Design Patterns

(Padrões de Projeto)

Chain of Responsibility

Definição

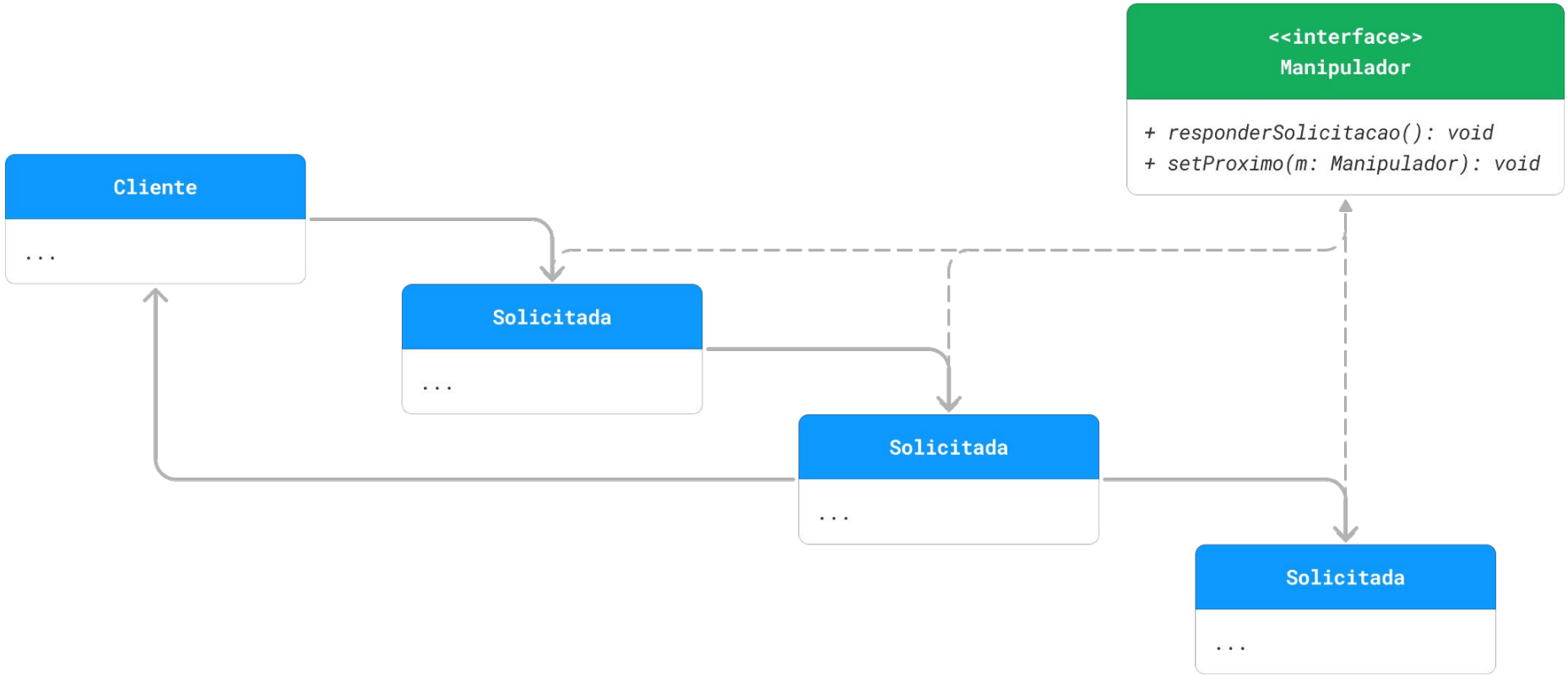
O padrão Chain of Responsibility *evita o acoplamento do remetente de uma solicitação* ao seu receptor, dando a mais de um objeto a oportunidade de tratar a solicitação. Ele encadeia os objetos receptores, *passando a solicitação ao longo da cadeia até que um objeto a trate*.

Chain of Responsibility

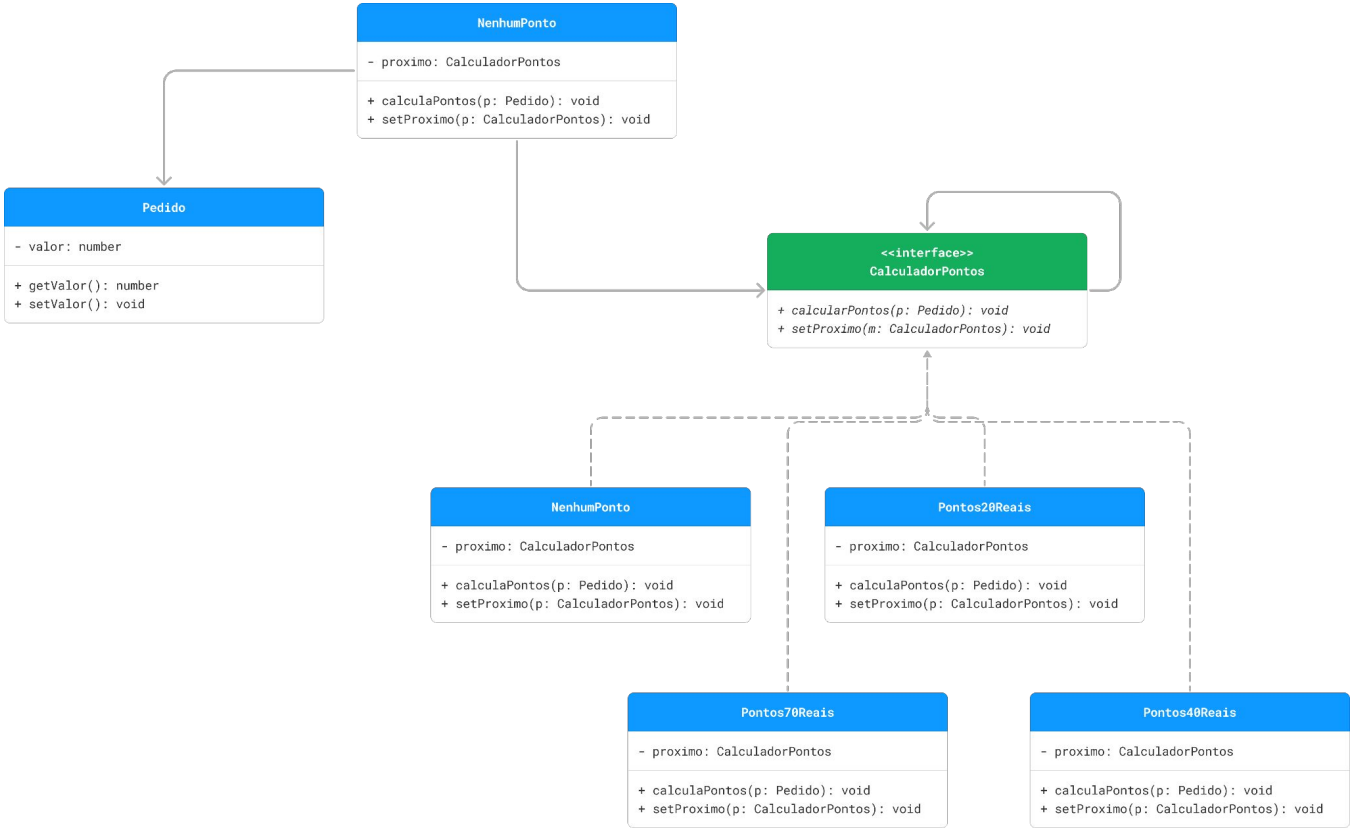

```
if                  switch:
else if             case:
else if             case:
    .       ou          .
    .                   .
    .                   .
else                default:
```

Critérios	Pontos ganhos, primeira quinzena	Pontos ganhos, segunda quinzena
Pedido acima de R$ 69,99	1 ponto a cada R$ 5,00	2 ponto a cada R$ 5,00
Pedido acima de R$ 39,99	1 ponto a cada R$ 7,00	2 ponto a cada R$ 7,00
Pedido acima de R$ 19,99	1 ponto a cada R$ 10,00	2 ponto a cada R$ 10,00
Pedido abaixo de R$ 20,00	0 pontos	0 pontos

Chain of Responsibility



Chain of Responsibility



Chain of Responsibility

Aplicabilidade

- Quando mais de uma objeto pode tratar uma solicitação e não se sabe qual objeto que fará tal tratamento. O objeto que trata a solicitação deve ser escolhido automaticamente.
- Ao ser necessário fazer uma solicitação para um dentre vários objetos sem especificar explicitamente para qual.
- Quando um conjunto de objetos que pode tratar uma solicitação deve ser especificado dinamicamente.

Chain of Responsibility

Consequências

- Redução de acoplamento.
- Maior flexibilidade na atribuição de responsabilidades aos objetos.
- A resposta da solicitação não é garantida.

Chain of Responsibility

Exercício

Usando o padrão Chain of Responsibility, implemente um sistema de validação de formulário de cadastro de usuário. O formulário tem três campos obrigatórios: nome, email, e senha. Cada campo deve ser validado de forma independente, e, caso a validação falhe em algum campo, a cadeia de validação deve ser interrompida e o erro deve ser mostrado.

As regras de validação são:

1. O nome deve ter pelo menos 3 caracteres.
2. O email deve conter um "@".
3. A senha deve ter pelo menos 8 caracteres.

Instruções:

1. Crie uma interface `Validador` com o método `setProximo(proximo: Validador): void` e `validar(dados: any): void`.
2. Implemente validadores para cada campo: `ValidadorNome`, `ValidadorEmail` e `ValidadorSenha`.
3. Configure a cadeia de validadores e teste com diferentes entradas de dados.