable-dev-shm-usage')
browser=webdriver.Chrome("/home/centos/chromedriver",chrom
e_options=chrome_options)
browser.get("https://135.112.219.179/auth/admin")

inputUsername=browser.find_element_by_name("username")
inputUsername.send_keys("cbamuser")
inputPassword=browser.find_element_by_name("password")
inputPassword.send_keys("vtc#8Net")

loginButton=browser.find_element_by_name("login")
loginButton.click()

browser.close()
```

Python Codes from web

29 July 2019 19:53

https://searchcode.com/codesearch/view/3355742/

Jinja2 Template system

31 July 2019 13:45

How easy jinja?

Let's gets started with it

http://zetcode.com/python/jinja/

https://gist.github.com/sevennineteen/4400462

>>> from jinja2 import Template

Original JSON:

string='''{
"cbam": {
"image": "cbammultinode-IMAGE",
"model_version": "V1"}'''

>>> **tem={'name':'suresh','age':23}**

Jinja2 Template system:

```
>>> string='''{
 "cbam": {
  "image": {{item.name}},
  "model_version": {{item.age}}}'''

>>> template=Template(string)
>>> msg=template.render(item=tem)        ---> tem is nothing but the JSON variable name  (item is replaced with JSON variable name)
>>> print msg
{
 "cbam": {
  "image": suresh,
  "model_version": 23
>>>
```

==========

```
with open('sample_model.json','r') as data:
    model_var = json.load(data)                    #sample Template    # JSON format to dict value

serializedJson = json.dumps(model_var)

input_var = { 'image': 'cbammultinode-IMAGE', 'ssh_key': 'cbammultinode', 'flavor': 'cbammultinode'}

template = Template(serializedJson)

converted_json = template.render(cbam=input_var)        # JSON String

print converted_json

==========

proper_format = json.loads(converted_json)  # Convert JSON string to Python Dict
```

```
with open("model.json", 'w') as data:
    json.dump(proper_format, data, indent=2, sort_keys=True)
```

How to Create class object in init function

31 July 2019 18:47

Class how we are going to initialise func

init func:

self.openstack=openstack_api.OpenstackAPI(self.ip, self.user)

Class methods:

```
def sum():
        self.openstack.createflavor()
```

CSV to JSON using Python

01 August 2019 16:33

Useful Links:

http://ayeshalshukri.co.uk/dev/python-script-to-extract-key-value-pairs-from-csv-file/

https://medium.com/@hannah15198/convert-csv-to-json-with-python-b8899c722f6d

CSV FILE:

UNDERCLOUD_IP,10.71.37.86
UNDERCLOUD_User,stack
UNDERCLOUD_Password,password
OVERCLOUD_VIP,10.71.37.87

Python Code:

```
import csv
import json

with open('cbam_Input_Sheet.csv') as data:
    csvReader = csv.reader(data)
    cbam_Input_details = {}

    for row in csvReader:      # row (first row) will give us value in list format ['UNDERCLOUD_IP', '10.71.37.86']
        key = row[0]
        value = row[1]

        cbam_Input_details[row[0]] = row[1]

        #cbam_Input_details[key] = value

#print key,value

print cbam_Input_details
```