Principais dúvidas

git, docker e outras coisinhas

boss

Quem vai falar hoje?

Bruna M.

Como pego as atualizações do repositório da boss?

Depois de *forkar*, lá no github, você deu **git clone**.

```
$  git remote -v
origin git@github.com:BrunaNayara/rasa-ptbr-boilerplate (fetch)
origin git@github.com:BrunaNayara/rasa-ptbr-boilerplate (push)
```

Agora o seu computador só está linkado com o seu repositório remoto. Não está linkado com o da boss.

Depois de *forkar*, lá no github, você deu **git clone**.

```
$  git remote -v
origin git@github.com:BrunaNayara/rasa-ptbr-boilerplate (fetch)
origin git@github.com:BrunaNayara/rasa-ptbr-boilerplate (push)
```

Agora o seu computador só está linkado com o seu repositório remoto. Não está linkado com o da boss.

Precisamos adicionar outro remoto

Depois de *forkar*, lá no github, você deu **git clone**.

```
$  git remote -v
origin git@github.com:BrunaNayara/rasa-ptbr-boilerplate (fetch)
origin git@github.com:BrunaNayara/rasa-ptbr-boilerplate (push)
```

Agora o seu computador só está linkado com o seu repositório remoto. Não está linkado com o da boss.

Precisamos adicionar outro remoto

```
$ git remote add nome-repo link-clone-github
```

Depois de *forkar*, lá no github, você deu **git clone**.

```
$  git remote -v
origin git@github.com:BrunaNayara/rasa-ptbr-boilerplate (fetch)
origin git@github.com:BrunaNayara/rasa-ptbr-boilerplate (push)
```

Agora o seu computador só está linkado com o seu repositório remoto. Não está linkado com o da boss.

Precisamos adicionar outro remoto

```
$ git remote add nome-repo link-clone-github
```

nome-repo é o apelido para o outro remoto. Você pode escolher

link-clone-gihtub é o link que você pega no botão no github

```
brunamoreira@spf1lt-pj000985:~/Projects/off_5a/rasa-ptbr-boilerplate (master-boss)$ git remote -v
boss    git@github.com:BOSS-BigOpenSourceSister/rasa-ptbr-boilerplate.git (fetch)
boss    git@github.com:BOSS-BigOpenSourceSister/rasa-ptbr-boilerplate.git (push)
lappis  git@github.com:lappis-unb/rasa-ptbr-boilerplate.git (fetch)
lappis  git@github.com:lappis-unb/rasa-ptbr-boilerplate.git (push)
personal        git@github.com:BrunaNayara/rasa-ptbr-boilerplate.git (fetch)
personal        git@github.com:BrunaNayara/rasa-ptbr-boilerplate.git (push)
```

Nesse exemplo, tem três remotos linkados.

- boss: linkado com o repositório da BOSS
- lappis: linkado com o repositório original do projeto
- personal: linkado com o meu repositório

```
$ git fetch boss
```

O comando **git fetch** pega as atualizações do repositório

```
$ git pull boss master
```

O comando **git pull** atualiza a branch

```
$ git checkout -b nova-branch
```

Agora a sua **nova-branch** está atualizada com as últimas alterações do repositório da BOSS.

Quais são os principais comandos para desenvolver o bot?

```
$ make first-run
```

O comando **make first-run** prepara os containers para trabalhar no bot.

O ideal é rodar apenas a primeira vez.

```
$ make run-shell
```

O comando **make run-shell** roda o bot no terminal, junto com informações de debug.

Toda vez que quiser conversar com o bot.

```
$ make train
```

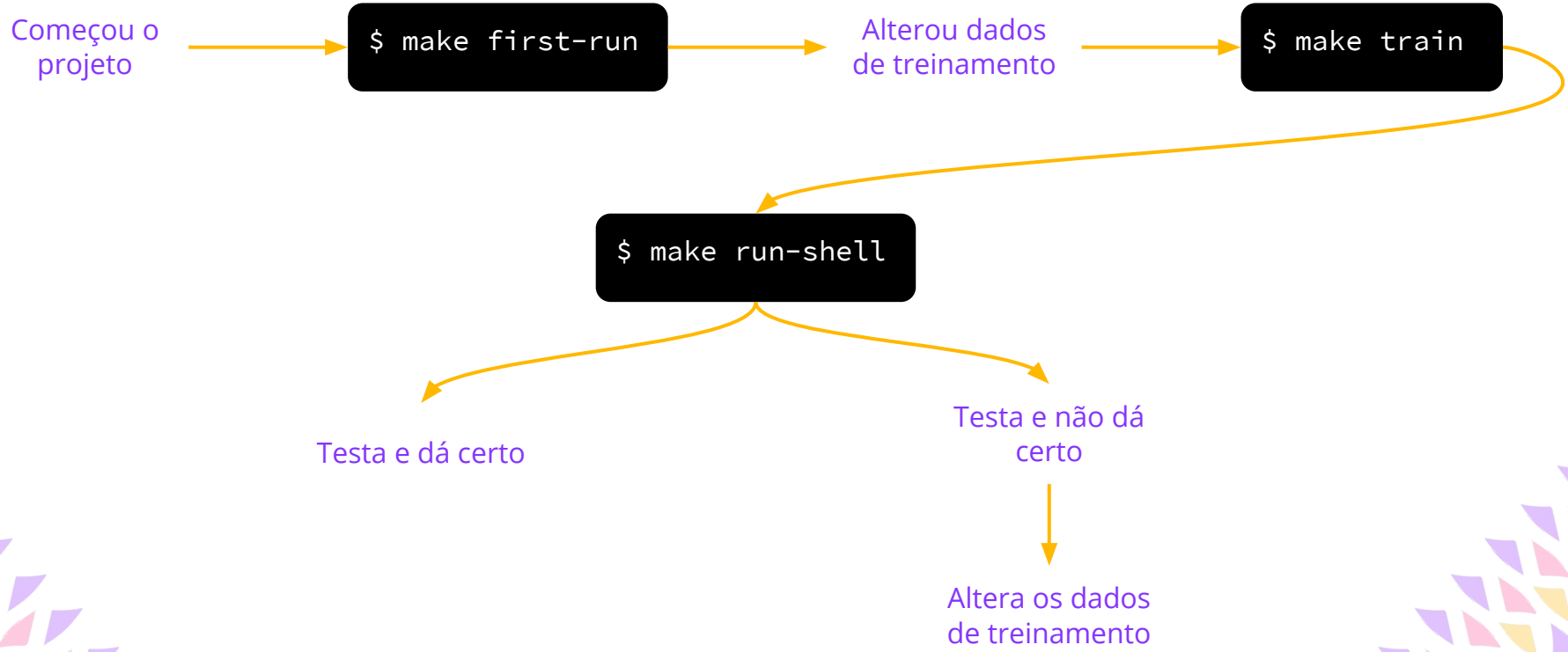
O comando **make train** treina o modelo de entendimento do bot.

Deve ser executado a cada alteração dos dados de treinamento

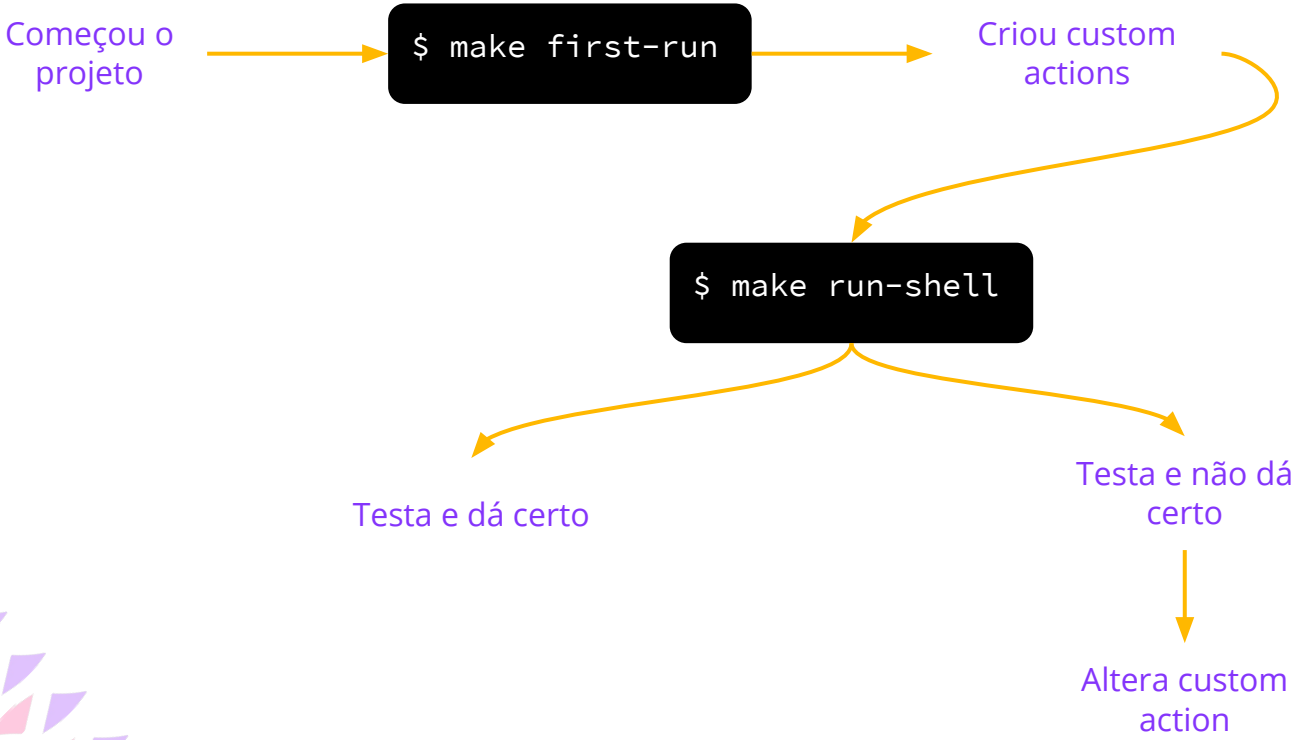
Quer achar mais comandos?

O arquivo **Makefile** contém os comandos que já estão configurados para o boilerplate

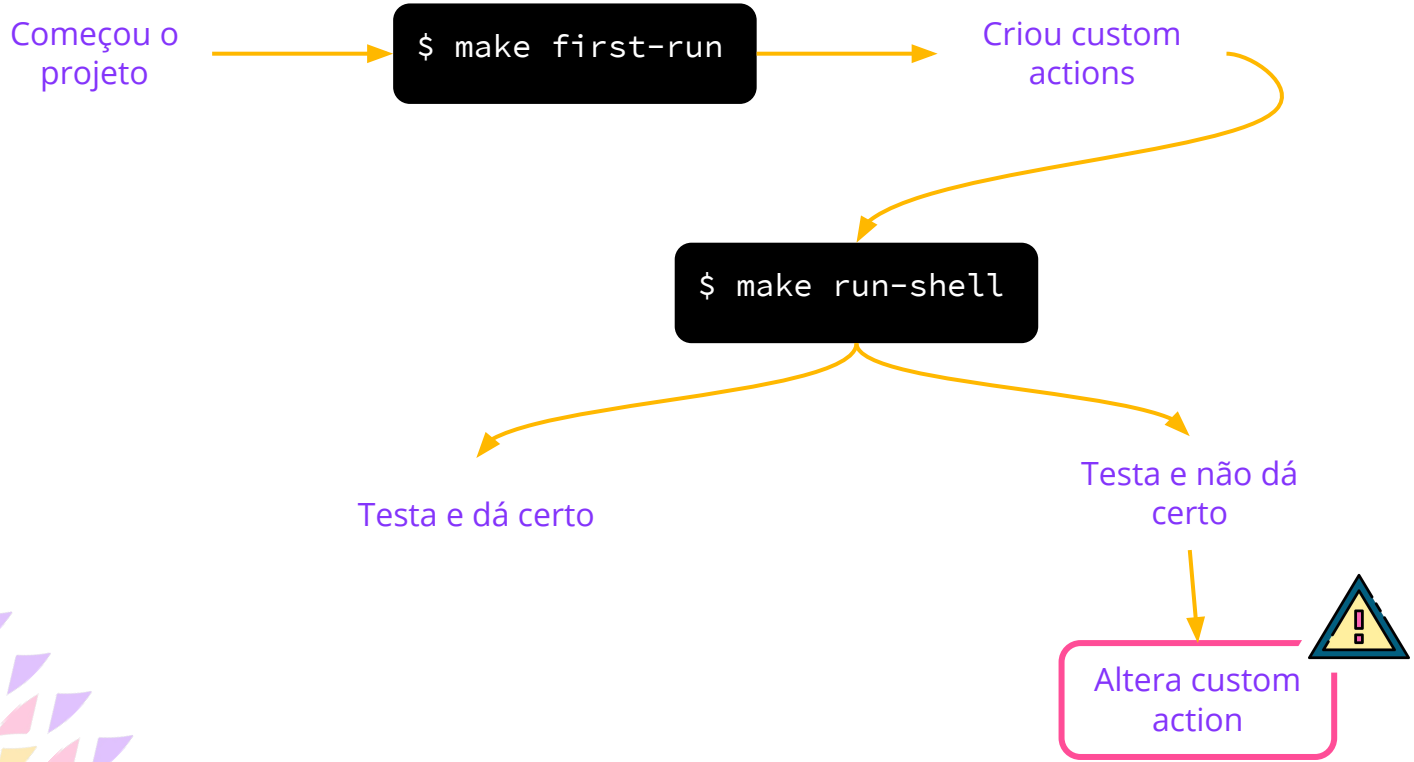
Trabalhar com o conteúdo do bot



Trabalhar com custom action



Trabalhar com custom action



Alterei as custom actions e elas continuam com erro. E agora?

Atualmente dois comandos sobem o container das **custom actions**

```
$ make run-actions
```

```
$ make run-shell
```

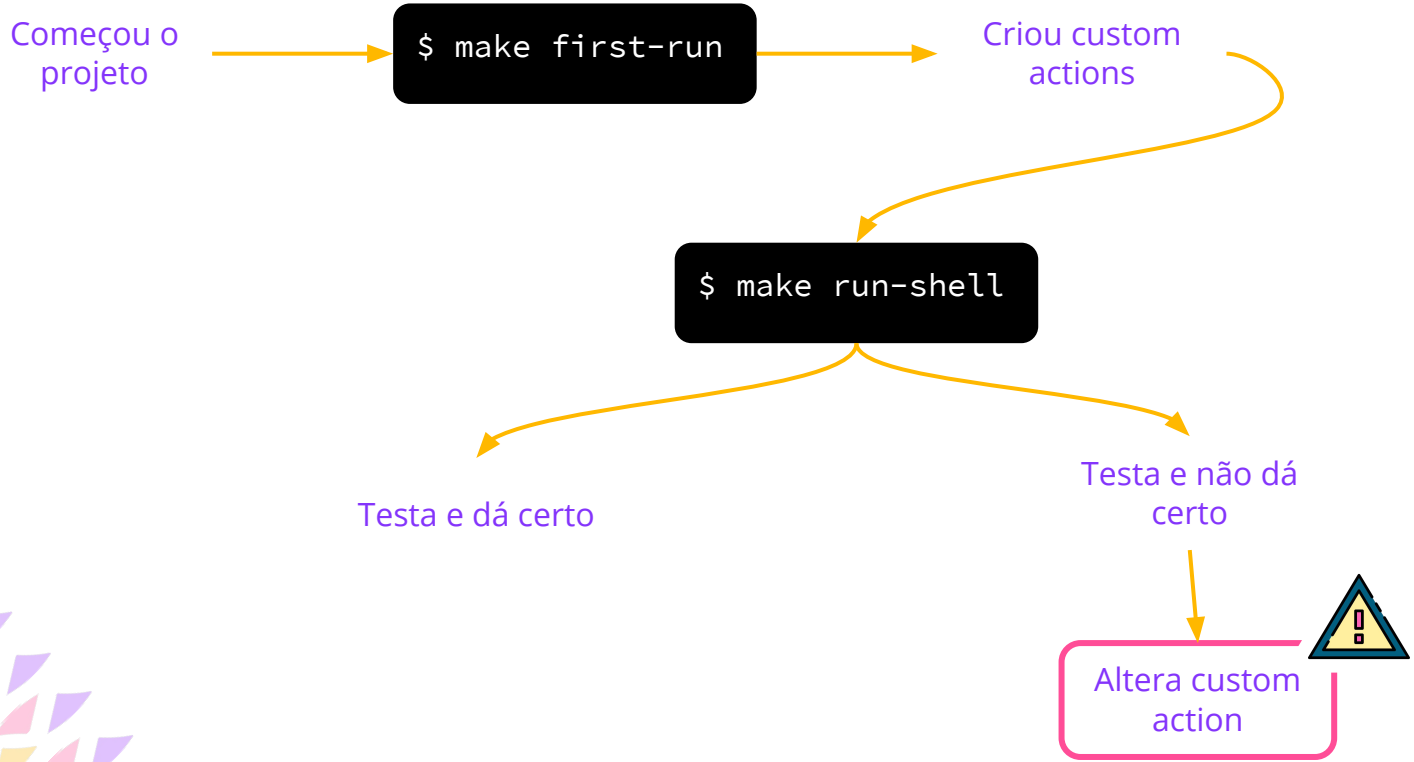
Mas porque?

Para conversar no terminal, precisamos das actions rodando.

Então, quando executamos o comando de conversar pelo terminal, ela já sobe as actions.

Porém, se o container das actions, já estiver rodando, ele não atualiza.

Trabalhar com custom action



Podemos ver quando o container de custom actions foi criado com o comando **docker ps**

```
$ docker ps
CONTAINER ID   IMAGE                  CREATED         STATUS          NAMES
123ab          rasa...plate_actions   10 days ago     Up 20 minutes   rasa-ptbr-boilerplate_actions_1
6ce08          rasa...plate_bot       2 seconds ago   Up 1 second     rasa-ptbr-boilerplate_bot_run
```

E agora?

Precisamos parar e remover o container que estava com o código antigo das custom actions.

Custom actions com código atualizado

```
$ docker stop 123ab
123ab
```

Usamos o comando **docker stop <container-id>** para pararmos o container com o código antigo das custom actions

Custom actions com código atualizado

```
$ docker stop 123ab
123ab

$ docker ps -a
CONTAINER ID   CREATED       STATUS                   NAMES
123ab          10 days ago   Exited(0) 2 hours ago    rasa-ptbr-boilerplate_actions_1
```

Podemos ver com o comando **docker ps -a** que o container ainda é o que foi criado há dias. Então está desatualizado.

Custom actions com código atualizado

```
$ docker stop 123ab
123ab

$ docker ps -a
CONTAINER ID   CREATED       STATUS                   NAMES
123ab          10 days ago   Exited(0) 2 hours ago    rasa-ptbr-boilerplate_actions_1

$ docker rm 123ab
123ab
```

Usamos o comando **docker rm <container-id>** para remover totalmente esse container.

Custom actions com código atualizado

```
$ docker stop 123ab
123ab

$ docker ps -a
CONTAINER ID   CREATED       STATUS                   NAMES
123ab          10 days ago   Exited(0) 2 hours ago    rasa-ptbr-boilerplate_actions_1

$ docker rm 123ab
123ab

$ docker ps -a
CONTAINER ID   CREATED       STATUS                   NAMES
```

Agora podemos ver que não tem mais o container com código antigo. Podemos subir o container e ele vai criar atualizado.

Minhas custom actions estão dando errado e não sei o porquê.

Logs

```
brunamoreira@spf1lt-pj000985:~/Projects/off_5a/process-crawler-api (master)$ flask run
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
710802-55.2018.8.02.0001
127.0.0.1 - - [13/Oct/2020 10:34:34] "GET /processo/710802-55.2018.8.02.0001 HTTP/1.1" 200 -
806233-85.2019.8.02.0000
127.0.0.1 - - [13/Oct/2020 10:34:52] "GET /processo/806233-85.2019.8.02.0000 HTTP/1.1" 200 -
0821901-51.2018.8.12.0001
TJMS
website:https://esaj.tjms.jus.br/cposg5/search.do?conversationId=&paginaConsulta=0&cbPesquisa=NU
ificado=0001&dePesquisaNuUnificado=0821901-51.2018.8.12.0001&dePesquisaNuUnificado=UNIFICADO&deP
127.0.0.1 - - [13/Oct/2020 10:35:03] "GET /processo/0821901-51.2018.8.12.0001 HTTP/1.1" 200 -
127.0.0.1 - - [13/Oct/2020 10:35:16] "GET / HTTP/1.1" 200 -
0821901-51.2018.8.12.0001
TJMS
website:https://esaj.tjms.jus.br/cposg5/search.do?conversationId=&paginaConsulta=0&cbPesquisa=NU
ificado=0001&dePesquisaNuUnificado=0821901-51.2018.8.12.0001&dePesquisaNuUnificado=UNIFICADO&deP
127.0.0.1 - - [13/Oct/2020 10:35:43] "GET /processo/0821901-51.2018.8.12.0001 HTTP/1.1" 200 -
```

Exemplo de log de uma aplicação (Python + Flask)

Usamos logs para ver o que aconteceu na aplicação.

Podemos ver os IDs dos container com o comando **docker ps**

```
$ docker ps
CONTAINER ID   IMAGE                  CREATED         STATUS          NAMES
123ab          rasa...plate_actions   10 days ago     Up 20 minutes   rasa-ptbr-boilerplate_actions_1
6ce08          rasa...plate_bot       2 seconds ago   Up 1 second     rasa-ptbr-boilerplate_bot_run
```

Como vejo os logs?

Precisamos saber o ID do container pra ver os logs

```
$ docker logs -f <container-id>
```

O comando **docker logs** mostra os logs das coisas que aconteceram no container.

A flag **-f** (follow) fica acompanhando os logs

```
sudo docker logs -f 4ea3e1db375e

PostgreSQL Database directory appears to contain a database; Skipping initialization

2020-10-13 11:21:12.595 -03 [1] LOG:  starting PostgreSQL 12.4 on x86_64-pc-linux-musl, compiled by
2020-10-13 11:21:12.595 -03 [1] LOG:  listening on IPv4 address "0.0.0.0", port 5432
2020-10-13 11:21:12.595 -03 [1] LOG:  listening on IPv6 address "::", port 5432
2020-10-13 11:21:12.599 -03 [1] LOG:  listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
2020-10-13 11:21:12.619 -03 [20] LOG:  database system was shut down at 2020-10-11 18:22:50 -03
2020-10-13 11:21:12.625 -03 [1] LOG:  database system is ready to accept connections
2020-10-13 11:21:19.838 -03 [1] LOG:  received fast shutdown request
2020-10-13 11:21:19.840 -03 [1] LOG:  aborting any active transactions
2020-10-13 11:21:19.841 -03 [1] LOG:  background worker "logical replication launcher" (PID 26) exi
2020-10-13 11:21:19.841 -03 [21] LOG:  shutting down
2020-10-13 11:21:19.858 -03 [1] LOG:  database system is shut down

PostgreSQL Database directory appears to contain a database; Skipping initialization

2020-10-13 11:21:25.091 -03 [1] LOG:  starting PostgreSQL 12.4 on x86_64-pc-linux-musl, compiled by
2020-10-13 11:21:25.092 -03 [1] LOG:  listening on IPv4 address "0.0.0.0", port 5432
2020-10-13 11:21:25.092 -03 [1] LOG:  listening on IPv6 address "::", port 5432
2020-10-13 11:21:25.097 -03 [1] LOG:  listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
2020-10-13 11:21:25.117 -03 [20] LOG:  database system was shut down at 2020-10-13 11:21:19 -03
2020-10-13 11:21:25.123 -03 [1] LOG:  database system is ready to accept connections
```

Screenshot dos logs de um container.

Comandos úteis de docker

```
$ docker ps
```

Para ver os comandos que estão rodando, basta usar **docker ps** . Usando a flag **-a** vê os que estão preparados, mas não estão rodando.

```
$ docker logs -f <container-id>
```

Para ver os logs do container, basta usar o **docker logs**. Para

```
$ docker stop <container-id>
```

Para parar a execução de um container, use o comando **docker stop**.

Lembre-se, ele não deixa de existir.

```
$ docker rm <container-id>
```

Para remover um container, é só usar o comando **docker rm**.

Ele só remove containers parados. Não remove se estiver em execução.

Obrigada!

Todas que contribuíram com dúvidas e fizeram essa apresentação possível

brunanayaramlima@gmail.com

https://github.com/orgs/BOSS-BigOpenSourceSister

Licença

Estes slides são concedidos sob uma Licença Creative Commons. Sob as seguintes condições: **Atribuição, Uso Não-Comercial e Compartilhamento pela mesma Licença**.

Mais detalhes sobre essa licença em: creativecommons.org/licenses/by-nc-sa/3.0/

Cores

Roxo: 9730ff

Rosa: ff4d97

Amarelo: ffb800


boss
boss

Todos os elementos usados