Universidade Tecnológica Federal do Paraná

Engenharia de Software - Curso de Arquitetura de Software (AS27S)

INSTRUTOR: Prof. Dr. Gustavo Santos

Aluno: Alex Domingues da Silva, RA: 1496727

CCH - Padrões de Projetos Criacionais - Builder

Problema

O propósito deste código é lidar com a criação de objetos de Patinete, oferecendo a possibilidade de personalizar características como cor, marca e quantidade de rodas. O objetivo é estabelecer uma estrutura flexível que permita construir esses objetos passo a passo, separando a lógica de construção da representação final do patinete.

Descrição da Solução

Neste código, utilizei o padrão de projeto Builder para a criação de objetos de Patinete em Java. O padrão Builder separa a lógica de construção do objeto da sua representação final, permitindo a criação flexível de objetos com diferentes configurações opcionais.

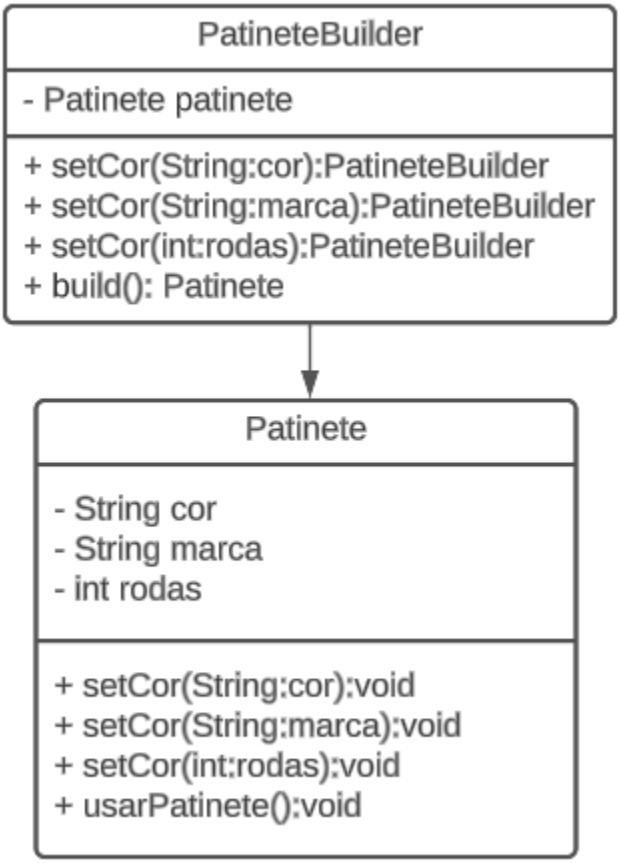
A solução consiste em duas principais classes:

- PatineteBuilder: implementa o padrão Builder e possui métodos para definir as características opcionais do Patinete, como cor, marca e número de rodas. Cada método retorna a própria instância do PatineteBuilder, permitindo a construção encadeada do objeto.

- Patinete: representa o objeto Patinete, contendo propriedades como cor, marca e número de rodas. Além disso, a classe possui métodos para exibir informações sobre o Patinete, como cor, marca e quantidade de rodas.

O objetivo desse código é demonstrar como usar o padrão Builder para criar objetos de Patinete de forma flexível e organizada, possibilitando a customização das características do Patinete conforme necessário.

Visão Geral



A classe PatineteBuilder é responsável pela construção do objeto Patinete. Ela possui uma propriedade patinete que representa o objeto sendo construído. Os métodos setCor(), setMarca() e setRodas() permitem definir as características opcionais do Patinete passo a passo. O método build() retorna o objeto Patinete final.

A classe Patinete representa o objeto Patinete e possui as propriedades cor, marca e rodas. Ela também contém o método usarPatinete() para realizar a ação de usar o patinete e exibir as informações relacionadas a ele.

Exemplo de Código Java

Exemplo do código: CCHS_AS

Consequências

- Vantagens
    - Criação flexível de objetos do Patinete com diferentes configurações opcionais, sem muitos construtores.
    - Encadeamento de chamadas de métodos para configurar as propriedades do Patinete de forma concisa.
    - Controle detalhado e consistente do processo de construção, evitando objetos inconsistentes ou em estados inválidos.
    - Reutilização de código, permitindo o reaproveitamento dos métodos de configuração em diferentes cenários de construção do Patinete
- Desvantagens
    - Introdução de complexidade adicional, principalmente em situações simples onde não há muitas opções de configuração.
    - Necessidade de criar um objeto adicional (PatineteBuilder) antes do objeto final (Patinete).
    - Aumento da quantidade de classes no projeto, o que pode ser indesejado em projetos muito simples ou com restrições de tamanho