



Serviço Nacional de Aprendizagem Industrial

PELO FUTURO DO TRABALHO

Estrutura de Projeto

Desenvolvimento de Sistemas

Prof. Me. Reneilson Santos

Março/2024

Agenda

- Estruturação de Projetos Spring Boot
 - ◆ Por “*features*”
 - ◆ Por camadas



Estruturação de Projetos Spring Boot

Estrutura de Arquivos

Não há layout específico ou estrutura de código para projetos Spring Boot.

No entanto, existem algumas **práticas recomendadas** seguidas pelos desenvolvedores:

- **divisão em camadas** (como camada de serviço, camada de entidade, camada de repositório, etc).
- divisão em módulos. Por exemplo, o projeto pai tem dois módulos filhos. O primeiro módulo é para a camada de dados e o segundo módulo para a camada web.
- **divisão em recursos** (*features*).

Estrutura de Arquivos

Recomenda-se colocar a classe Main Application no pacote raiz com anotações como `@SpringBootApplication` ou `@ComponentScan` ou `@EnableAutoConfiguration`.

Ele permite que o Spring verifique todas as classes no pacote raiz e nos subpacotes.

Por exemplo, se você estiver escrevendo um aplicativo, o `MainApplicaiton.java` é colocado no pacote raiz e todas as demais classes relacionadas relacionadas ao projeto em subpacotes (que podem ser relacionados às camadas ou aos recursos, conforme escolha de qual estrutura seguir).

Divisão em Recursos

Nesta abordagem, todas as classes pertencentes a um determinado recurso são colocadas no mesmo pacote.

com

+ - example

+ - demo

+ - MyApplication.java

+ - **customer**

| + - Customer.java

| + - CustomerController.java

| + - CustomerService.java

| + - CustomerRepository.java

+ - **order**

| + - Order.java

| + - OrderController.java

| + - OrderService.java

| + - OrderRepository.java

cria-se um pacote para cada recurso

Vantagens

- Encontrar uma classe a ser modificada é fácil.
- Ao excluir um determinado subpacote, todas as classes relacionadas a um determinado recurso serão excluídas.
- Testar e refatorar é fácil.
- Os recursos podem ser usados separadamente.

Desvantagens

- Maior número de pacotes dentro do projeto à medida que o mesmo se torna mais complexo e aumenta a quantidade de recursos.

Divisão em Camadas

Nesta abordagem, todos os controladores podem ser colocados no pacote de controladores e serviços no pacote de serviços e todas as entidades no domínio ou modelo, etc. Ou seja, cria-se uma camada para cada tipo de classe do projeto.

com

+ - example

+ - demo

+ - MyApplication.java

+ - **models**

| +- Customer.java

| +- Order.java

+ - **controllers**

| +- OrderController.java

| +- CustomerController.java

+ - **services**

| +- CustomerService.java

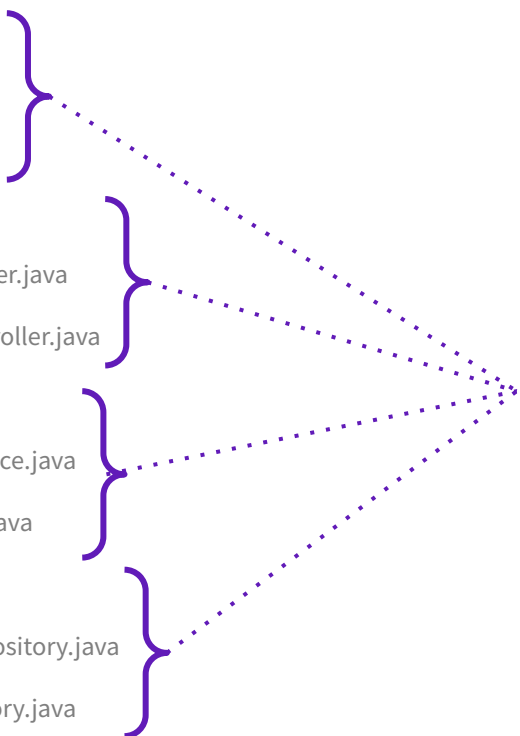
| +- OrderService.java

+ - **repositories**

| +- CustomerRepository.java

| +- OrderRepository.java

cria-se um pacote para cada camada



Vantagens

- Fácil de encontrar as classes por camadas;

Desvantagens

- Recursos ou módulos não podem ser usados separadamente.
- Difícil localizar uma classe pertencente a um determinado recurso.
- A refatoração de código em um determinado recurso é mais difícil, pois as classes de recursos estão localizadas em todas as camadas.
- Pode causar uma maior quantidade de conflitos de mesclagem entre desenvolvedores que usam sistemas de versionamento de código (GitHub, BitBucket etc).

Bibliografia

Referências Bibliográficas

- Code Structure. Disponível em:
<https://www.geeksforgeeks.org/spring-boot-code-structure/>. Acessado em: 02 de março de 2023.