Design Patterns

(Padrões de Projeto)

Padrões de Projeto Criacionais

Relembrando...

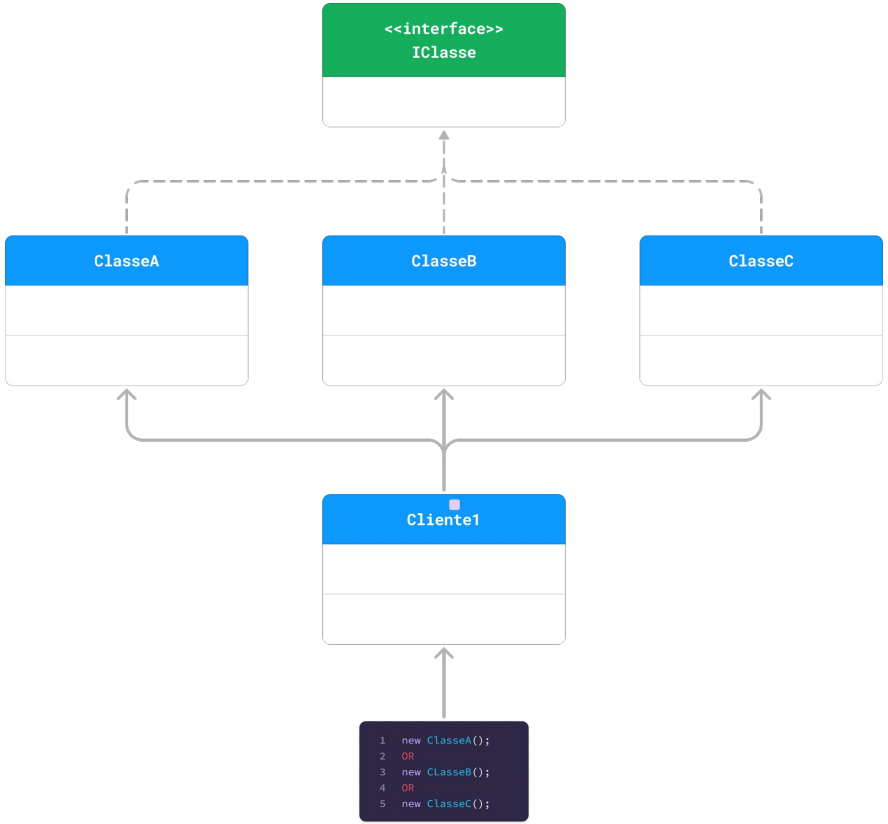
Os **padrões criacionais** são um tipo de padrão em que se concentram em abstrair e simplificar a criação de objetos. Eles ajudam a controlar o processo de instanciação de objetos de uma maneira que o código fique mais flexível e fácil de manter. Em vez de criar objetos diretamente, os padrões criacionais fornecem métodos ou mecanismos para criar instâncias de classes, permitindo maior controle sobre o processo de criação.

Factory Method

Definição

O padrão Factory Method *define uma interface para criar um objeto*, mas *permite que as subclasses possam decidir qual classe instanciar*, possibilitando que uma superclasse seja capaz de prorrogar a instanciação de uma classe para as subclasses.

Factory Method

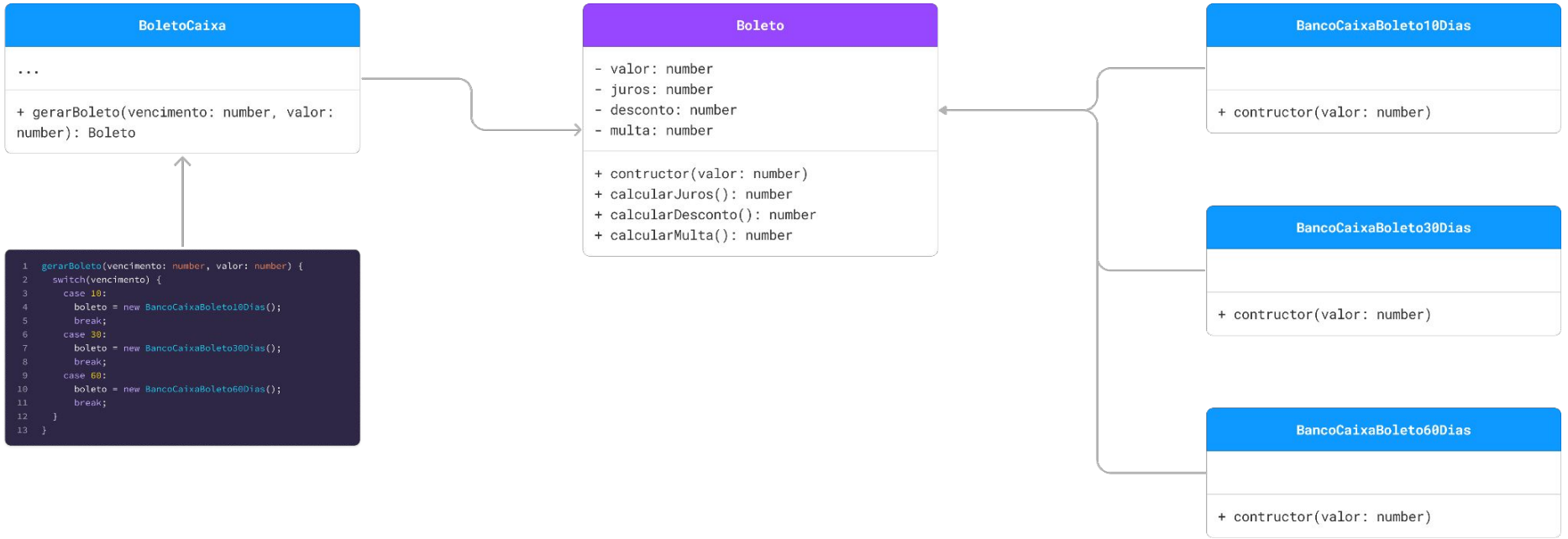


Factory Method

Para exemplificar

- Você está criando um módulo para geração de boletos em sua aplicação.
- Esses boletos são de um banco específico.
- Diferentes taxas e cálculos para cada tipo de boleto.
    - Vencimento 10 dias, Juros 2% Desconto 10% Multa 5%
    - Vencimento 30 dias, Juros 5% Desconto 5% Multa 10%
    - Vencimento 60 dias, Juros 10% Desconto 0% Multa 20%

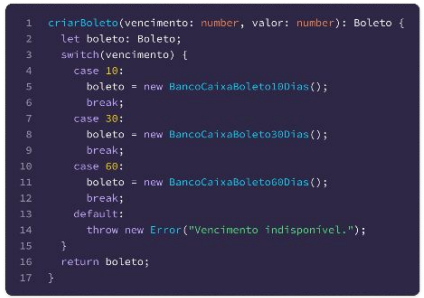
Factory Method



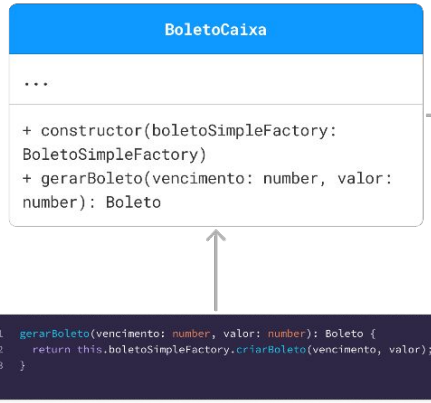
Factory Method

Show me the code!

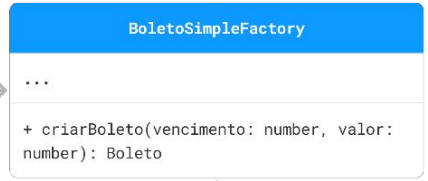
Factory Method

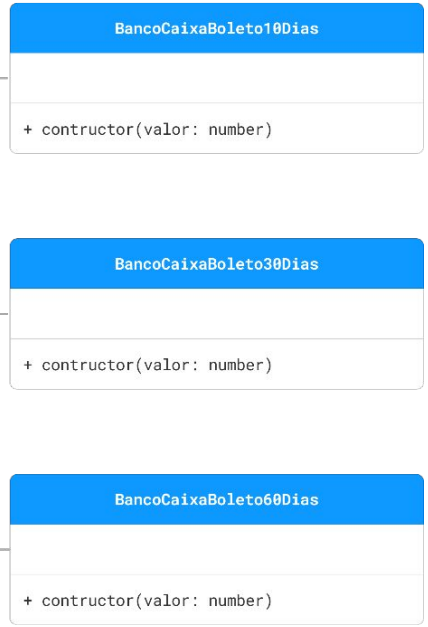


```
criarBoleto(vencimento: number, valor: number): Boleto {
  let boleto: Boleto;
  switch(vencimento) {
    case 10:
      boleto = new BancoCaixaBoleto10Dias();
      break;
    case 30:
      boleto = new BancoCaixaBoleto30Dias();
      break;
    case 60:
      boleto = new BancoCaixaBoleto60Dias();
      break;
    default:
      throw new Error("Vencimento indisponivel.");
  }
  return boleto;
}
```




```
gerarBoleto(vencimento: number, valor: number): Boleto {
  return this.boletoSimpleFactory.criarBoleto(vencimento, valor);
}
```





Factory Method

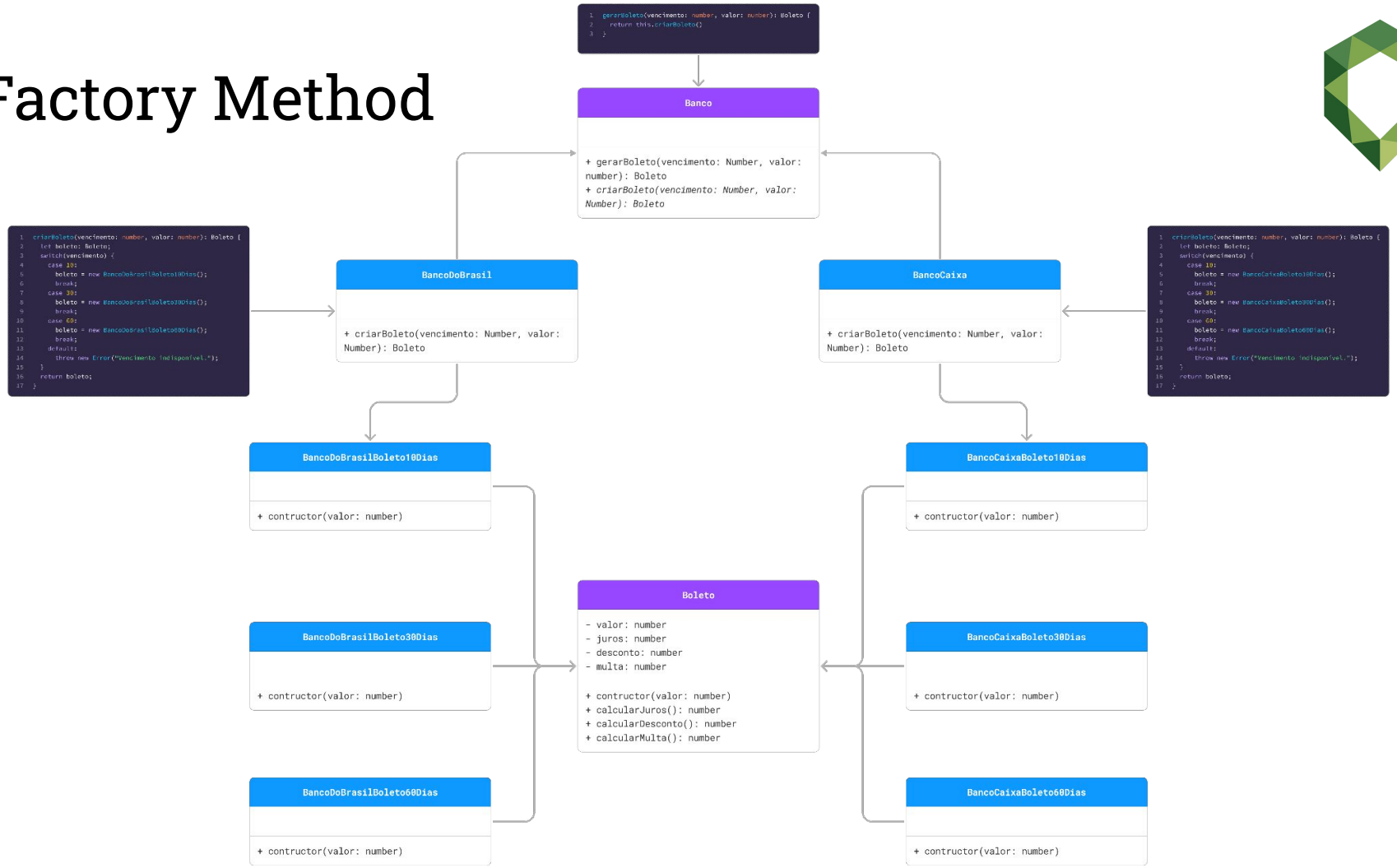
Show me the code!

Factory Method

Evolução

- Devemos evoluir nosso módulo para aceitar a geração de boletos de um novo banco.
- Diferentes taxas e cálculos para cada tipo de boleto.
    - Vencimento 10 dias, Juros 3% Desconto 5% Multa 2%
    - Vencimento 30 dias, Juros 5% Desconto 2% Multa 5%
    - Vencimento 60 dias, Juros 10% Desconto 0% Multa 15%

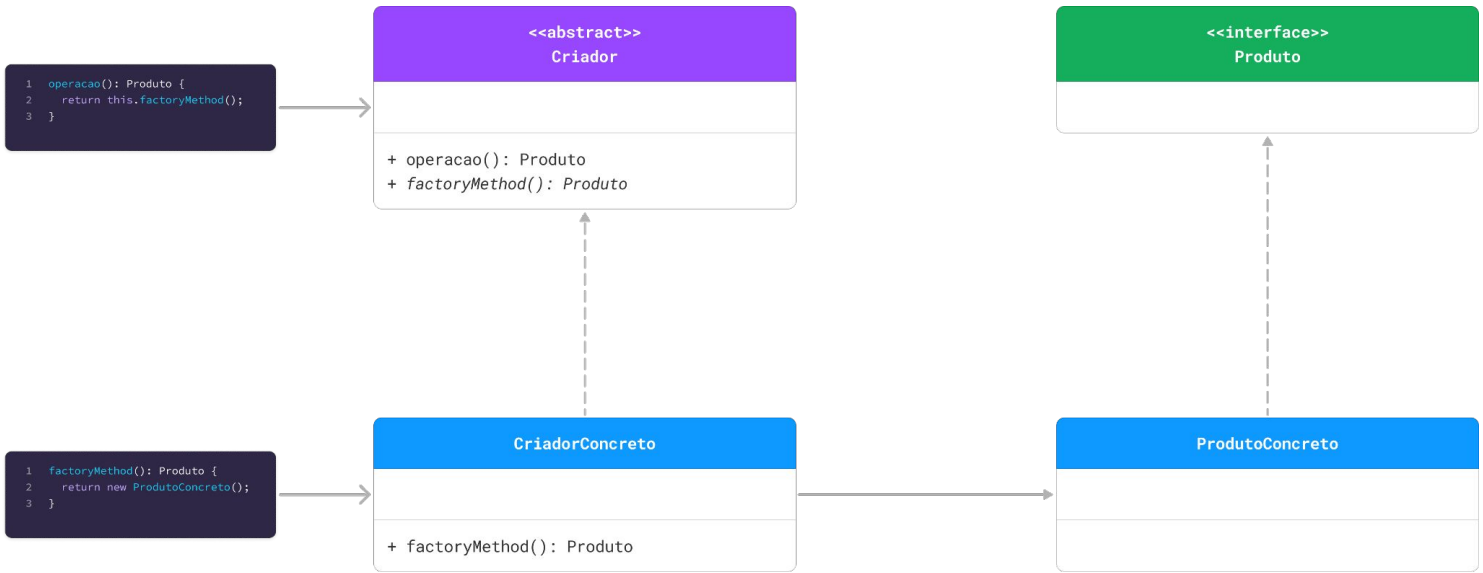
Factory Method



Factory Method

Show me the code!

Factory Method



Factory Method

Aplicabilidade

- Quando uma classe não sabe antecipar qual tipo de objeto deve criar, ou seja, entre várias classes possíveis não é possível prever qual delas deve ser utilizada.
- Quando se precisa que uma classe delegue para suas subclasses as especificações dos objetos que instanciam.
- Quando classes delegam responsabilidade a uma dentre várias subclasses auxiliares, se deseja manter o conhecimento nelas e ainda saber qual subclasse foi utilizada em determinado contexto.

Factory Method

Consequências

- O padrão Factory Method elimina o forte acoplamento entre classes concretas.
- Criar objetos dentro de uma classe com um método factoryMethod() é sempre mais flexível do que criar um objeto diretamente.
- Os clientes podem achar os métodos de fábrica úteis, e os utilizar diretamente.

Factory Method

Exercício

- Usando o Factory Method, iremos desenvolver o módulo de criação de personagem de um jogo de RPG.
- As classes jogáveis serão Guerreiro, Mago e Arqueiro.
- Cada personagem possui características específicas, como nome, força, inteligência e destreza.
- Inicialmente devemos implementar somente os métodos atacar() e defender().

Desenvolva um cliente que possibilite a instanciação de personagens de diferentes tipos usando uma Simple Factory.