Altar Code Labs

Prática 1

UP ACADEMY
LEARNING FOR SUCCESS

Prática

1. Crie um novo projeto Java
2. Crie um package io.altar.jseproject onde todas as classes deste projeto irão existir (ou subpackages)
3. Crie os subpackages *textinterface* (*io.altar.jseproject.textinterface*), *model* (*io.altar.jseproject.model*) e *test* (*io.altar.jseproject.test*). O package *textinterface* irá conter a/as classes para gerar os inputs/outputs da interface, o package *model* irá conter o modelo da aplicação e o package *test* irá conter a/as classes que achar necessárias para testar o seu código

Prática

4. Crie as classes Shelf (prateleira) e Product (produto) no package model. Deve adicionar os atributos descritos no projeto com os tipos que achar mais convenientes. Para cada um destes atributos deve criar um getter e um setter. Caso mais à frente ache que são necessários mais métodos, eles poderão ser adicionados nessa altura
5. Crie a classe TextInterface onde deve criar o código da interface
6. Dentro dessa classe, crie um método por cada “ecra” descrito no enunciado. Por agora, cada método terá apenas a interação com o utilizador (apenas System.out.println e Scanner para ler cada linha de input).

Prática

6. Após criar cada um destes métodos, altere os métodos onde as opções a introduzir são fechadas (1, 2 e 3, por exemplo, como no 1º "ecra") de modo a que o utilizador se possa "enganar" e introduzir opções inválidas como números inexistentes ou texto. Nesse caso, deve ser apresentada uma mensagem de erro ao utilizador e a lista de opções deve ser apresentada de novo. O número de enganos possíveis deve poder ser "infinito", ou seja, o código deve ser flexível o suficiente para que o utilizador se "engane" quantas vezes quantas quiser, mas, assim que acertar, a opção deve ser corretamente tomada em conta.

7. Altere os métodos criados até agora de modo a que, após entrar no primeiro "ecra", seja possível navegar para os outros "ecras" de modo a criar o fluxo de funcionamento da aplicação (tem toda a liberdade para criar novos métodos ou reestruturar os métodos existentes ou ainda para criar novas classes de modo a reorganizar os métodos de uma forma que ache mais conveniente)

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