



## Primeiros passos – Criar o Start Spring Boot

Na linguagem Java, existem diversos frameworks destinados a diferentes tarefas, além dos mais variados pacotes que se interligam, funcionando em conjunto para atender as regras de negócio de um projeto. Antes, sem um framework Spring Boot, era necessário configurar pacote por pacote e framework por framework em diversos arquivos de configurações de extensões .XML, além de configurações de variáveis de ambiente e classpath, o que, no início do projeto, gastava um tempo considerável para primeiramente configurar o projeto e depois iniciar a aplicação da regra de negócio e dos requisitos.

O Spring Boot é um framework de desenvolvimento de sistemas e aplicações e é responsável por gerenciar a aplicação em tempo de execução e as dependências da aplicação, facilitando as configurações entre diversos frameworks e pacotes. Além disso, esse framework também possui automatização de códigos do projeto Spring com as anotações @annotations. Essa automatização considera configurações, instanciações e comportamentos da classe.

Para utilizar o Spring Boot, existem diversas formas permitidas pelo próprio Eclipse ou Maven. Porém, é preciso inserir vários formatos .XML para o correto funcionamento. No intuito de evitar esses passos dispendiosos para configurar o projeto Spring Boot, os mantenedores do framework criaram o site [start.spring.io](https://start.spring.io) para auxiliar nas configurações iniciais do projeto.

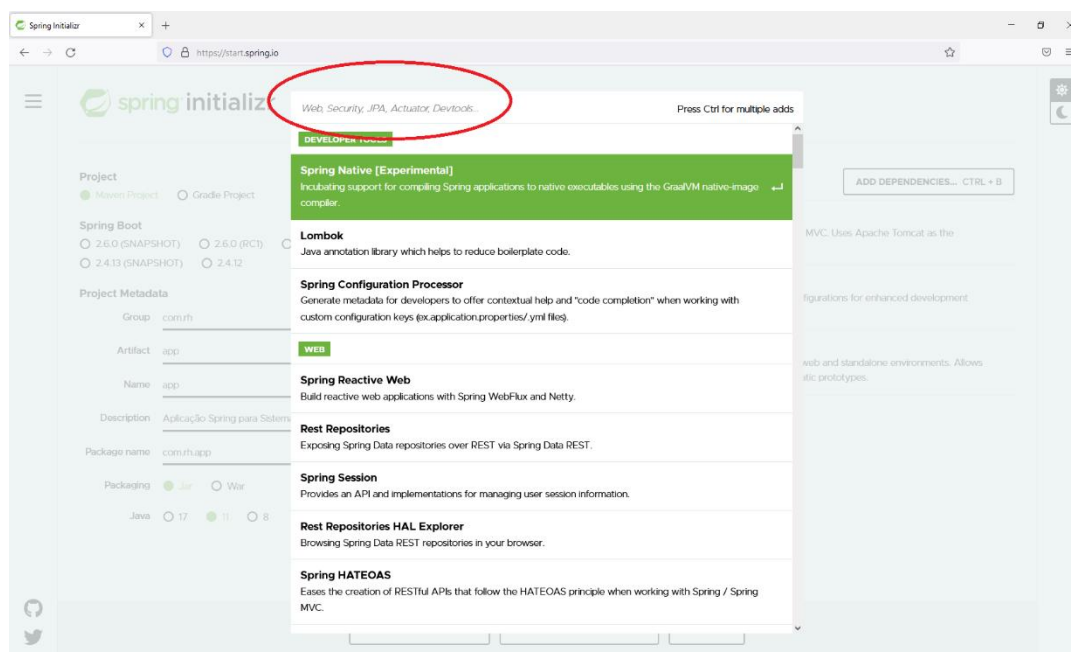
### Inicializando

Spring Initializr – **Passo 1:** entrar em <https://start.spring.io/>.

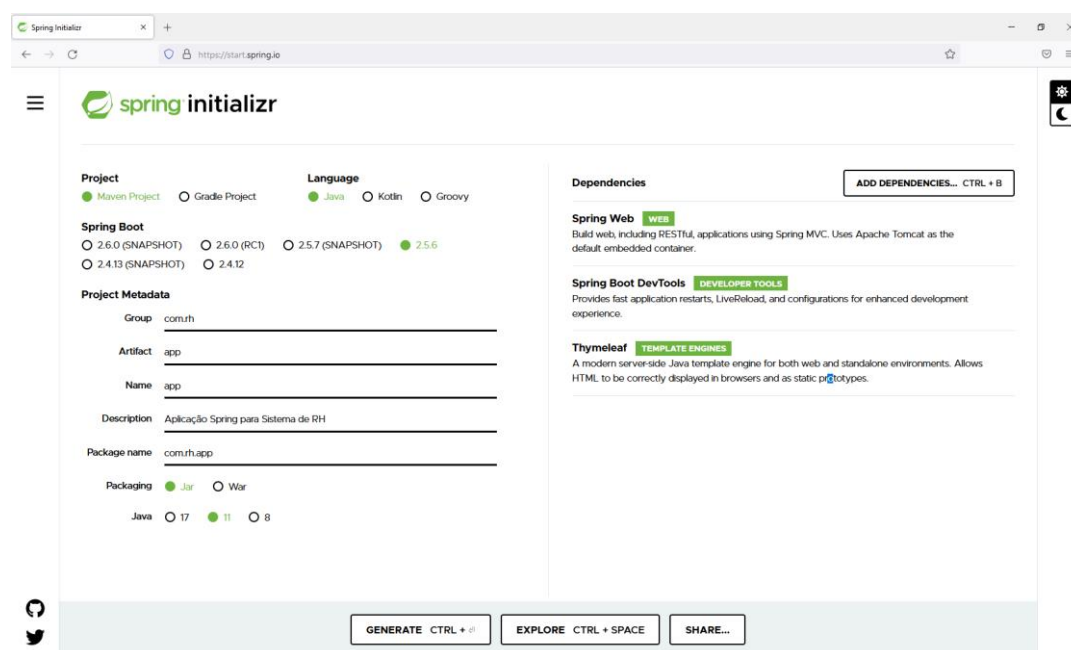
No campo Group é possível nomear o pacote do sistema. Já o campo Artifact será reservado ao nome do arquivo gerado, e o campo Name é o nome do sistema. O Package name é gerado automaticamente. Os campos Group, Artifact e Description devem ser preenchidos, no caso do exemplo, como demonstrado na figura a seguir.

**Passo 2:** no campo Packaging, selecionar **.jar** e **Java 11**. Em Project, selecionar **Maven Project**, e em Dependencies, escolher **ADD dependencies**.

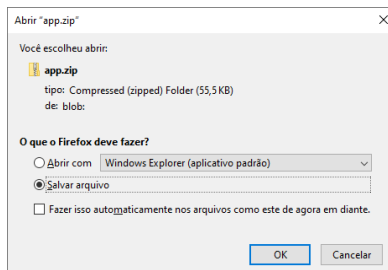
**Passo 3:** as dependências são inseridas ao clicar na opção **ADD DEPENDENCIES**. Na sequência, aparecerá uma tela como na figura a seguir.



Deve-se **digitar e selecionar** as seguintes dependências: **Spring Web**, **Spring Boot DevTools** e **Thymeleaf**.

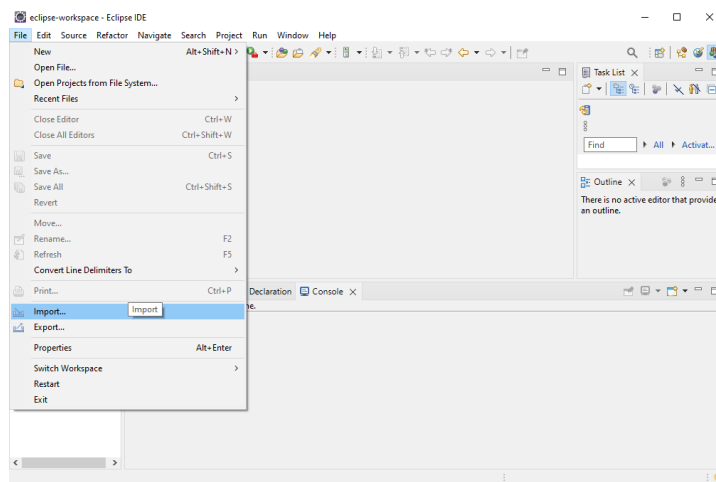


Será gerado um arquivo **app.zip** com as configurações básicas, pacotes e frameworks iniciais para implementação do projeto. Selecionar **Salvar arquivo** e depois clicar em **OK**.

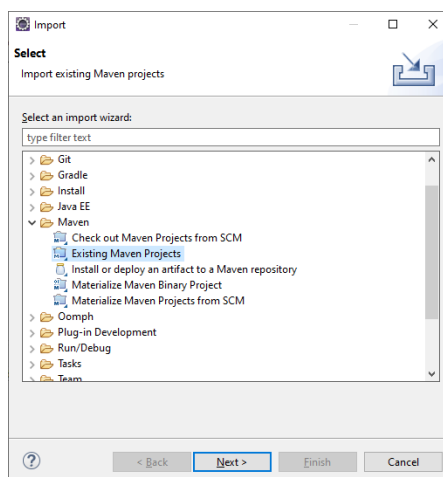


**Passo 4:** copiar o arquivo **app.zip** para a pasta **eclipse-workspace** (essa pasta geralmente está na pasta do usuário do sistema) da **IDE Eclipse**. Em seguida, extrair o arquivo **app.zip** de forma que gere a pasta **app**.

**Passo 5:** após efetuar o passo anterior, clicar em **File** e depois em **Import**.

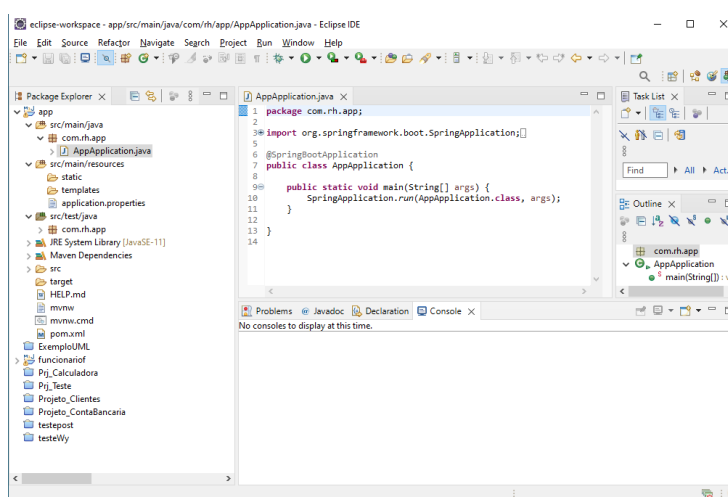


**Passo 6:** selecionar **Existing Maven Projects** e clicar em **Next**.

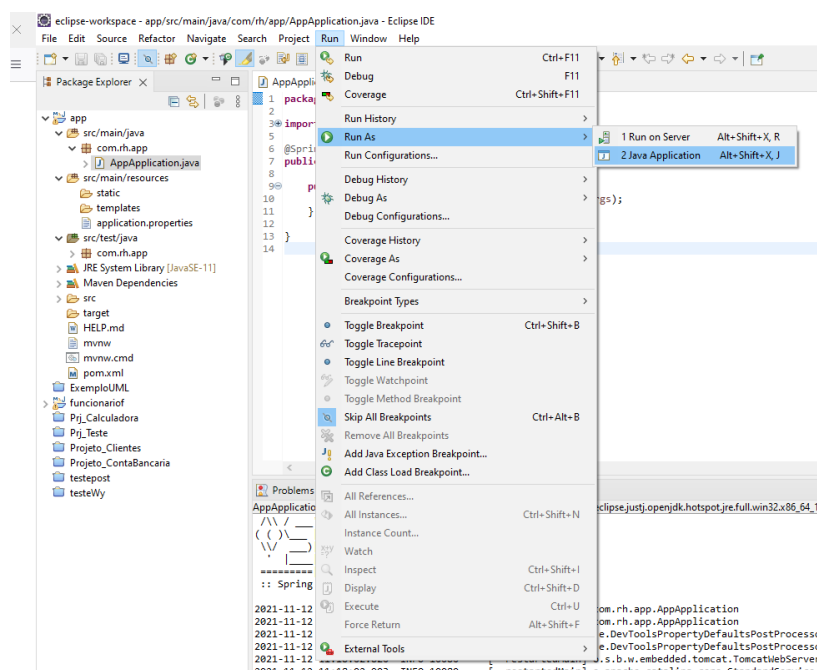


No **Browser**, selecionar a pasta **eclipse-workspace** e, em seguida, a pasta **app**.

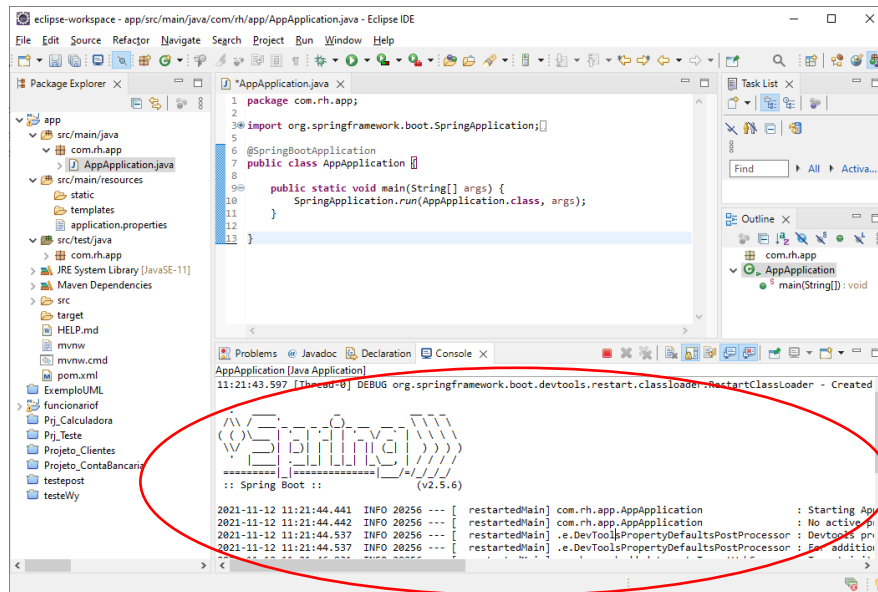
Ao carregar o projeto, abra a classe **AppApplication.java** que será executada no projeto, pois possui o método **main**. Essa classe está localizada em **src/main/java**, no pacote **com.app.rh**.



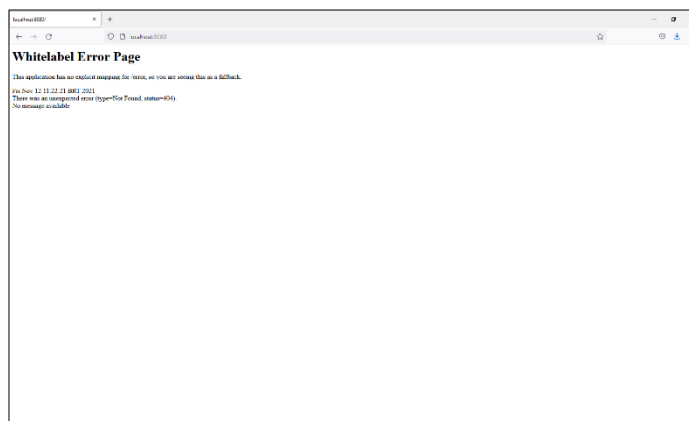
Para executar: no menu superior, clicar em **Run** e, em seguida, em **Run As**; depois selecionar **2 Java Application**.



Ao executar a classe **AppApplication.java**, o console apresenta o Spring carregando o servidor Tomcat e abrindo o Spring Boot, para executar o projeto criado pelo **Start Initializr**, como na figura a seguir.



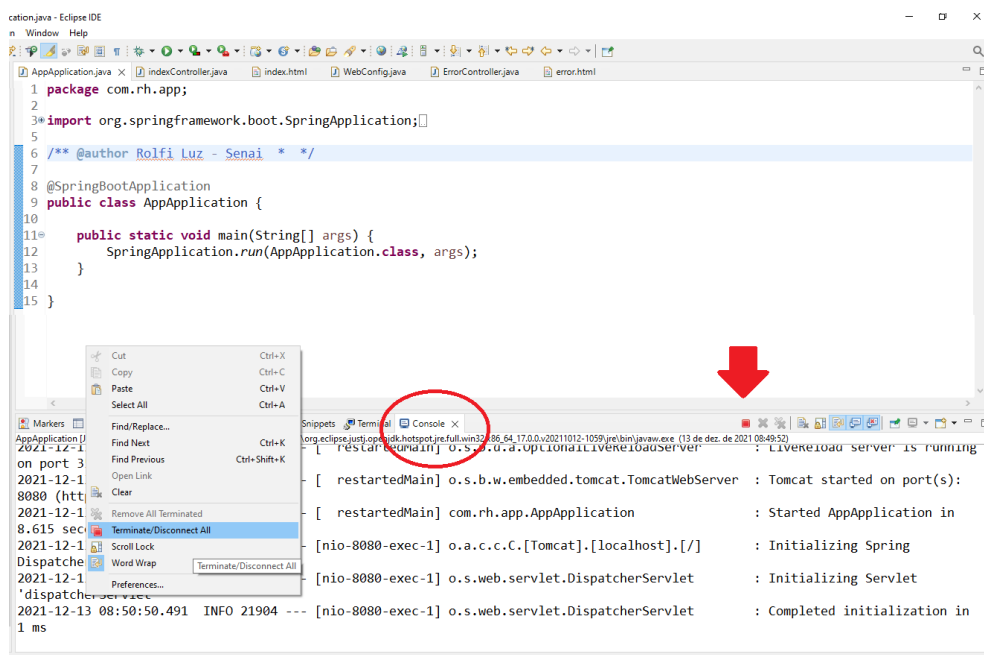
Após o carregamento, é possível testar o sistema abrindo o **Browser** com o endereço: **localhost:8080**.



Esse erro de página não encontrada é normal no início do Spring, caso não tenha uma página inicial.

Estes passos são apenas para configuração inicial do Spring Boot, após tais etapas, o projeto ficará salvo na pasta do Eclipse, e toda programação poderá ser feita pela IDE Eclipse e depois ser executada.

Para atualizar o projeto, basta salvá-lo, o projeto iniciará automaticamente no servidor. Caso precise desligar o servidor por algum erro inesperado ou situação específica do projeto, basta clicar em **stop** ou, clicar com o botão direito do mouse sobre a opção console, selecionar **terminate/disconnect all**, como ilustrado na figura a seguir.



The screenshot shows the Eclipse IDE interface. The main editor displays a Java file named `AppApplication.java` with the following code:

```
1 package com.rh.app;
2
3 import org.springframework.boot.SpringApplication;
4
5
6 /** @author Rolfi Luz - Senai */
7
8 @SpringBootApplication
9 public class AppApplication {
10
11     public static void main(String[] args) {
12         SpringApplication.run(AppApplication.class, args);
13     }
14
15 }
```

A context menu is open over the code, with the **Terminate/Disconnect All** option highlighted. A red arrow points to the console window at the bottom right, which shows the following output:

```
org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64.jdk-17.0.0.v20211012-1059\jre\bin\javaw.exe (13 de dez. de 2021 08:49:52)
[ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s):
[ restartedMain] com.rh.app.AppApplication : Started AppApplication in
[nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring
[nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet
[nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in
2021-12-13 08:50:50.491 INFO 21904 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in
1 ms
```