

Posgrado en Robótica e Inteligencia Artificial

Step-by-Step

Simulador EVOROBOTPY

<https://github.com/snolfi/evorobotpy/>

Material desenvolvido por

Prof. Eng. André Kelbouscas





Steps

1. Instalação Docker
2. Acesso ao container
3. Manipulação dos arquivos

Notas: Testado em uma VM com Ubuntu 16.04 LTS (70 GB de HD)

O caractere “\$” utilizado no começo de algumas partes deste material serve para referenciar que a partir desse caractere é representado um comando a ser executado no terminal, portanto não se deve colocar o “\$” no terminal, e sim apenas o restante.

Deve-se executar os comandos um a um. Assuma como cada novo caractere “\$”, um novo comando a ser executado separadamente.



Pré-Requisitos

Ambiente com Linux

Para a nossa aplicação, você vai precisar de uma versão com arquitetura de 64-bit de uma das versões do Ubuntu:

- Ubuntu Focal 20.04 (LTS)
- Ubuntu Eoan 19.10
- Ubuntu Bionic 18.04 (LTS)
- Ubuntu Xenial 16.04 (LTS)

Caso você utilize **Windows**, uma alternativa é a utilização de uma Máquina Virtual com uma das versões do Ubuntu acima listadas.

Step 1) Instalação Docker

(<https://docs.docker.com/engine/install/ubuntu/>)

```
$ sudo apt-get update
```

```
$ sudo apt-get install \
  apt-transport-https \
  ca-certificates \
  curl \
  gnupg-agent \
  software-properties-common
```

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

```
$ sudo apt-get update
```

```
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

"O pacote 'docker-ce' não tem candidato à instalação" ?

```
$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic test"
```

```
$ sudo apt update
```

```
$ sudo apt install docker-ce
```

"Não foi possível obter trava /var/lib/dpkg/lock-frontent" ?

```
$ sudo rm /var/lib/apt/lists/lock
```

```
$ sudo rm /var/lib/dpkg/lock
```

```
$ sudo rm /var/lib/dpkg/lock-frontent
```

```
$ sudo rm /var/cache/apt/archives/lock
```

Reconfigure os pacotes na sequência:

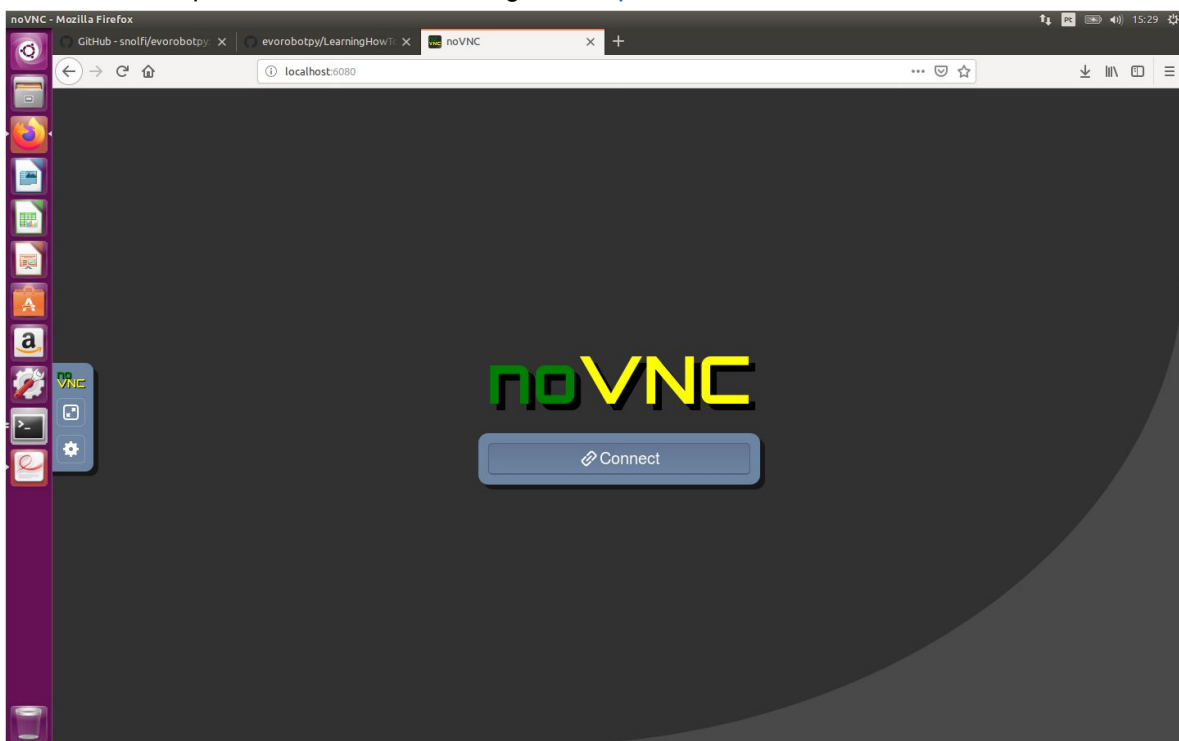
```
$ sudo dpkg --configure -a
```

Step 2) Acesso ao container

(<https://github.com/snolfi/evorobotpy/>)

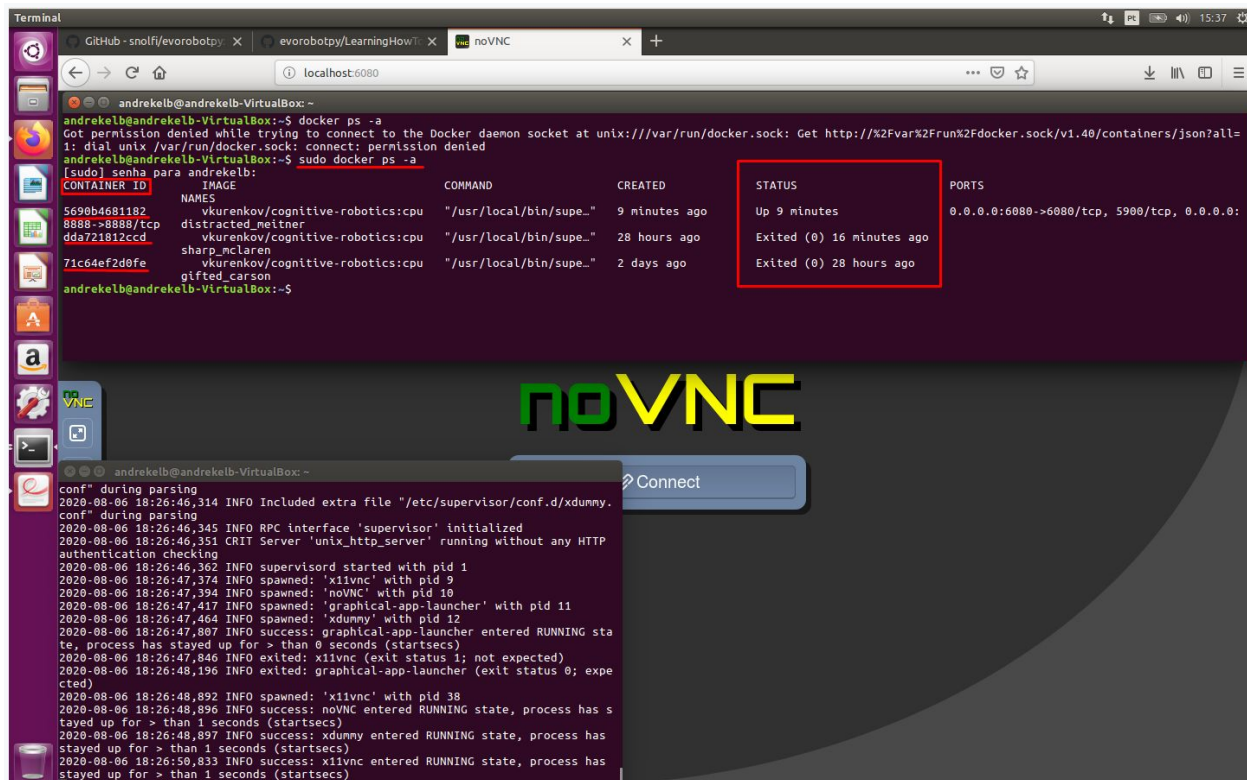
```
$ sudo docker pull vkurenkov/cognitive-robotics:cpu  
  
$ sudo docker run -it \  
-p 6080:6080 \  
-p 8888:8888 \  
--mount source=cognitive-robotics-opt-volume,target=/opt \  
vkurenkov/cognitive-robotics:cpu
```

Com isso, podemos acessar no navegador: <http://localhost:6080>



Abra um outro terminal e execute os comandos, este comando serve para verificar os containers em andamento e os IDs:

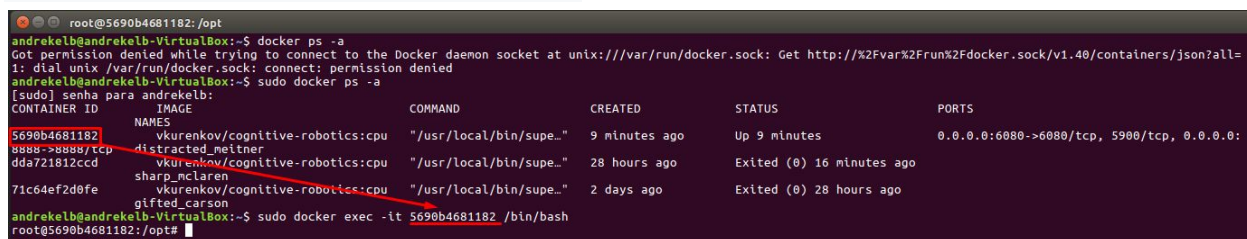
```
$ sudo docker ps -a
```



CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
5690b4681182	vkurenkov/cognitive-robotics:cpu	"/usr/local/bin/supe..."	9 minutes ago	Up 9 minutes	0.0.0.0:6080->6080/tcp, 5900/tcp, 0.0.0.0:
8888->8888/tcp	distracted_meltnet	"/usr/local/bin/supe..."	28 hours ago	Exited (0) 16 minutes ago	
dda721812ccd	vkurenkov/cognitive-robotics:cpu	"/usr/local/bin/supe..."	28 hours ago	Exited (0) 16 minutes ago	
71c64ef2d0fe	sharp_nclaren	"/usr/local/bin/supe..."	2 days ago	Exited (0) 28 hours ago	
	vkurenkov/cognitive-robotics:cpu	"/usr/local/bin/supe..."	2 days ago	Exited (0) 28 hours ago	
	gifted_carson	"/usr/local/bin/supe..."	2 days ago	Exited (0) 28 hours ago	

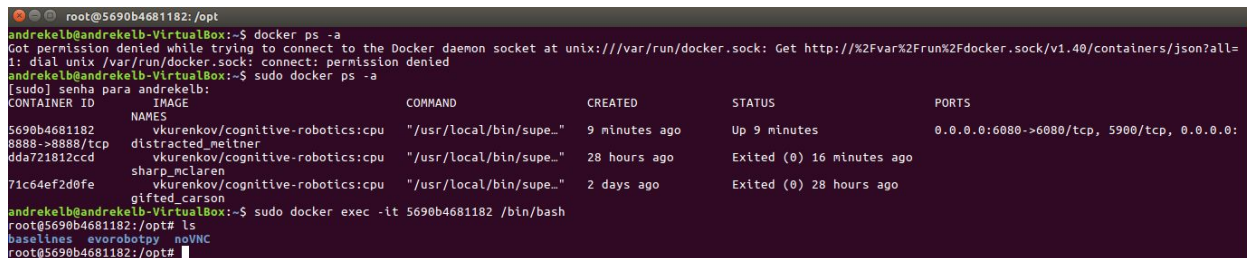
Por fim, fazemos o login no container que está em funcionamento, nesse caso o ID é 5690b4681182:

```
$ sudo docker exec -it <id> /bin/bash
```



```
root@5690b4681182:/opt#
```

Feito isso, podemos executar comandos padrões de manipulação de arquivos e pastas dentro do container que estamos, como: \$ cd (entrar em pasta) , ls (listas pastas e arquivos).



```
root@5690b4681182:/opt# ls
baselines  evorobotpy  noVNC
root@5690b4681182:/opt#
```

Agora podemos seguir os passos do [LearningHowToTrainRobots.pdf](https://github.com/snolfi/evorobotpy/blob/master/doc/LearningHowToTrainRobots.pdf) (<https://github.com/snolfi/evorobotpy/blob/master/doc/LearningHowToTrainRobots.pdf>)

Alguns exemplos da utilização do simulador EVOROBOTPY:

