



Spring

Treinamento

Professor: Anderson Almada

Introdução

- O Spring Boot é um projeto da Spring que veio para facilitar o processo de configuração e publicação de nossas aplicações.
- A intenção é ter o seu projeto rodando o mais rápido possível e sem complicação.
- Basta que você diga pra ele quais módulos deseja utilizar (Web, Persistência, Segurança, ...) que ele vai reconhecer e configurar.

Introdução

- Você escolhe os módulos que deseja através dos starters que inclui no **pom.xml** do seu projeto.
- O maior benefício do Spring Boot é que ele nos deixa mais livres para pensarmos nas regras de negócio da nossa aplicação.

Dependências

- Instalar Eclipse
- Instalar PostgreSQL

Eclipse

- Acesse:
 - <https://www.eclipse.org/downloads/packages/release/2019-12/r/eclipse-ide-enterprise-java-developers>
- Escolha o SO da sua máquina:
 - Linux

Eclipse

- No terminal:
 - Entre na pasta em que foi feito o download do eclipse
 - Execute o seguinte comando:
 - `tar -xvf eclipse-jee-2019-12-R-linux-gtk-x86_64.tar.gz`
- Feito isso, acesse a pasta do eclipse extraído.
- Execute o arquivo eclipse.

PostgreSQL

- Acesse:
 - <https://www.postgresql.org/download/linux/ubuntu/>
- No terminal:
 - `sudo apt install postgresql-11 pgadmin4`
- Credenciais do usuário postgres
 - Usuário: postgres
 - Senha: postgres

PostgreSQL (Opcional)

- No terminal:
 - `sudo -u postgres psql`
 - `\password postgres`
 - `\q`

Pgadmin4

- Abra o PgAdmin4
- Configure a senha para acesso
- Clique em **Add new server**

Welcome



Feature rich

pgAdmin is an Open Source PostgreSQL procedural code editor.

Quick Links

Getting Started

PostgreSQL

 Create - Server

General

Connection

SSL

SSH Tunnel

Advanced

Name

projeto|

Server group

Servers

Background




Foreground



Connect now?



Comments

 Either Host name, Address or Service must be specified.



✕ Cancel

 Reset Save

includes a graphical administration interface, an SQL query tool, a backup utility, and a security tool. It is suitable for use by DBAs and system administrators alike.



Configure pgAdmin



et PostgreSQL



Community Support

demo.zip

eclipse-jee-2...tar.gz

eclipse-inst-li...tar.gz

[Show all](#)

127.0.0.1:36835/browser/#

pgAdmin File Object Tools Help

Browser Servers

Dashboard Properties

Welcome

Feature rich

pgAdmin is an Open Source PostgreSQL administration and development tool. It includes a graphical administration interface, an SQL query tool, a procedural code editor, and much more. It is designed for database administrators, developers, DBAs and system administrators alike.

Quick Links

Getting Started

PostgreSQL

Create - Server

General Connection SSL SSH Tunnel Advanced

Host name/address: localhost

Port: 5432

Maintenance database: postgres

Username: postgres

Password:

Save password? ☒

Role:

Service:

Cancel Reset Save

demo.zip eclipse-jee-2....tar.gz eclipse-inst-li....tar.gz

Show all

Configure pgAdmin

Get PostgreSQL

Community Support

Browser 127.0.0.1:36835/browser/#

pgAdmin File Object Tools Help

Browser Servers (1) projeto Databases (1) postgres Login/Group Roles Tablespaces

Create Database... Refresh...

Dashboard Properties SQL Statistics Dependencies Dependents

Server sessions

Session State	Count
Idle	6.0
Other	1.0
Other	0.0

Transactions per second

Metric	Value
Transactions	~1.0
Commits	~5.0
Rollbacks	~0.0

Tuples in

Operation	Value
Inserts	~0.00
Updates	~1.00
Deletes	~0.00

Tuples out

Operation	Value
Fetched	~1000
Returned	~1000

Block I/O

Operation	Value
Reads	~0
Hits	~100

Server activity

Sessions Locks Prepared Transactions Configuration

			PID	Database	User	Application	Client	Backend start	State	Wait event	Blocking PIDs

demo.zip eclipse-jee-2....tar.gz eclipse-inst-li....tar.gz

Show all



Servers (1)

projeto

Databases (2)

postgres

projeto

Casts

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Schemas

Login/Group Roles

Tablespaces

Database sessions



Transactions per second



Tuples in



Tuples out



Block I/O



Server activity

Sessions

Locks

Prepared Transactions



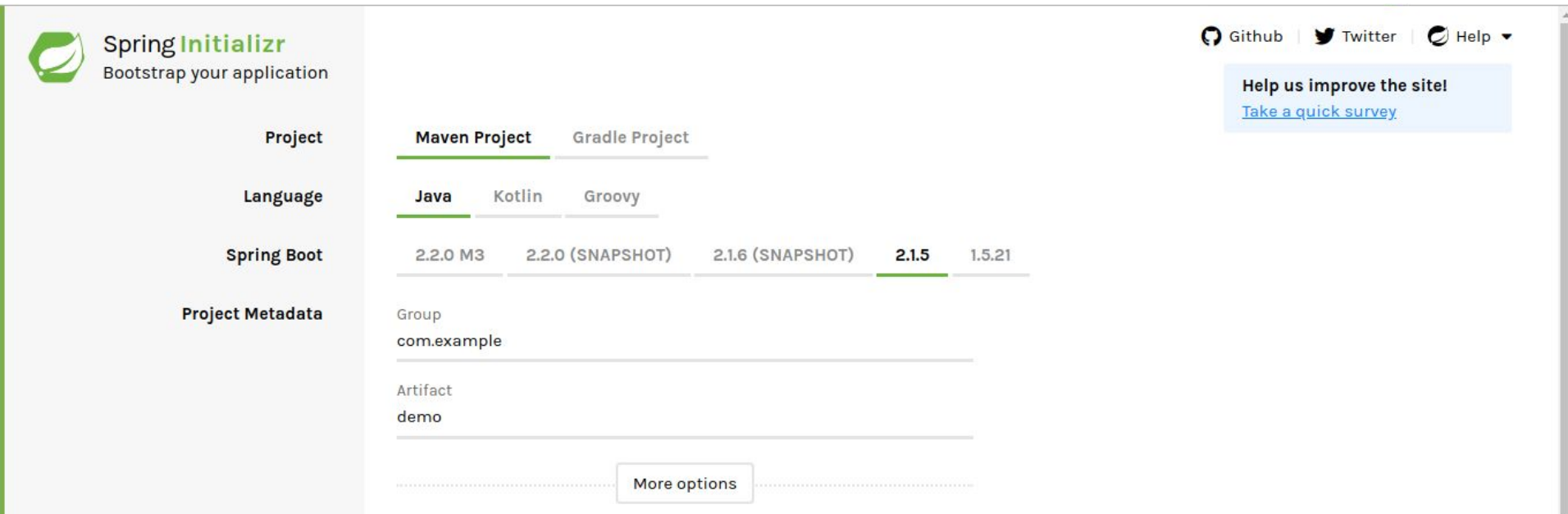
Search



		PID	User	Application	Client	Backend start	State	Wait event	Blocking PIDs
--	--	-----	------	-------------	--------	---------------	-------	------------	---------------

Iniciando o projeto

- Como opcional, utilize a ferramenta do spring para a criação dos projetos
 - <https://start.spring.io/>



The screenshot shows the Spring Initializr web application. On the left is a sidebar with the Spring logo and the text 'Spring Initializr Bootstrap your application'. Below this are four sections: 'Project' (with 'Maven Project' and 'Gradle Project' tabs, 'Maven Project' is selected), 'Language' (with 'Java', 'Kotlin', and 'Groovy' tabs, 'Java' is selected), 'Spring Boot' (with version tabs: '2.2.0 M3', '2.2.0 (SNAPSHOT)', '2.1.6 (SNAPSHOT)', '2.1.5' (selected), and '1.5.21'), and 'Project Metadata' (with input fields for 'Group' containing 'com.example' and 'Artifact' containing 'demo'). At the bottom of the sidebar is a 'More options' button. On the right side of the page, there are links for 'Github', 'Twitter', and 'Help'. Below these is a blue box with the text 'Help us improve the site!' and a link 'Take a quick survey'.

Spring Initializr
Bootstrap your application

Project

Language

Spring Boot

Project Metadata

Group
com.example

Artifact
demo

More options

Github | Twitter | Help

Help us improve the site!
[Take a quick survey](#)



Spring Initializr

Bootstrap your application

Project

Maven Project

Gradle Project

Language

Java

Kotlin

Groovy

Spring Boot

2.2.0 M3

2.2.0 (SNAPSHOT)

2.1.6 (SNAPSHOT)

2.1.5

1.5.21

Project Metadata

Group

com.example

Artifact

demo

Name

demo

Description

Demo project for Spring Boot

Generate Project - alt + ⌘



Github



Twitter



Help

Help us improve the site!

[Take a quick survey.](#)

© 2013-2019 Pivotal Software

start.spring.io is powered by

[Spring Initializr](#) and [Pivotal Web Services](#)

Package Name
com.example.demo

Packaging

Jar **War**

Java Version

12 11 **8**

Fewer options

Dependencies

[See all](#)

Search dependencies to add

Web, Security, JPA, Actuator, Devtools...

Selected dependencies

DevTools [Core]

Spring Boot Development Tools

Web [Web]

Servlet web application with Spring MVC
and Tomcat

Dependências no Spring

- Spring WEB
- Spring Data JPA
- Spring DevTools
- PostgreSQL

Package Name

com.example.demo

Packaging

Jar

War

Java Version

12

11

8

Fewer options

Dependencies

[See all](#)

Search dependencies to add

Web, Security, JPA, Actuator, Devtools...

Selected dependencies

DevTools [Core]

Spring Boot Development Tools

Web [Web]

Servlet web application with Spring MVC
and Tomcat

Generate Project - alt + ⌘

Iniciando o projeto no Eclipse

- Extraí o arquivo o qual foi realizado o download
- Importa no Eclipse (maven)
- Espere a realização dos downloads das dependências

application.properties

PostgreSQL

spring.datasource.url=**jdbc:postgresql://localhost:5432/projeto**

spring.datasource.username=**postgres**

spring.datasource.password=**postgres**

spring.jpa.hibernate.ddl-auto=**update**

spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect

Rodando o projeto no Eclipse

- Feito a importação e a configuração no arquivo `application.properties`, execute o projeto como um **java application**

- Java Resources
 - src/main/java
 - com.example.demo
 - DemoApplication.java
 - ServletInitializer.java
 - src/main/resources
 - static
 - templates
 - application.properties
 - src/test/java
 - Libraries

JavaScript Resources

Deployed Resources

 - src
 - target
 - HELP.md
 - mvnw
 - mvnw.cmd
 - pom.xml

Markers Properties Servers Data Source Explorer Snippets Console

DemoApplication [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (Feb 17, 2020, 6:36:56 PM)

```
2020-02-17 18:37:01.814 INFO 4895 --- [ restartedMain] org.hibernate.dialect.Dialect : HHH000400: Using dialect: c
2020-02-17 18:37:02.343 INFO 4895 --- [ restartedMain] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000490: Using JtaPlatform
2020-02-17 18:37:02.352 INFO 4895 --- [ restartedMain] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManag
2020-02-17 18:37:02.370 INFO 4895 --- [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is runnin
2020-02-17 18:37:02.451 WARN 4895 --- [ restartedMain] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is
2020-02-17 18:37:02.642 INFO 4895 --- [ restartedMain] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService
2020-02-17 18:37:02.971 INFO 4895 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s):
2020-02-17 18:37:02.976 INFO 4895 --- [ restartedMain] com.example.demo.DemoApplication : Started DemoApplication in
2020-02-17 18:37:17.534 INFO 4895 --- [nio-8080-exec-2] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring Dispatc
2020-02-17 18:37:17.534 INFO 4895 --- [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispa
2020-02-17 18:37:17.561 INFO 4895 --- [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet : Completed initialization in
```

Crie os pacotes

- `com.example.demo.model`
- `com.example.demo.repository`
- `com.example.demo.service`
- `com.example.demo.controller`

Classe Users

- Package: model
 - **@Entity**
 - **@Id**
 - **@GeneratedValue**
 - **@Column(columnDefinition="text")**
 - Código-fonte
 - <https://pastebin.com/520daWKh>

Classe UserRepository

- Package: repository

```
public interface UserRepository extends JpaRepository<Users, Integer> {  
  
}
```

(Optional) Classe UserRepository

- Package: repository

```
public interface UserRepository extends JpaRepository<Users, Integer> {  
    Users findFirstByName(String name);  
}
```

Native Queries

https://docs.spring.io/spring-data/jpa/docs/current/reference/html/#_native_queries

Classe UserService

- Package: service

@Service

```
public class UserService {
```

```
    @Autowired
```

```
    UserRepository userRepo;
```

```
}
```

Classe UserService

```
public Users addUser(Users users) {  
    return userRepo.save(users);  
}
```

Classe UserService

```
public boolean removeUser(Integer id) {  
    if(userRepo.existsById(id)) {  
        userRepo.deleteById(id);  
        return true;  
    }  
  
    return false;  
}
```

Classe UserService

```
public List<Users> getUsers() {  
    return userRepo.findAll();  
}
```

```
public Users getUser(Integer id) {  
    return userRepo.findById(id).get();  
}
```

```
public Users getUserByName(String name) {  
    return userRepo.findFirstByName(name);  
}
```

Classe UserService

```
public Users updateUser(Integer id, String name, String password) {  
    Users userAux = userRepo.findById(id).get();  
    if(userAux != null) {  
        userAux.setName(name);  
        userAux.setPassword(password);  
        userRepo.save(userAux);  
    }  
  
    return userAux;  
}
```

Classe UserService

- Código-fonte
 - <https://pastebin.com/TqVWSkpK>

Classe UserController

- Package: controller

@RestController

@RequestMapping(path = "/api/users")

public class UserController {

@Autowired

UserService userService;

GET ALL

```
@RequestMapping(method = RequestMethod.GET)
public ResponseEntity<List<Users>> getUsers() {
    return new ResponseEntity<List<Users>>(userService.getUsers(),HttpStatus.OK);
}
```

GET ID

```
@RequestMapping(method = RequestMethod.GET, value = "{id}")  
public ResponseEntity<Users> getUser(@PathVariable("id") Integer id) {  
    return new ResponseEntity<Users>(userService.getUser(id),HttpStatus.OK);  
}
```

(Optional) GET CURSO BY NOME

```
@RequestMapping(method = RequestMethod.GET, value = "/search")  
public ResponseEntity<Users> getUserByName(@RequestParam("name") String name) {  
    return new ResponseEntity<Users>(userService.getUserByName(name),HttpStatus.OK);  
}
```

POST

```
@RequestMapping(method = RequestMethod.POST)
public ResponseEntity<Users> addUser(@RequestBody Users user) {
    return new ResponseEntity<Users>(
        userService.addUser(
            new Users(-1, user.getName(), user.getPassword())), HttpStatus.OK);
}
```

PUT

```
@RequestMapping(method = RequestMethod.PUT, value = "{id}")
public ResponseEntity<Users> updateUser(@PathVariable("id") Integer id,
    @RequestBody Users user) {
    return new ResponseEntity<Users>(
        userService.updateUser(id, user.getName(), user.getPassword()), HttpStatus.OK);
}
```

DELETE

```
@RequestMapping(method = RequestMethod.DELETE, value = "{id}")
public ResponseEntity<Void> deleteUser(@PathVariable("id") Integer id) {
    if(userService.removeUser(id)) {
        return new ResponseEntity<Void>(HttpStatus.NO_CONTENT);
    }

    return new ResponseEntity<Void>(HttpStatus.NOT_FOUND);
}
```

Classe UserController

- Código-fonte
 - <https://pastebin.com/vvDB9XsK>

Links importantes

- <https://grokonez.com/spring-framework/spring-data/spring-boot-vue-exemple-spring-data-jpa-rest-postgresql-crud-example>
- <https://www.mkyong.com/spring-boot/spring-boot-spring-data-jpa-postgresql/>
- <https://docs.spring.io/spring-data/jpa/docs/current/reference/html/>
- <https://blog.algaworks.com/spring-data-jpa/>



Dúvidas??

E-mail: almada@crateus.ufc.br