Factory Method

Instituto Federal do Espirito Santo – Campus Colatina
Disciplina: Projeto de Sistemas
Professor: Giovany Frossard Teixeira

Elementos Essenciais

- **Nome:** *Factory Method (Virtual Constructor)*
- **Problema:** Desconhecer o tipo do objeto que será criado de forma prévia e estática. Em *frameworks*, por exemplo, pode-se definir um padrão de construção de objetos (através da definição de interfaces, por exemplo) e possibilitar que a construção em si seja feita pelo programador que está usando o *framework*.
- **Solução:** Definir uma interface para criar um objeto, mas deixar as subclasses decidirem que classe instanciar. Permite adiar a instanciação para subclasses.
- **Consequências:** Facilita o acréscimo de novas classes concretas. Grande quantidade de classes para resolver um problema pequeno.

Problema

- Suponha que você deve trabalhar em um projeto computacional com um conjunto de carros, cada um de uma determinada fábrica. Para exemplificar suponha os quatro seguintes modelos/fabricantes:
  - Palio – Fiat
  - Gol – Volkswagen
  - Celta – Chevrolet
  - Fiesta – Ford

Será necessário manipular este conjunto de carros em diversas operações, como poderíamos modelar este problema?

Fonte: https://brizeno.wordpress.com/category/padroes-de-projeto/factory-method/ (acessado em 23/01/2020)

Factory Method

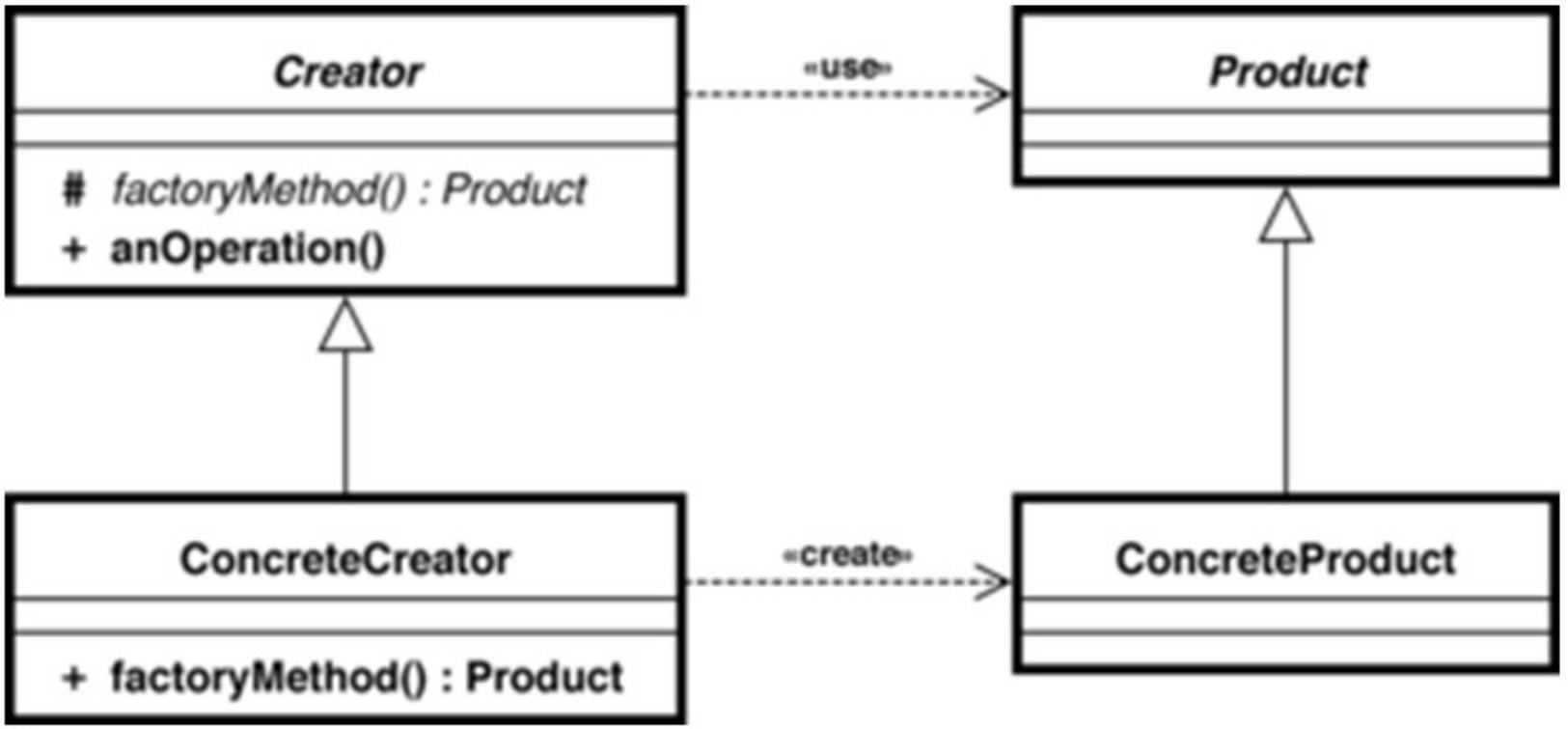


Diagrama de classes do exemplo

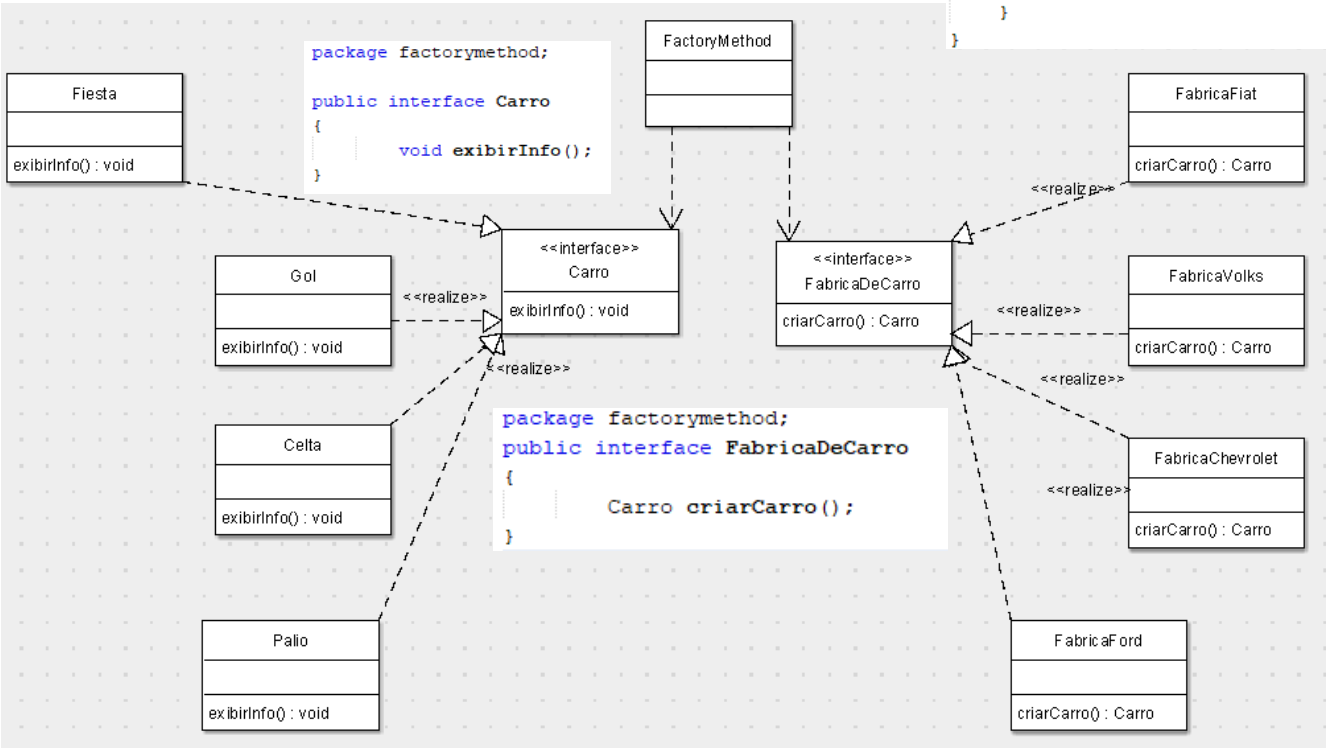
```java
package factorymethod;
public class Fiesta implements Carro
{
        @Override
        public void exibirInfo() {
                System.out.println("Modelo: Fiesta\nFabricante: Ford");
        }
}
```

```java
package factorymethod;
public class FactoryMethod {
    public static void main(String[] args) {
                FabricaDeCarro fabrica = new FabricaFiat();
                Carro carro = fabrica.criarCarro();
                carro.exibirInfo();
                System.out.println();

                fabrica = new FabricaVolks();
                carro = fabrica.criarCarro();
                carro.exibirInfo();
                System.out.println();

                fabrica = new FabricaFord();
                carro = fabrica.criarCarro();
                carro.exibirInfo();
                System.out.println();

                fabrica = new FabricaChevrolet();
                carro = fabrica.criarCarro();
                carro.exibirInfo();
    }
}
```




```java
package factorymethod;
public class FabricaFord implements FabricaDeCarro {
        @Override
        public Carro criarCarro() {
                return new Fiesta();
        }
}
```

Características da solução

- Facilidade que temos para incluir novos produtos. Não é necessário alterar NENHUM código, apenas precisamos criar o produto e a sua fábrica.
- Custo. Criamos uma estrutura grande para resolver o pequeno problema, temos um conjunto grande de pequenas classes, cada uma realizando uma operação simples.

Considerações sobre padrões de projeto

- Na página do Marcos Brizeno (https://brizeno.wordpress.com/category/padroes-de-projeto/factory-method/) no diagrama de classes, **Carro** e **FabricaDeCarro** são apresentadas como classes (e não interfaces). Na prática, para fins de aplicação desse padrão de projeto, os conceitos de Classe Abstrata e Interface são praticamente a mesma coisa, pois ambos possuem o objetivo de definir um molde a partir do qual classes concretas podem ser implementadas. O próprio exemplo poderia ser implementado com classes abstratas sem prejuízo para a aplicação do método, entretanto o uso de interfaces é preferível para diminuir o acoplamento.
- Uma outra questão sobre o exemplo são os nomes das classes de Fabrica. A priori FabricaFord ou FabricaFiat não são um problema para o exemplo apresentado, mas se tivéssemos 2 tipos de carros sendo fabricados num mesmo fabricante (FabricaFord, por exemplo iria fabricar Fiesta e EcoSport? O padrão da FabricaDeCarro é criar apenas um tipo de carro) daí teríamos um problema. Nesse contexto uma refatoração do projeto como nomes como FabricaFiesta, FabricaCelta resolveria o problema e talvez fosse uma solução melhor (entendo que por fins didáticos as Fábricas tiveram os nomes apresentados, pois isso aproveita o conceito de Fábrica do leitor).
- Notar que esse é um padrão de criação de classes pois trata de como os objetos devem ser criados e definem como classes e subclasses (ou interfaces e as classes que as implementam) devem se relacionar (estruturalmente).

Dúvidas?