

Manual de Instalação - Exercício de Programa de Sistemas Distribuídos

Gustavo Henriques Vieira

Sumário

Instalação	2
Configuração	2

Nota importante: Os testes foram executados em ambientes Linux, mas este projeto pode ser executado em máquinas Windows, macOS ou Linux, com ou sem Docker. Abaixo estão as instruções para configurar o ambiente em cada cenário.

Instalação

Instalando pelo Linux

```
sudo apt update && sudo apt upgrade -y
sudo apt install -y python3 python3-pip python3-env
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
sudo curl -L \
  "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" \
  -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
```

- **(Opcional):** Adicionando o usuário ao grupo do Docker para não precisar usar `sudo`

```
sudo usermod -aG docker $USER
newgrp docker
```

Instalando pelo Windows

1. Instalando o Python:

- Acesse o site [!\(https://www.python.org/downloads/\)](https://www.python.org/downloads/)[\[https://www.python.org/downloads/\]](https://www.python.org/downloads/) e siga com a instalação
- Marque a opção “Add Python to PATH” na instalação.

2. Instalando o Docker Desktop:

- Acesse o site [!\(https://docs.docker.com/desktop/setup/install/windows-install/\)](https://docs.docker.com/desktop/setup/install/windows-install/)[\[https://docs.docker.com/desktop/setup/install/\]](https://docs.docker.com/desktop/setup/install/windows-install/) e siga com a instalação
- Habilite a opção “Instalar Docker Compose”.

Instalando pelo macOSwire

1. Instalando o Homebrew (Caso ainda não tenha)

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

2. Instalando Python e Docker

```
brew install python
brew install --cask docker
```

Configuração

Usando Docker

1. Obtenha o endereço IP da máquina

```
ifconfig # Linux/macOS
ipconfig # Windows
```

2. Construa a imagem do servidor

```
# ex: benchmark_servers (Recomendado: "gustavohnsv/benchmark_servers:latest")
docker build -f Dockerfile.servers -t <image_name> .
```

3. Execute o servidor Docker

```
docker run -p 50050:50050 -p 50051:50051 <image_name>
```

- **(Opcional):** Caso tenha Docker Compose instalado:

```
docker-compose up --no-build # caso já tenha construído a imagem anteriormente
docker-compose up --build    # caso ainda não tenha construído a imagem anteriormente
```

Nome das imagens usando Docker Compose: Caso tenha construído a imagem a partir do Docker Compose, o nome será diferente

Usando Docker para o cliente (NÃO RECOMENDADO)

1. Construa as imagens dos clientes

```
# ex: benchmark_servers (Recomendado: "gustavohnsv/grpc_client:latest")
docker build -f Dockerfile.grpc-client -t <image_name> .
```

```
# ex: benchmark_servers (Recomendado: "gustavohnsv/jsonrpc_client:latest")
docker build -f Dockerfile.jsonrpc-client -t <image_name> .
```

2. Execute o cliente Docker

```
# segue a mesma lógica das variáveis ao usar Python
docker run -e SERVER_IP=X.X.X.X -e BENCHMARKING=X <image_name>
```

Variáveis de ambiente: SERVER_IP serve para realizar conexão com o servidor, enquanto BENCHMARKING é booleano e serve para ativar a coleta de dados, caso contrário, o cliente só envia chamadas

Sem usar Docker

1. Na raiz do projeto

```
python3 -m venv <env_name> # ex: env
source <env_name>/bin/activate # Linux/macOS
<env_name>\Scripts\activate.bat # Windows (via CMD)
<env_name>\Scripts\Activate.ps1 # Windows (via Powershell)
pip install -r requirements.txt
```

Nota: Caso queira utilizar Python nativamente em ambas as máquinas, deve executar o comando em ambas. Caso contrário, apenas na máquina cliente.

2. Obtenha o endereço IP da máquina

```
ifconfig # Linux/macOS
ipconfig # Windows
```

3. Executando o servidor

```
cd src/rpc-grpc # ou '/rpc-jsonrpc'
python server.py
```

4. Executando o cliente

```
cd src/rpc-grpc # ou '/rpc-jsonrpc'
```

```
# ex: SERVER_IP=192.168.68.100 BENCHMARKING=False (IP deve ser o mesmo da máquina servidor)
SERVER_IP=X.X.X.X BENCHMARKING=X python client.py
```