



Python

Diego Pacheco

About Me



- ❑ Cat's Father
- ❑ Principal Software Architect
- ❑ Agile Coach
- ❑ SOA/Microservices Expert
- ❑ DevOps Practitioner
- ❑ Speaker
- ❑ Author



diegopacheco



@diego_pacheco



<http://diego-pacheco.blogspot.com.br/>



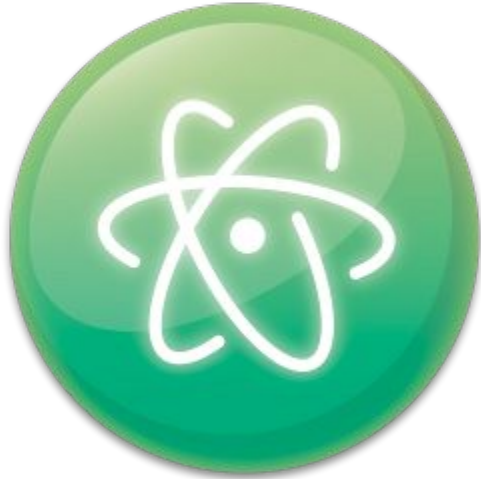
Python



Guido van Rossum

- ❑ Programming Language
- ❑ Comes with most of Linux
- ❑ Created in 1991
- ❑ OO language, Dynamic Typing, Interpreted
- ❑ Good for Scripts
- ❑ Easy to work with Process, Files, Json, Yaml.
- ❑ IMHO too flexible and can create bad code.
- ❑ Package Manager: PIP, i.e: `pip install PyGithub`
- ❑ Popular in DevOps and ML contexts.

Tooling



ATOM



VS CODE

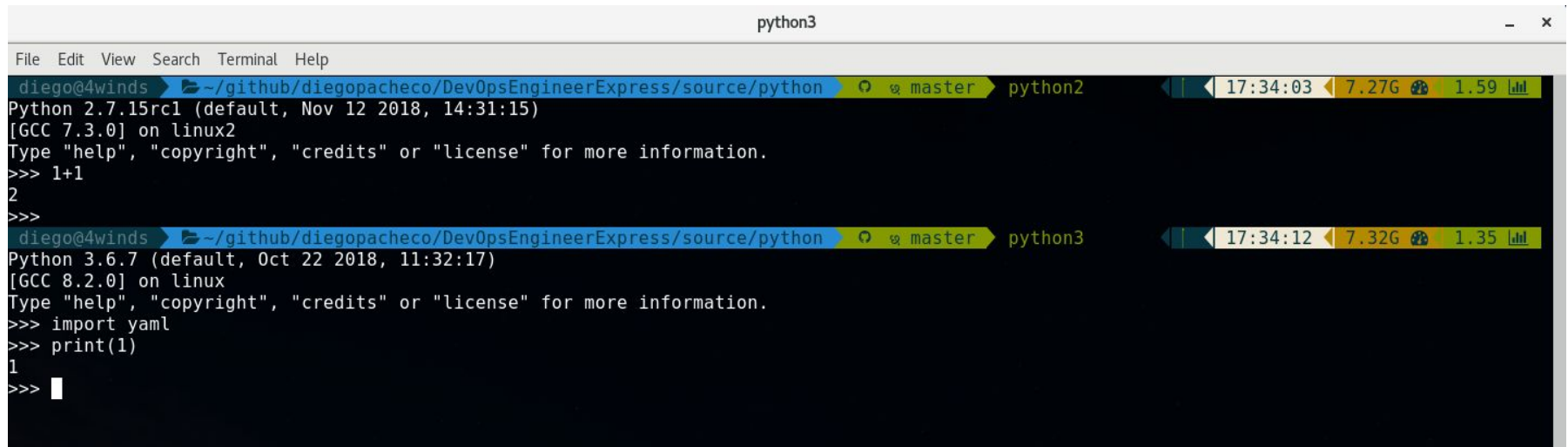


IntelliJ | Eclipse

Tabs Vs Spaces



Python REPL

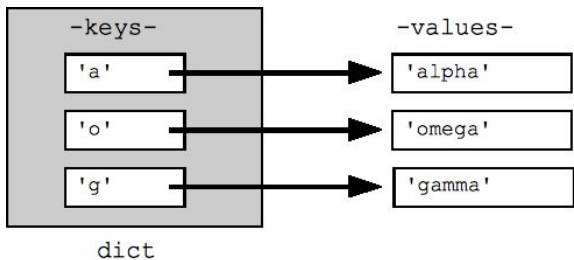


```
python3
File Edit View Search Terminal Help
diego@4winds > ~/github/diegopacheco/DevOpsEngineerExpress/source/python master python2 17:34:03 7.27G 1.59
Python 2.7.15rc1 (default, Nov 12 2018, 14:31:15)
[GCC 7.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> 1+1
2
>>>
diego@4winds > ~/github/diegopacheco/DevOpsEngineerExpress/source/python master python3 17:34:12 7.32G 1.35
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
[GCC 8.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import yaml
>>> print(1)
1
>>> 
```

Simple OOP In Python

```
oop.py x
1 class Math:
2     def __init__(self, x, y):
3         self.x = x
4         self.y = y
5
6     def add(self):
7         return self.x + self.y
8
9     def subtract(self):
10        return self.x - self.y
11
12    def multiply(self):
13        return self.x * self.y
14
15    def division(self):
16        return self.x / self.y
17
18 m = Math(2,3)
19 print(m.add())
20 print(m.multiply())
21 print(m.subtract())
22 print(m.division())
```

Dict Data Structure



```
dict-sample.py x
1
2 thisdict = {
3     "brand": "Ford",
4     "model": "Mustang",
5     "year": 1964
6 }
7
8 print(thisdict)
9
10 thisdict["year"] = 2018
11 print(thisdict["year"])
12
13 for k in thisdict:
14     print("Key: " + k)
15
16 for v in thisdict.values():
17     print("Value: " + str(v))
```


Sample Slack Msg



```
slack.py x
1 import urllib2
2 import json
3 import logging
4 import traceback
5 from contextlib import closing
6
7 logging.basicConfig()
8 log = logging.getLogger()
9 log.setLevel(logging.INFO)
10
11 SLACK_DEBUG_URL = 'https://hooks.slack.com/services/xxxx/yyyy/zzzz'
12 SLACK_DEBUG_CHANNEL = 'test_channel'
13 SLACK_AUTHOR = 'test_diego'
14 app_name = 'APP-Test'
15 TIMEOUT_PER_REQUEST = 5
16
17 slack_channel = SLACK_DEBUG_CHANNEL
18 message = "test_msg"
19 author_name = "test_author_diego"
20 slack_url = SLACK_DEBUG_URL
21
22 try:
23     slack_headers = {'Content-type': 'application/json', 'Accept': 'application/json', 'User-Agent': 'custom agent'}
24     color = "#f90c0c"
25     data = \
26     {
27         "channel": slack_channel,
28         "username": app_name,
29         "attachments":
30         [
31             {
32                 "author_name": 'Notification',
33                 "title": author_name,
34                 "color": color,
35                 "text": message,
36                 "mrkdwn_in": ["text", "pretext"]
37             }
38         ]
39     }
40     request = urllib2.Request(slack_url, json.dumps(data), headers=slack_headers)
41     with closing(urllib2.urlopen(request, timeout=TIMEOUT_PER_REQUEST)) as response:
42         the_page = response.read()
43         log.debug("Slack response: '%s'", the_page)
44 except:
45     log.error("Fail to send slack message, url: %s, channel: %s, message: %s\n\n%s", slack_url, slack_channel, message, traceback.format_exc())
46
```

HTTP Service With Flask



Flask

```
http-service.py x
1  #!/usr/bin/python
2
3  from flask import Flask
4  app = Flask(__name__)
5
6  @app.route("/healthchecker")
7  def healthchecker():
8      return "OK"
9
10 @app.route("/")
11 def hello():
12     return "Hello World!"
13
14 if __name__ == '__main__':
15     app.run(debug=True)
```

Using Github Api



github-sample.py x

```
1  #!/usr/bin/python3
2
3  from github import Github
4
5  g = Github("u", "p")
6
7  user = g.get_user("diegopacheco")
8  print(user.name)
9
10 for repo in g.get_user().get_repos():
11     print(repo.name)
```

Doing a Linux OS Call



```
os-sample.py x
1  import os
2
3  f = os.popen('hostname -i')
4  r = f.read()
5  print("Host: ", r)
6
7  f = os.popen('echo "2*3" | bc')
8  r = f.read()
9  print("2 x 3 == ", r)
```

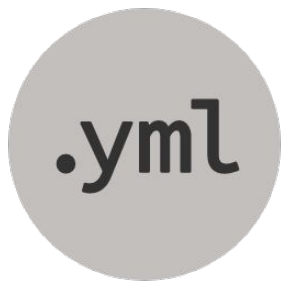
Working With Json



```
json-sample.py x
1 import json
2
3 j = json.loads('{"gravatai" : "1763", "porto_alegre" : "1769"}')
4
5 print("Was founded in: " +j['porto_alegre'])
```

```
json-sample-2.py x
1 import json
2
3 dict_obj = {
4     "name": "Diego Pacheco",
5     "age": 34,
6     "city": "Gravatai"
7 }
8
9 j = json.dumps(dict_obj)
10
11 print(j)
```

Working With Yaml



```
yaml-sample.py x
1  import yaml
2
3  with open("/home/diego/bin/apache-cassandra-3.9/conf/cassandra.yaml", 'r') as stream:
4      try:
5          y = yaml.load(stream)
6          print(y["seed_provider"][0]["parameters"][0])
7          print(y)
8      except yaml.YAMLError as exc:
9          print(exc)
```

```
yaml-sample-2.py x
1  import sys
2  from ruamel.yaml import YAML
3
4  string_yaml = """\
5  # example
6  name:
7      # details
8      family: Pacheco    # from somewhere
9      given: Diego      # me
10 """
11
12 yaml = YAML()
13 code = yaml.load(string_yaml)
14 code['name']['given'] = 'DIEGO'
15
16 yaml.dump(code, sys.stdout)
```

Working With Tar Gz

```
tar-sample.py x
1  import tarfile
2  import os
3
4  files = os.listdir(".")
5
6  tar = tarfile.open("source.tar.gz", "w:gz")
7  for name in files:
8      tar.add(name)
9  tar.close()
```





Python

Diego Pacheco