



Curso completo de Python!





Instrutor: Vitor Mazuco

<http://facebook.com/vitormazuco>

Email: vitor.mazuco@gmail.com

WebSite:

<http://vmzsolutions.com.br>



Módulos

Em nossos módulos, vamos começar pelo arquivo

ProvisionOps.py que controla a identificação no que precisa ser feito durante a instalação de um serviço. Ele vai receber um ID de um serviço, e pode ser para instalação ou remoção.



Módulos

Nesse arquivo, precisamos de um função chamado de InstallService e que foi chamado no arquivo DevOps.py para fazer a instalação do serviço.

```
def InstallService(self):
    service = ServiceClass()
    service._id = self._id

    sd = ServiceDao(service)
    service = sd.get()

    docker = DockerOps()
    res =
    docker.createContainer(service._id,service._product._image)
```



Módulos

Temos que também importar os módulos necessários para instalar esse serviço.

```
from Service import Service as ServiceClass  
from ServiceDao import ServiceDao  
from DockerOps import DockerOps
```



Módulos

No arquivo **DockerOps.py**, ele chamará o **SshOps.py**, já que não vimos o conceito de API's.

```
from SshOps import SshOps
```

Na criação, podemos criar nossos containers em Docker



Módulos

O método que iremos criar será a função `createContainer`, que deverá receber um parâmetro ID de um container e sua imagem.

```
def createContainer(self,id,image):  
    command = "docker run -d -ti --name %s %s"%( "%s_  
%s"%(image,id),'webservercloud')  
    ssh = SshOps()  
    res = ssh.runCommand(command)  
    if res['status'] == 1:  
        print res['message']
```



Módulos

O nosso módulo Docker já foi feito e ele usa o SSHOps.py, logo vamos criar esse módulo também.

```
from paramiko.client import SSHClient  
import paramiko
```

```
class SshOps:  
    def __init__(self):  
        pass
```




Módulos

No construtor Classe SSHOps, necessitamos criar a conexão com o servidor:

```
def __init__(self):  
    self.client = SSHClient()  
    self.client.load_system_host_keys()  
  
self.client.set_missing_host_key_policy(paramiko.AutoAddPolicy())  
self.client.connect("192.168.1.2")
```



Módulos

Agora que a conexão foi feita, necessitamos de um método `runCommand` que irá fazer os comandos de forma remota:

```
def runCommand(self,command):  
    stdin,stdout,stderr = self.client.exec_command(command)  
    if stderr.channel.recv_exit_status() != 0:  
        return {"status":1,"message":stderr.read()}  
    else:  
        return {"status":0,"message":stdout.read()}
```



Módulos

Para encerrar a construção dos módulos, necessitamos criar uma conexão com o MongoDB.

```
from pymongo import MongoClient
```

```
class MongoOps:
```

```
    def __init__(self):  
        self._client = MongoClient("127.0.0.1")  
        self._db = self._client["devops"]
```



Módulos

Vamos ver um método que será responsável por buscar os serviços que estão pendentes na instalação.

```
def getServiceToInstall(self):  
    return self._db.queue.find({"status":0})
```

Lembrado que os números representam:

0 – Pendente de Instalação

1 – Instalado

2 – Pendente de remoção