

02



**Getting started
+ Docker**



AOBA!

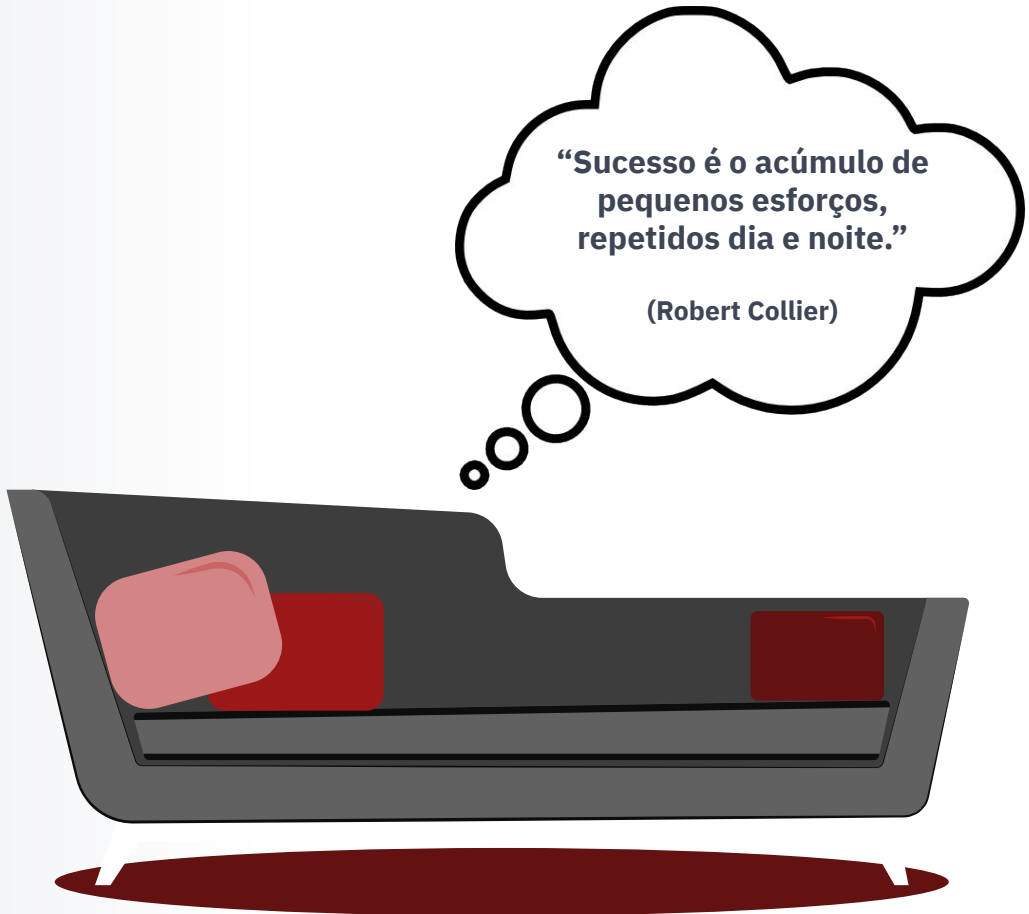
Ricardo de Luna Galdino
Software Engineer



LINKEDIN:
[linkedin.com/in/ricardo-galdino](https://www.linkedin.com/in/ricardo-galdino)

GITHUB:
github.com/ricardogaldino

WHATSAPP GROUP:
[engsoft.org](https://www.whatsapp.com/channel/0029va833333333333333333)



**“Sucesso é o acúmulo de
pequenos esforços,
repetidos dia e noite.”**

(Robert Collier)

Cultura ágil de aprendizado



Microlearning:

- É uma metodologia de ensino que subdivide um assunto em doses menores de conteúdo, com atividades rápidas, auxiliando na compreensão e retenção deste conteúdo.

Pílulas do Conhecimento:

- São pequenos conteúdos apresentados ao profissional para que consiga assimilar de forma mais focada e objetiva, melhorando a eficiência e potencializando os resultados obtidos.
- Microlearning é composto por diversas Pílulas do Conhecimento.

Glossário

Vamos rever alguns conceitos e termos...

Docker

- ▶ Docker é uma tecnologia de **virtualização** em nível de sistema operacional para entregar software em pacotes chamados containers.
- ▶ Os **containers** são isolados uns dos outros e **agrupam seus próprios softwares, bibliotecas e arquivos de configuração**.
- ▶ Eles podem se comunicar uns com os outros por meio de canais bem definidos.
- ▶ Todos os containers são executados por **um único kernel do sistema operacional** e, portanto, usam menos recursos do que as máquinas virtuais.

Overview:

<https://docs.docker.com/get-started/overview/>

Instalação:

<https://docs.docker.com/engine/install/>

Docker Compose

- ▶ Docker Compose é uma ferramenta para a **criação e execução de múltiplos containers** de aplicação **ao mesmo tempo**.
- ▶ Ele utiliza um **arquivo** do tipo “**yaml**” para **definir** como será o **ambiente** em execução e usando um único comando criará e **iniciará todos os containers** de aplicação definidos.

Overview:

<https://docs.docker.com/compose/>

Instalação:

<https://docs.docker.com/compose/install/>

Comandos do Docker Compose

- ▶ O Docker Compose possui alguns **comandos** a serem utilizados para o provisionamento e **gerenciamento dos contêineres**, vejamos então os principais:
 - ▷ `docker-compose ps`: lista os contêineres;
 - ▷ **docker-compose up**: cria e inicia os contêineres;
 - ▷ **docker-compose down**: paralisa e remove todos os contêineres e seus componentes como rede, imagem e volume.
 - ▷ `docker-compose build`: realiza apenas a etapa de build das imagens que serão utilizadas;
 - ▷ `docker-compose logs`: visualiza os logs dos contêineres;
 - ▷ `docker-compose restart`: reinicia os contêineres;
 - ▷ `docker-compose start`: inicia os contêineres;
 - ▷ `docker-compose stop`: paralisa os contêineres;
- ▶ **Overview:**
<https://docs.docker.com/compose/reference/>

RabbitMQ (YAML)

Docker Compose

Docker Compose (YAML) para RabbitMQ

<https://github.com/ricardogaldino/microlearning-rabbitmq/blob/main/docker/docker-compose.yml>

Estrutura do arquivo

- ▶ Como já dito, para executar nossos containers com o Docker Compose é necessário possuir um arquivo **YAML** que contenha todas as informações e parâmetros necessários para sua execução.
- ▶ Por padrão, os comandos docker-compose procuram um arquivo no diretório corrente nomeado como **docker-compose.yml**, porém é possível **indicar** um outro **nome de arquivo e também** em um outro **local** passando o **parâmetro -f**.
- ▶ Na figura podemos observar um exemplo de um arquivo YAML que contém a estrutura dos principais **parâmetros** utilizados para a execução do **contêiner RabbitMQ** via Docker Compose.

```
1  version: "3.2"
2
3  services:
4
5    rabbitmq:
6      image: "rabbitmq:3.9.12-management"
7      container_name: "rabbitmq"
8      hostname: "rabbitmq"
9      ports:
10         - 5672:5672
11         - 15672:15672
12      volumes:
13         - ./config/rabbitmq.conf:/etc/rabbitmq/rabbitmq.config
14      networks:
15         - rabbitmq_net
16
17  networks:
18    rabbitmq_net:
19      driver: bridge
```

Iniciando o RabbitMQ via Docker Compose

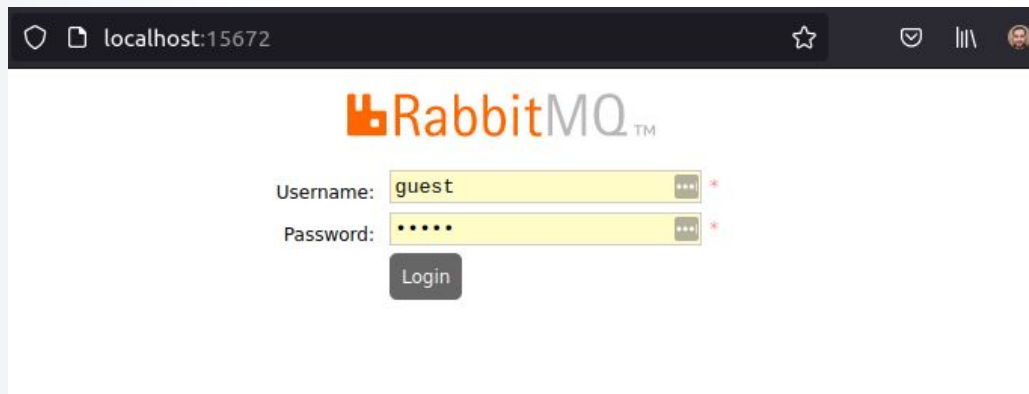
- ▶ Faça o **download** do projeto no **GitHub**:
<https://github.com/ricardogaldino/microlearning-rabbitmq/>
- ▶ Abra o **terminal** e navegue até a pasta **"/docker"**
- ▶ Execute o **comando**:

```
$ docker-compose -f docker-compose.yml up
```

```
rabbitmq | 2022-01-21 12:40:27.931668+00:00 [info] <0.222.0>
rabbitmq | 2022-01-21 12:40:27.931668+00:00 [info] <0.222.0> Starting RabbitMQ 3.9.12 on Erlang 24.2 [jit]
rabbitmq | 2022-01-21 12:40:27.931668+00:00 [info] <0.222.0> Copyright (c) 2007-2022 VMware, Inc. or its affiliates.
rabbitmq | 2022-01-21 12:40:27.931668+00:00 [info] <0.222.0> Licensed under the MPL 2.0. Website: https://rabbitmq.com
rabbitmq |
rabbitmq | ## ## RabbitMQ 3.9.12
rabbitmq | ## ##
rabbitmq | ##### Copyright (c) 2007-2022 VMware, Inc. or its affiliates.
rabbitmq | #####
rabbitmq | ##### Licensed under the MPL 2.0. Website: https://rabbitmq.com
rabbitmq |
rabbitmq | Erlang: 24.2 [jit]
rabbitmq | TLS Library: OpenSSL - OpenSSL 1.1.1m 14 Dec 2021
rabbitmq |
rabbitmq | Doc guides: https://rabbitmq.com/documentation.html
rabbitmq | Support: https://rabbitmq.com/contact.html
rabbitmq | Tutorials: https://rabbitmq.com/getstarted.html
rabbitmq | Monitoring: https://rabbitmq.com/monitoring.html
rabbitmq |
rabbitmq | Logs: /var/log/rabbitmq/rabbit@rabbitmq_upgrade.log
rabbitmq | <stdout>
rabbitmq |
rabbitmq | Config file(s): /etc/rabbitmq/conf.d/10-default-guest-user.conf
rabbitmq |
rabbitmq | Starting broker...2022-01-21 12:40:27.932394+00:00 [info] <0.222.0>
rabbitmq | 2022-01-21 12:40:27.932394+00:00 [info] <0.222.0> node : rabbit@rabbitmq
rabbitmq | 2022-01-21 12:40:27.932394+00:00 [info] <0.222.0> home dir : /var/lib/rabbitmq
rabbitmq | 2022-01-21 12:40:27.932394+00:00 [info] <0.222.0> config file(s) : /etc/rabbitmq/conf.d/10-default-guest-user.conf
rabbitmq | 2022-01-21 12:40:27.932394+00:00 [info] <0.222.0> cookie hash : jevuNPhWuNmZEscpPUB+Fg==
```

Acessando o RabbitMQ pelo Navegador

- ▶ Com o **RabbitMQ inicializado** vamos **acessar** a sua **Interface Administradora** pelo navegador Web.
- ▶ Abra seu **browser** de preferência e acesse o link padrão:
<http://localhost:15672/>
- ▶ Digite o usuário e senha padrão para se logar:
 - ▶ **Usuário:** “guest”
 - ▶ **Senha:** “guest”



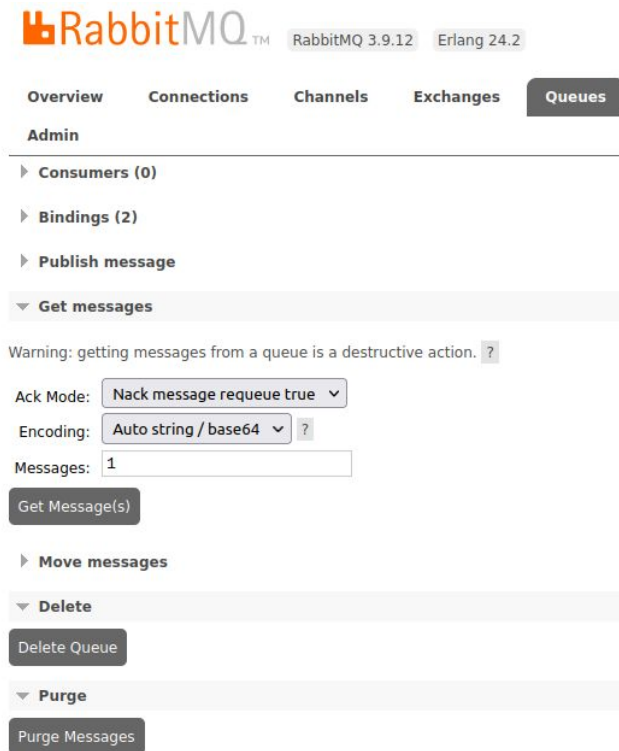
Interface do RabbitMQ

<https://www.rabbitmq.com/management.html>

- ▶ Estando **logado**, podemos utilizar a **interface** de usuário para **gerenciamento e monitoramento de** mensagens, queues, exchanges, bindings e demais **entidades** AMQP.

Sendo possível:

- ▶ **Adicionar, editar e deletar** queues, exchanges, bindings, etc...
- ▶ **Publicar, consultar e expurgar** mensagens.



The screenshot displays the RabbitMQ Management Interface. At the top, the RabbitMQ logo is followed by version information: RabbitMQ 3.9.12 and Erlang 24.2. Below this is a navigation bar with tabs for Overview, Connections, Channels, Exchanges, and Queues (which is selected). Under the Queues tab, there is an 'Admin' section with links for Consumers (0), Bindings (2), and Publish message. A 'Get messages' section is expanded, showing a warning: 'Warning: getting messages from a queue is a destructive action.' Below the warning are dropdown menus for 'Ack Mode' (set to 'Nack message requeue true') and 'Encoding' (set to 'Auto string / base64'). There is a text input for 'Messages' with the value '1'. A 'Get Message(s)' button is present. Below this is a 'Move messages' section. A 'Delete' section is expanded, showing a 'Delete Queue' button. A 'Purge' section is also expanded, showing a 'Purge Messages' button.

TO BE CONTINUED...

Getting started
RabbitMQ + Spring + Kotlin

<https://github.com/ricardogaldino/microlearning-rabbitmq/blob/main/docs/microlearning-rabbitmq-003-spring-kotlin.pdf>

