



UNIVERSIDADE
ESTADUAL DE LONDRINA

Padrões de Projeto (*Design Patterns*)

Laboratório de Programação (5COP011)

Prof. Bruno Bogaz Zarpelão

Objetivo

- Discutir os principais padrões de projeto catalogados por Gamma et al. e aprender a implementá-los.

Introdução

- Projetar software orientado a objeto é muito difícil.
- Projetar software orientado a objeto que seja realmente *reusável* é mais difícil ainda.
- É necessário utilizar apropriadamente conceitos de OO como herança, interfaces, polimorfismo, etc.

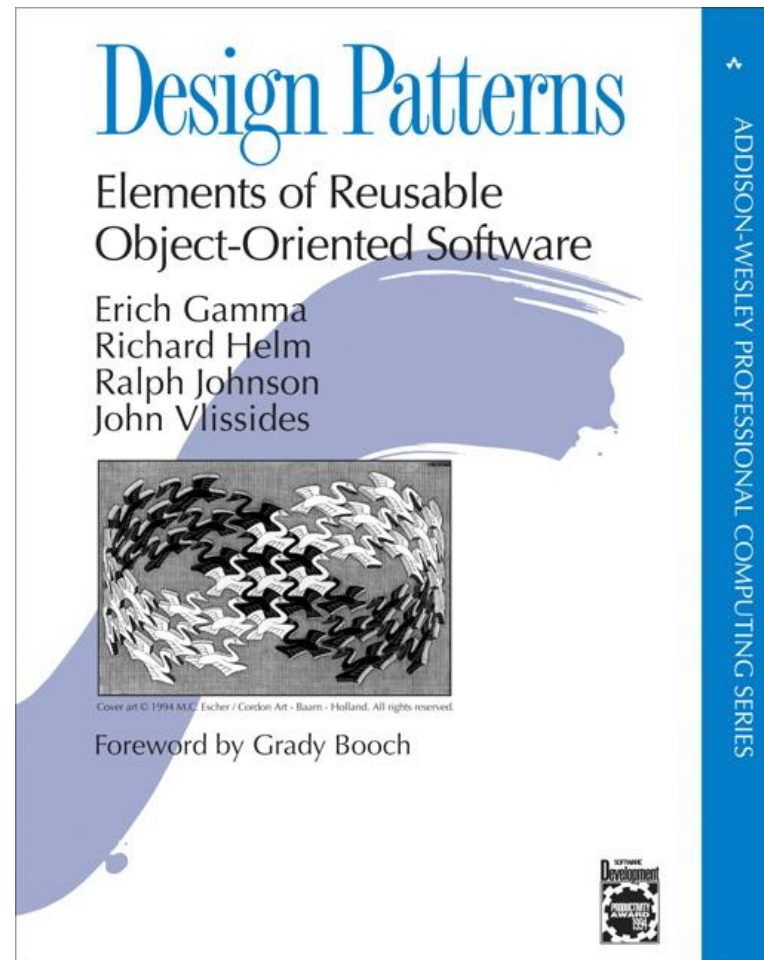
Introdução

- O problema reside em atender os requisitos de um problema específico mantendo a solução genérica o suficiente para ser reutilizada.
- Isso exige experiência, o que desenvolvedores iniciantes não tem.

Introdução

- Para atender os desenvolvedores mais jovens, os pesquisadores Erich Gamma, Richard Helm, Ralph Johnson e John Vlissides criaram um catálogo de padrões de projeto.
- Esse catálogo foi publicado na forma de um livro intitulado “Design Patterns: Elements of Reusable Object-oriented Software”.
- Desenvolver pensando nas mudanças (elas são inevitáveis).

Introdução



Padrão de projeto

- Um padrão descreve um problema que se repete frequentemente. Além disso, ele descreve uma solução que pode sempre ser aplicada para resolver esse problema.
- Gamma et al. definem os padrões de projeto de acordo com 4 propriedades: nome, problema, solução e consequências.

Padrão de projeto

- Em suma, um padrão de projeto permite que reutilizemos a solução dada por um desenvolvedor mais experiente a um problema recorrente.

Padrão de projeto

- O catálogo de padrões de projeto de Gamma et al. foi dividido em três categorias:
 - padrões comportamentais,
 - padrões de criação,
 - padrões estruturais.

Padrões comportamentais

- Os padrões comportamentais focam na distribuição de responsabilidades para os objetos.
- Dessa forma, eles facilitam a comunicação entre esses objetos, que normalmente envolve certa complexidade.
- Exemplos de padrões comportamentais: observer e template.

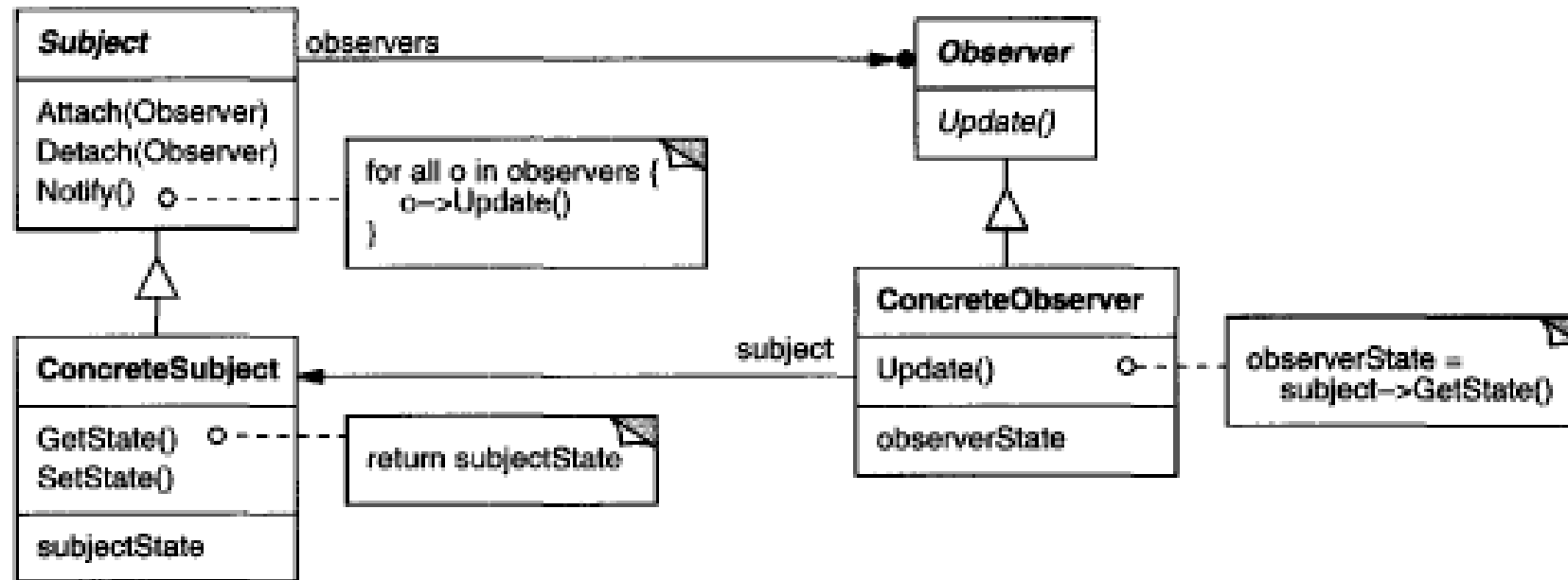
Observer

- O padrão de projeto Observer é utilizado quando precisamos manter alguns objetos atualizados de acordo com ocorrências importantes dentro do programa.
- O padrão Observer funciona em um modelo de *publisher/subscriber*.
- O objeto responsável por publicar a atualização é denominado Subject.

Observer

- Os objetos que devem ser atualizados são chamados de Observers.
- Os Observers devem se registrar no Subject para receberem as atualizações.

Observer



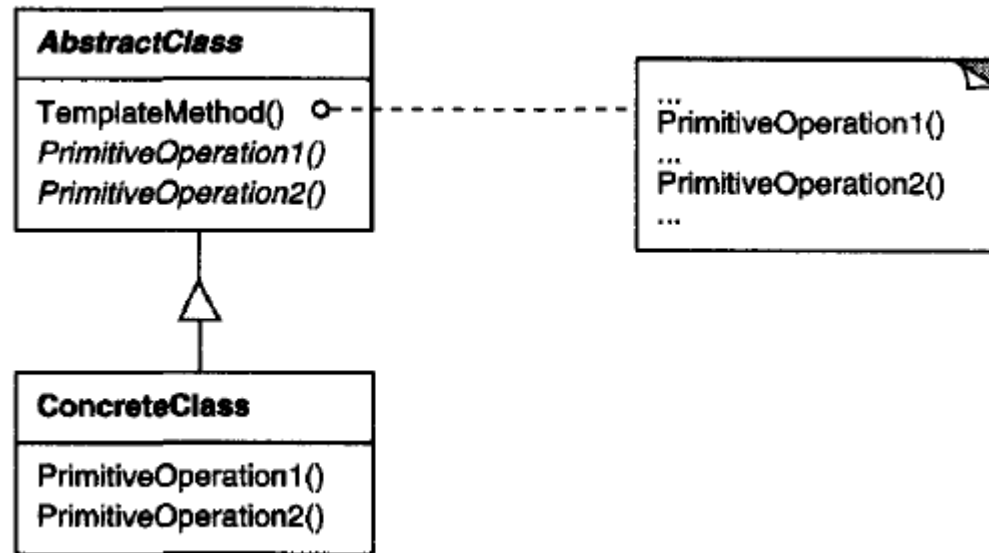
Exercício

- Implemente, utilizando o padrão Observer, um programa no qual os interessados são avisados quando uma porta da casa é aberta.

Template

- O padrão Template define o esqueleto de um algoritmo, delegando a implementação dos passos do algoritmo para sub-classes.
- Dessa forma, facilita-se a realização de modificações nas implementações dos passos do algoritmo.

Template



Exercício

- Fazer programa, usando o padrão template, que ordene uma lista de execução de músicas MP3. Cada música MP3 vai ter os seguintes dados: nome da música, artista e álbum. As músicas MP3 poderão ser ordenadas de acordo com um dos seguintes parâmetros: nome da música e artista.

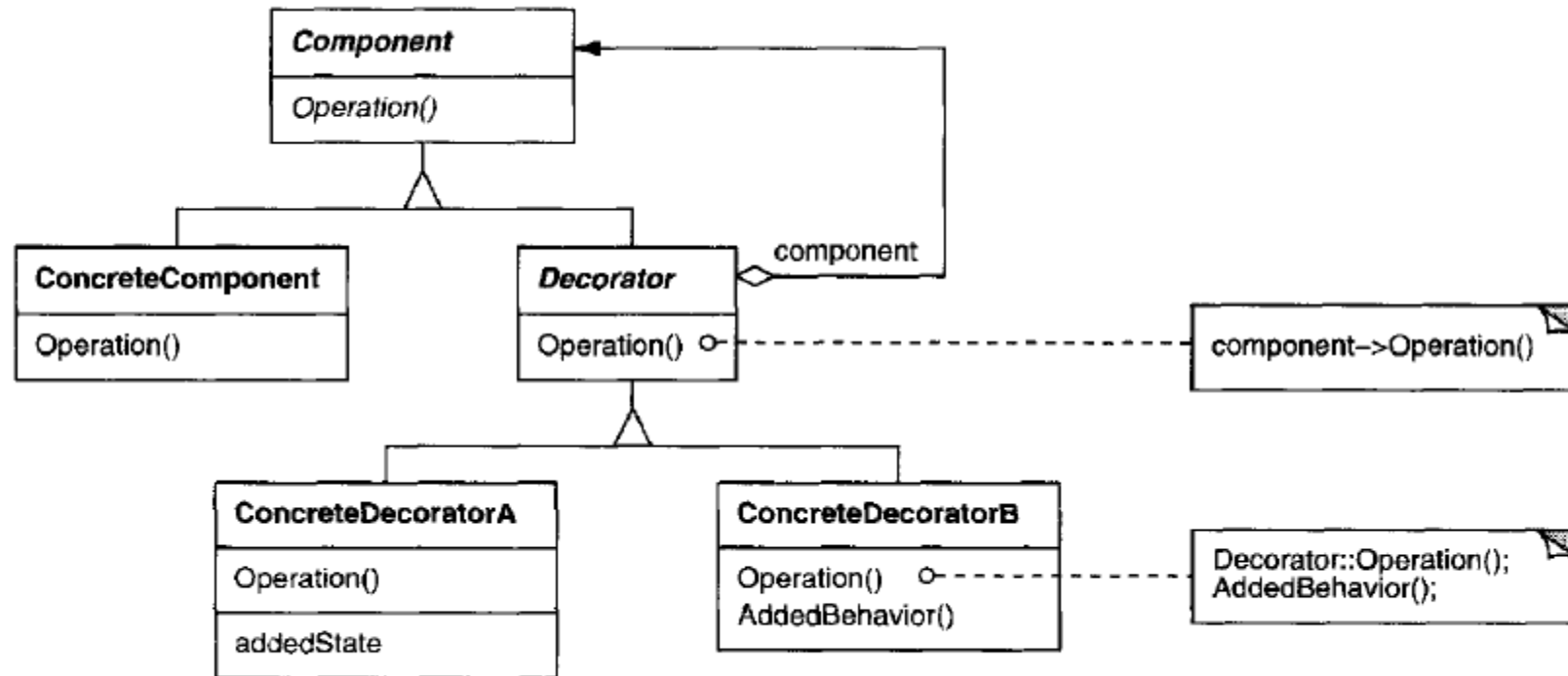
Padrões estruturais

- Padrões estruturais são relacionados à composição de classes e objetos para formar grandes estruturas.
- Normalmente, usam herança para combinar os comportamentos definidos em diferentes classes.
- É importante observar que essas estruturas são construídas de maneira que os componentes não fiquem muito acoplados.

Decorator

- O padrão Decorator pode ser utilizado quando temos um objeto com funcionalidades básicas e gostaríamos de ter à disposição outros comportamentos opcionais para serem usados em determinadas situações. No entanto, isso não deveria interferir no desenvolvimento do objeto principal.

Decorator



Exercício

- Implemente, usando o padrão Decorator, um programa para venda de sucos. O suco básico é feito apenas com água e uma fruta. Podemos adicionar leite ao suco, ao custo adicional de R\$ 1,00. Também podemos adicionar leite condensado, ao custo adicional de R\$ 2,00. O programa deve imprimir, além do preço, o nome do suco (fruta + opções).

Adapter

- O padrão Adapter normalmente é utilizado quando há alterações em um conjunto de classes que utilizamos (uma biblioteca, por exemplo) e o nosso código precisa se adaptar a essa mudança com o menor impacto possível.

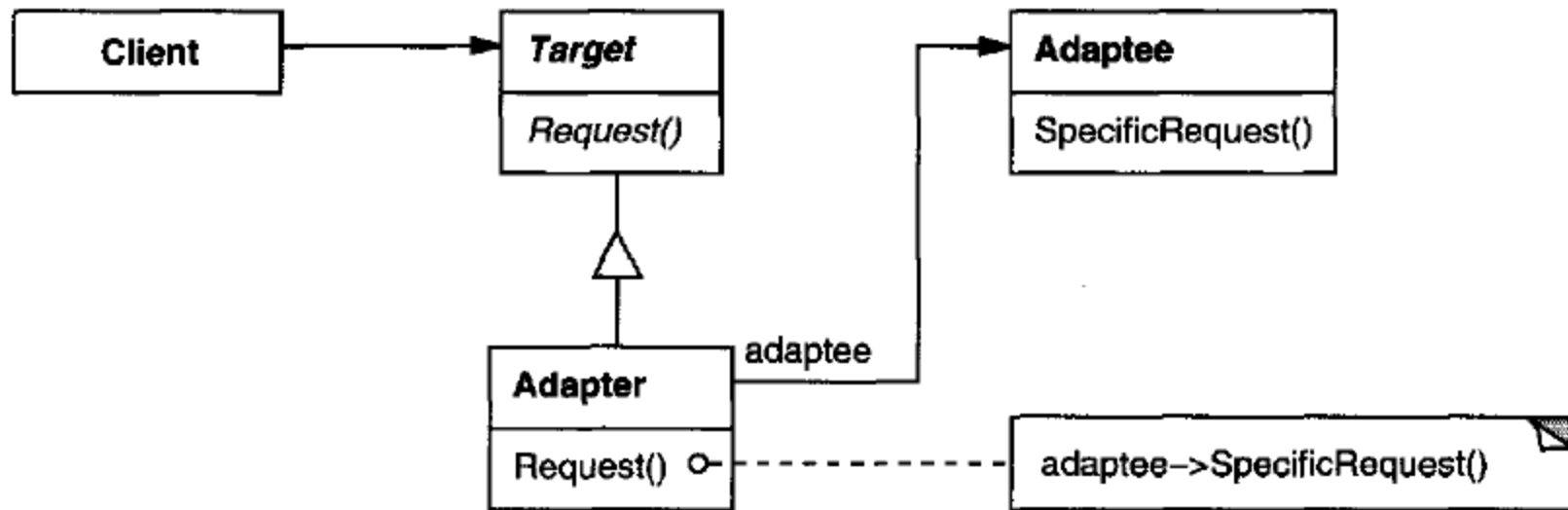
Adapter

- Por exemplo, suponha que utilizamos uma biblioteca para conexão em tomadas de dois pinos.

Adapter

- Então, o fabricante da biblioteca lança uma nova classe para tomadas com três pinos.
- A solução é usar o padrão de projeto Adapter para facilitar a adaptação.

Adapter



Exercício

- Baixe o projeto PrjAdapterExercicioInicial do Moodle.
- Nesse código, temos um registro de ponto que já estávamos acostumados a utilizar (ControleDePonto). Porém, foi disponibilizada uma nova classe (ControleDePontoNovo), para a qual será necessário adaptar a classe Principal. Use o padrão de projeto Adapter para fazer essa adaptação.

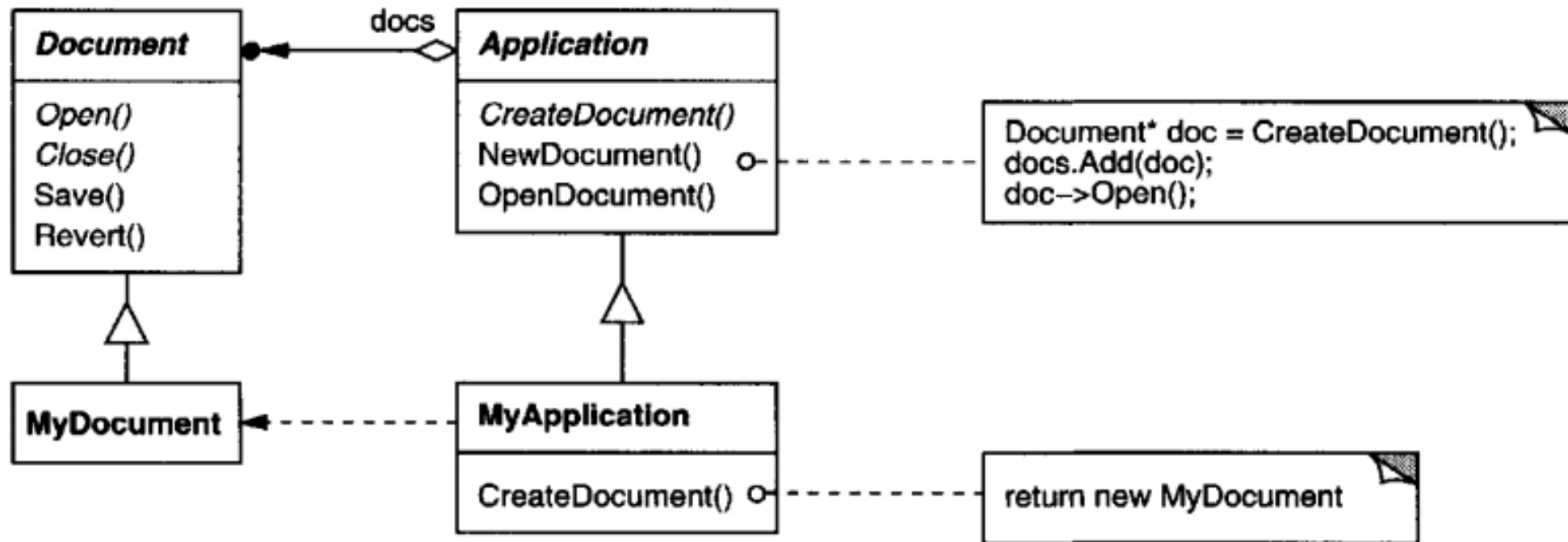
Padrões de criação

- A principal característica dos padrões de criação é atuar no processo de instanciação de objetos.
- Eles permitem que os sistemas sejam independentes de como seus objetos são criados, compostos e representados.

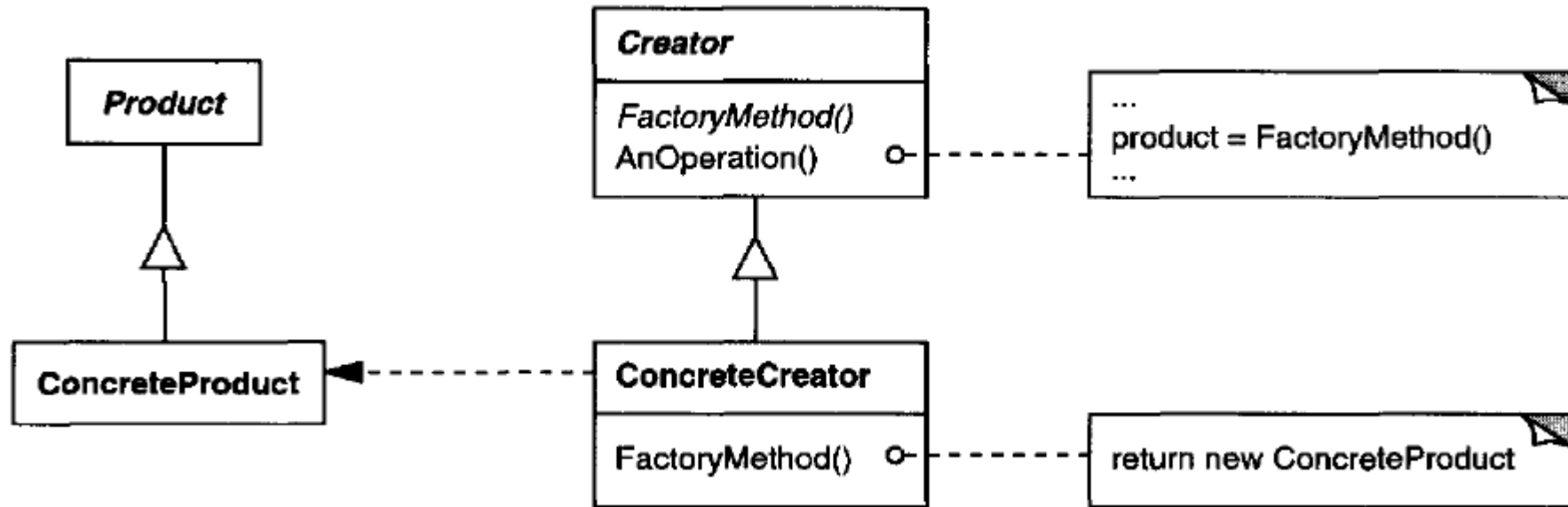
Factory

- O Factory é utilizado quando temos uma interface e queremos delegar a uma sub-classe a definição de qual será a classe concreta responsável por implementar essa interface.
- Dessa forma, o usuário da interface não tem contato direto com as implementações da interface. Ele interage com uma sub-classe responsável por instanciar as implementações.

Factory



Factory



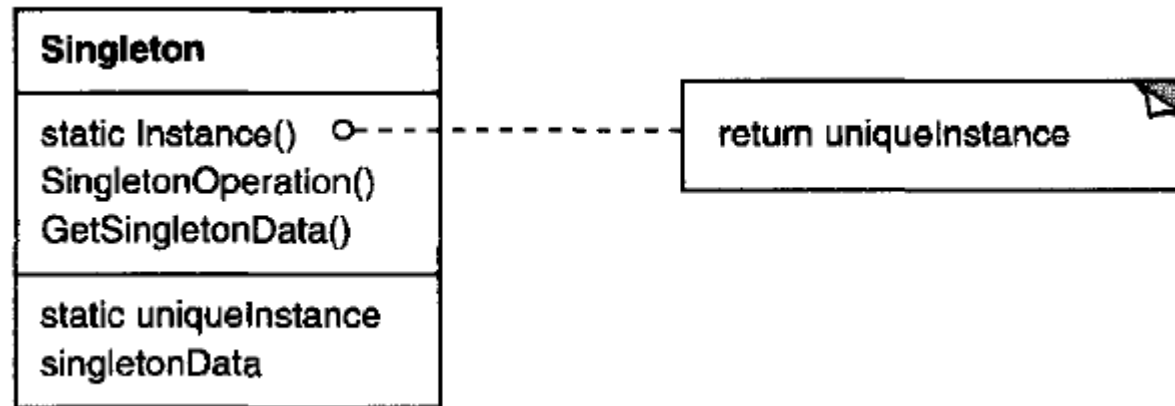
Exercício

- Usando o padrão de projeto Factory, faça um programa para uma pizzaria. Para fazer cada pizza, os seguintes passos serão executados: abrir a massa, adicionar molho, adicionar ingredientes da cobertura, assar e cortar. A maneira como esses passos são feitos depende da pizzaria e do sabor da pizza. Quando uma pizza for pedida, o programa deve imprimir na tela os passos sendo executados. O programa deve incluir diferentes pizzarias com seus sabores de pizza.

Singleton

- O padrão de projeto Singleton garante que uma classe sempre terá apenas uma instância. Além disso, ele provê um ponto de acesso global a essa instância.

Singleton



Exercício

- No Moodle, faça o download do projeto PrjSingletonExemploEnunciado. Analise o código fonte do projeto e reimplamente-o utilizando o padrão de projetos Singleton.