

FIAP GRADUAÇÃO

TECNOLOGIA EM DESENVOLVIMENTO DE SISTEMAS

DevOps Tools & Cloud Computing

Exercícios de Dockerfile

PROF. João Menk	profjoao.menk@fiap.com.br
PROF. Sálvio Padlipskas	salvio@fiap.com.br
PROF. Antonio Figueiredo	profantonio.figueiredo@fiap.com.br
PROF. Marcus Leite	profmarcus.leite@fiap.com.br
PROF. Thiago Rocha	profthiago.rocha@fiap.com.br
PROF. Thiago Moraes	proftiago.moraes@fiap.com.br

Este exercício tem como objetivo praticar a criação de um Dockerfile para rodar um aplicativo em Java Spring Boot

O Docker é uma ferramenta que permite empacotar uma aplicação junto com suas dependências e configurações em um Container, proporcionando assim uma maior portabilidade e facilidade na implantação de aplicações em diferentes ambientes

O Java Spring Boot é um framework de desenvolvimento Web que fornece uma plataforma robusta para a criação de Aplicativos Web escaláveis e de alto desempenho

Combinar essas duas ferramentas nos permite criar um ambiente de desenvolvimento eficiente e altamente portátil para aplicativos Java

Criando um App

Primeiramente crie seu App utilizando o **Spring Initializr**

Acesse em seu navegador: <https://start.spring.io/>

1) Usaremos as seguintes configurações:

Project: **Maven**
Language: **Java**
Spring Boot: **2.7.10**
Group: **com.example**
Artifact: **app**
Name: **app**
Packaging: **Jar**
Java: **8**

2) Clique no botão "Add Dependencies" e adicione a seguinte dependência:
Spring Web

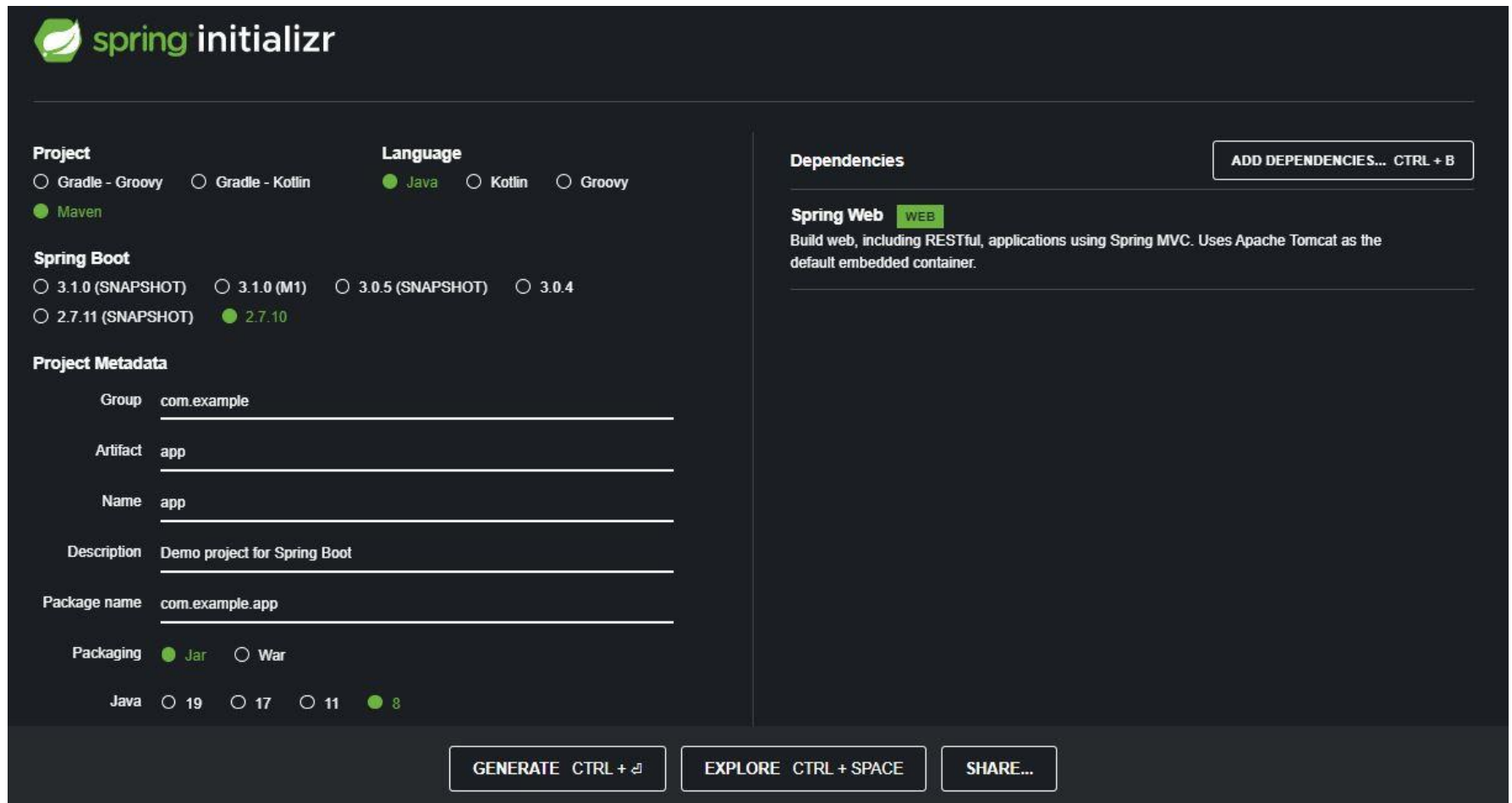
Para criar um aplicativo Web MVC

- 3) Clique no botão "Generate" para baixar um arquivo ZIP contendo o projeto Spring Boot
- 4) Depois de baixar e extrair o arquivo ZIP, você pode abrir o projeto em sua IDE favorita e começar a implementar sua aplicação Spring Boot
- 5) Fique a vontade para realizar qualquer tipo de codificação nesse App Java Spring Boot. Aqui iremos passar uma página simples, pois o foco é a criação do Dockerfile e Container com o App funcionando
- 6) Altere o código gerado em:
`\src\main\java\com\example\app\AppApplication.java`

O professor irá passar o fonte desse arquivo

Criando um App

Tela demonstrando as propriedades do App a ser gerado



The image shows the Spring Initializr web interface, which is used to generate a Spring Boot project. The interface is dark-themed and contains several sections for configuring the project.

Project

- ☐ Gradle - Groovy
- ☐ Gradle - Kotlin
- ☒ **Java**
- ☐ Kotlin
- ☐ Groovy

Language

- ☒ **Maven**

Spring Boot

- ☐ 3.1.0 (SNAPSHOT)
- ☐ 3.1.0 (M1)
- ☐ 3.0.5 (SNAPSHOT)
- ☐ 3.0.4
- ☐ 2.7.11 (SNAPSHOT)
- ☒ **2.7.10**

Project Metadata

Group:

Artifact:

Name:

Description:

Package name:

Packaging: ☒ **Jar** ☐ War

Java: ☐ 19 ☐ 17 ☐ 11 ☒ **8**

Dependencies

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

[ADD DEPENDENCIES... CTRL + B](#)

[GENERATE CTRL + ⌘](#) [EXPLORE CTRL + SPACE](#) [SHARE...](#)

Criando um App

Agora vá até o diretório raiz do Projeto, compile e gere o arquivo JAR do App

cd app

mvn clean compile

mvn package

Temos agora um App para rodar em um Container

Agora chegou a sua vez!

1) Crie o Dockerfile, a imagem e o Container para rodar esse App

O Dockerfile deve conter as seguintes tarefas:

1 - Imagem a ser utilizada como base: **openjdk:8u111-alpine**

2 - Atualize o repositório do Alpine

3 - Crie um novo usuário com o nome: **usrapp**

4 - Use esse usuário para executar os próximos comandos

5 - Defina um diretório padrão como **/app**

6 - Copie o arquivo: **app-0.0.1-SNAPSHOT.jar** para o diretório padrão criado anteriormente

7 - Exponha a porta **8080**

8 - Execute o comando: **java -jar app.jar** (Utilize CMD)

Criar a Imagem e Rodar o Container

- 2) Crie a imagem **meu-app-docker** a partir do seu Dockerfile criado
- 3) Rode o Container em segundo plano com o nome **myapp**
- 4) Realize os testes em **<http://localhost:8080>**
- 5) Entre no terminal do Container através do comando **docker container exec** e verifique as tarefas de SO do Dockerfile (diretório e usuário criados, arquivo copiado etc)

\src\main\java\com\example\app\AppApplication.java

```
package com.example.app;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication
@RestController
public class AppApplication {
    public static void main(String[] args) {
        SpringApplication.run(AppApplication.class, args);
    }

    @GetMapping("/")
    public String greeting() {
        return "Saudações do Java!";
    }
}
```

Dockerfile

```
FROM openjdk:8u111-alpine
```

```
RUN apk update -y
```

```
RUN adduser -h /home/usrapp -s /bin/bash -D usrapp
```

```
USER usrapp
```

```
WORKDIR /app
```

```
COPY target/app-0.0.1-SNAPSHOT.jar ./app.jar
```

```
EXPOSE 8080
```

```
CMD ["java", "-jar", "app.jar"]
```

Compilar o Projeto

mvn clean compile

Gerar o arquivo JAR

mvn package

Gerar a Imagem

```
docker build -t meu-app-docker .
```

Rodar o Container

```
docker run --name myapp -p 8080:8080 -d meu-app-docker
```

Entrar no Terminal

```
docker exec -it myapp /bin/sh
```

LIMPAR O LABORATÓRIO DO DOCKER COMPOSE

`docker container stop myapp`

`docker system prune -a -f --volumes`



Copyright © 2023 Prof. João Carlos Menk

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).