Relatório Exercício Motivador do uso de padrões de Projeto

Júlia Bettiol, Dhara Hadi e Marina Bon

Março 2025

Sumário:

Sumário: 1
Introdução: 1
Solicitações: 1
Implementação das solicitações 1, 2 e 3: 1
Implementação das solicitações 4 e 5: 2
Análise do código original: 2
Conclusão: 2

Introdução:

Este relatório tem como objetivo discutir o impacto das modificações nas classes originais para suportar as solicitações de mudanças no código e avaliar se a estrutura original poderia ter sido mais bem projetada. As implementações foram feitas de maneira em que o código possa estar cada vez mais eficiente e compreensível.

Solicitações:

Implementação das solicitações 1, 2 e 3:

A implementação das solicitações seguiu uma abordagem similar, garantindo a adição dos veículos e suas respectivas características. As modificações feitas foram:

- Adição dos novos modelos de veículos. Foram desenvolvidos três modelos distintos: **Esportivo, SUV e Utilitário**, cada um com atributos específicos de combustível, consumo e capacidade do tanque.
- Testes e execuções no *main* para instanciar os novos modelos, abastecê-los e realizar viagens.
- Correção na classe de abastecimento, garantindo que o carro “Utilitário” pudesse ser abastecido de maneira correta, evitando erro de tipo de combustível.

Implementação das solicitações 4 e 5:

A solicitação 4 pede para criar um modelo de carro chamado "SUVFlex". Este é FLEX (gasolina ou álcool) e consome 8 Km/litro de gasolina ou 6 Km/litro de álcool. O tanque é "FLEX" e tem 65 litros de capacidade. Para isso, foi implementada uma subclasse chamada “SUVFlex” que contém as caracteristicas do carro e um @*Override*. O *Override* está usando o método “abastece" e verificando qual tipo de combustível que está sendo usado no momento pelo veículo FLEX. Possibilitando que ele assume que o conteúdo do tanque é sempre o do último abastecimento.

A solicitação 5 pede a implementação de um modelo de carro chamado "Econo", que possui algumas diferenças, tem o consumo de 20 Km/litro, porém esse valor reduz em 1 Km/litro a cada 5000 Km rodados, até ficar estável em 10 Km/litro. Para poder implementar a solicitação foi necessário utilizar o "*@override*" sobre o método *viaja,* mudando o cálculo do consumo de combustível conforme a quilometragem.

Análise do código original:

O código original foi bem estruturado, utilizou um modelo orientado a objetos, dividindo "tarefas" em diferentes classes para representar a mecânica de um veículo. Um ponto de possível melhoria seria na classe *Motor*, pois o motor recebe um único valor de consumo, gerando um leve problema para combustível *FLEX*. Além disso, o código original não possui uma hierarquia que permita a criação de subclasses para diferentes tipos de veículos. Então para adicionar novos modelos como Esportivo, Utilitário e SUV, foi preciso instanciar novos objetos com os atributos corretos manualmente, em vez de definir subclasses que já contenham essas especificações.

Conclusão:

Após as análises e modificações no código original, podemos observar a importância de um código bem estruturado para facilitar nos ajustes e futuras adições no mesmo. No intuito de aprimorar o código, algumas modificações foram feitas, permitindo a adição de novos modelos sem alterar a estrutura principal da classe Carro.

Ao analisar o trabalho, percebemos como padrões de projeto podem tornar o código mais flexível e ajustável, reduzindo necessidades de grandes modificações. Sendo assim, o desenvolvimento do trabalho atendeu as solicitações de mudanças e mostrou a importância da utilização de boas práticas quando trabalhamos com software.