Altar Code Labs

Prática 2

UP ACADEMY
LEARNING FOR SUCCESS

Prática

1. Crie a classe Entity no package model. Esta classe irá conter os campos e métodos comuns do Product e Shelf. Faça com que Product e Shelf herdem de Entity. O ID na Entity deve ser do tipo Long (wrapper do tipo primitivo long)
2. Crie o package *repositories* (dentro do package *io.altar.jseproject*)

Prática

3. Dentro do package *repositories*, crie uma classe abstrata EntityRepository. Esta classe irá ter como parâmetro generic uma classe que seja subclasse de Entity. Esta classe deverá ter um atributo que é um Mapa (por exemplo um LinkedHashMap) cuja key é o ID da entidade e cujo value é a classe parâmetro generic. Este mapa vai ser a nossa "Base de Dados". Deve também ser criado um outro atributo do tipo Long que vai conter o maior ID presente na base de dados. Deve também existir um método privado que devolve o próximo ID. Deve depois inserir nesta classe os métodos necessários para:
    a. **Criar** entidades (recebe um objeto do tipo da classe parâmetro generic que não possuirá ainda um ID e devolve um Long com o novo ID),
    b. **Consultar** entidades (sugiro 2 métodos: 1 que não recebe parâmetros e que devolve todos os elementos da base de dados e outro que recebe um ID e devolve a entity correspondente)
    c. **Editar** entidades (recebe como parâmetro uma Entity que deve ter já um ID)
    d. **Remover** entidades (recebe como parâmetro um ID)

Prática

4. Dentro do package *repositories*, crie uma classe ProductRepository. Esta classe irá ser uma subclasse da classe EntityRepository. Ela existe apenas para implementar o padrão de desenho Singleton para o repositório do produto. Este padrão serve para garantir que uma e uma só uma instância de uma determinada classe existe no sistema. Para isso deve:
    a. Criar um atributo (chamado *INSTANCE*) private static final do tipo da própria classe, ao qual é atribuída na sua inicialização, uma instância da própria classe
    b. Criar um método public static chamado getInstance que não recebe parâmetros e que devolve um ProductRepository. Este método devolve o conteúdo da variável *INSTANCE*
    c. Criar um construtor (o único) que não recebe parâmetros e que é private
5. Fazer o mesmo para a classe ShelfRepository
6. O passo seguinte será usar os Product e Shelf repositories nos métodos de interface já criados na aula prática passada

Prática

7. Por último, podemos alterar a forma de construir a nossa interface inspirando-nos no padrão de desenho descrito aqui:
    a. https://sourcemaking.com/design_patterns/state/java/5

Luís Ribeiro

I'm a software engineer with more than 14 years of experience in software development in Java and other technologies, software architecture, team management and project management.

My mission is to transform people's ideas into fully functional, production ready and user friendly software applications that can change the world!

luismmribeiro@gmail.com

Thank you