INSTITUTO FEDERAL
Santa Catarina

Padrões de Projeto de Software

INSTITUTO FEDERAL
Santa Catarina

Revisão

- O que vimos aula passada?

INSTITUTO FEDERAL
Santa Catarina

Histórico

- O conceito de padrões de software foi inspirado em livros de arquitetura e engenharia civil.
- Christopher Alexander encontrou temas recorrentes em arquitetura e em planejamento urbano e os capturou em descrições e instruções que ele chamou de padrões.
- Ele escreveu (1970) dois livros sobre padrões de projeto para arquitetura:
  - “The timeless way of building” (1977)
  - “A pattern language (Towns, Buildings, Construction)” (1979)
- Na década de 90, os projetistas de software se inspiraram na idéia de Alexander e a aplicaram no desenvolvimento de software.

INSTITUTO FEDERAL
Santa Catarina

O que é um padrão?

- "Cada padrão descreve um problema que ocorre repetidas vezes em nosso ambiente, e então descreve o núcleo da solução para aquele problema, de tal maneira que pode-se usar essa solução milhões de vezes sem nunca fazê-la da mesma forma duas vezes", Christopher Alexander, et. al (1977).
- "Os padrões de projeto são descrições de objetos que se comunicam e classes que são customizadas para resolver um problema genérico de design em um contexto específico", Erich Gamma, et. al (1994).

INSTITUTO FEDERAL
Santa Catarina

Por que aprender padrões?

- Aprender com a experiência dos outros.
- Aprender a programar bem com orientação a objetos.
- Desenvolver software de melhor qualidade.
- Vocabulário comum.
- Ajuda na documentação e na aprendizagem.

INSTITUTO FEDERAL
Santa Catarina

Catálogos de padrões

- Registram as experiências bem-sucedidas de um grupo de pessoas:
  - Livro “Design Patterns”:

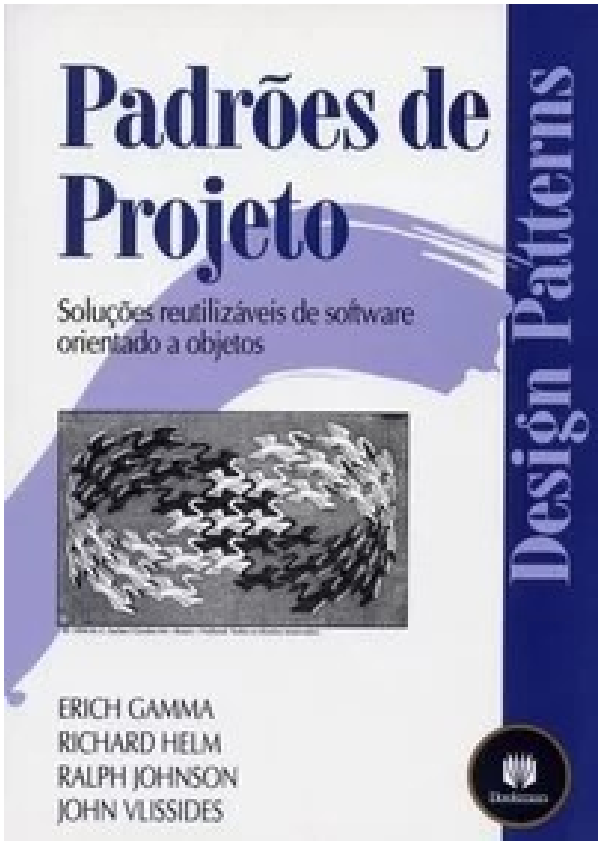
    Erich Gamma,

    Richard Helm,

    Ralph Johnson,

    John Vlissides

    (Gang of Four)



INSTITUTO FEDERAL
Santa Catarina

Categorias de Padrões do GoF

- Criacional: processo de criação de objetos.
- Estrutural: composição de classes e objetos.
- Comportamental: interação e distribuição de responsabilidades entre objetos e classes.

INSTITUTO FEDERAL
Santa Catarina

Descrição dos padrões do GoF

		Propósito		
		1. Criação	2. Estrutura	3. Comportamento
Escopo	Classe	Factory Method	Class Adapter	Interpreter Template Method
	Objeto	Abstract Factory Builder Prototype Singleton	Object Adapter Bridge Composite Decorator Facade Flyweight Proxy	Chain of Responsibility Command Iterator Mediator Memento Observer State Strategy Visitor

INSTITUTO FEDERAL
Santa Catarina

Strategy

INSTITUTO FEDERAL
Santa Catarina

Strategy

- Intenção:
  - Definir uma família de algoritmos, encapsular cada um, e fazê-los intercambiáveis.
- Permite que algoritmos mudem independentemente entre clientes que os utilizam.
- Também conhecido como Policy.

INSTITUTO FEDERAL
Santa Catarina

Motivação

- Diversos algoritmos para quebra de um texto em linhas.
- Não é desejável acoplar estes algoritmos nos clientes:
  - Se tornam maiores e mais difíceis de manter .
  - Diferentes algoritmos serão apropriados em momentos diferentes. Não queremos suportar o que não usaremos
  - É difícil adicionar novos algoritmos ou modificar os já existentes.

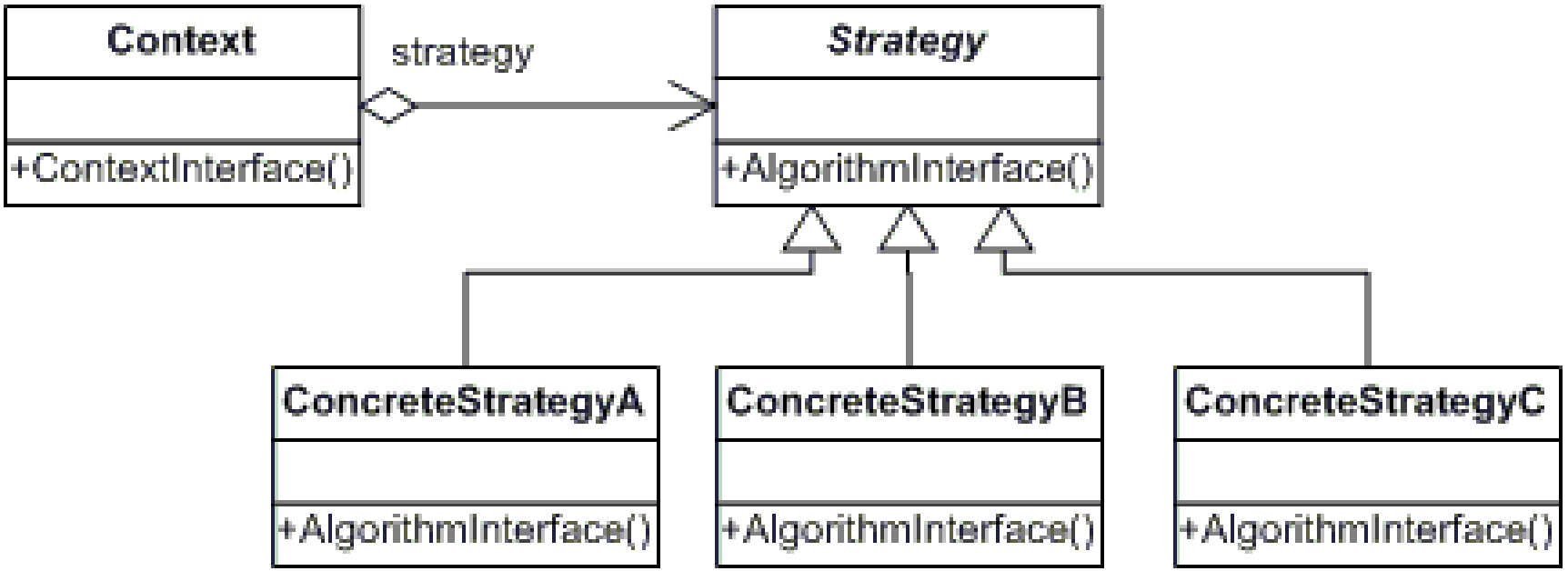
INSTITUTO FEDERAL
Santa Catarina

Aplicabilidade

- Quando muitas classes relacionadas diferem somente no seu comportamento.
- Quando é necessário utilizar diferentes variações de um algoritmo para, por exemplo, fazer o cálculo de imposto.
- Quando um algoritmo utiliza dados que não devem estar expostos aos clientes.
- Quando uma classe define múltiplos comportamentos através de sentenças condicionais em seus métodos.

INSTITUTO FEDERAL
Santa Catarina

Estrutura



INSTITUTO FEDERAL
Santa Catarina

Participantes

- Strategy:
  - Declara uma interface comum a todos os algoritmos suportados.
  - O Context utiliza esta interface para chamar um algoritmo definido por um ConcreteStrategy.
- ConcreteStrategy:
  - Implementa o algoritmo utilizando a interface Strategy.
- Context:
  - É configurado com um ConcreteStrategy.
  - Mantém uma referência para o objeto Strategy.
  - Pode definir uma interface que permite que os ConcreteStrategies acessem seus dados.

INSTITUTO FEDERAL
Santa Catarina

Colaborações

- O Strategy e o Context interagem para implementar o algortitmo escolhido:
  - O Context pode passar todos os dados necessários à execução do algoritmo, ou
  - O Context passa ele próprio como argumento da operação, permitindo que o Strategy consulte os dados necessários
- Um Context repassa requisições do cliente para o seu Strategy. O cliente geralmente cria e configura o Context com um ConcreteStrategy.
- A partir dai o cliente interage somente com o Context.

INSTITUTO FEDERAL
Santa Catarina

Consequências

- Família de algoritmos relacionados:
  - Hierarquias de classes Strategy definem uma família de algoritmos e comportamentos.
  - Herança pode ajudar a fatorar comportamento comum entre os algoritmos.
- Uma alternativa ao uso de sub-classes:
  - Poderia-se derivar o Contexto para prover diferentes implementações.
  - Entretanto, isso congela o comportamento do Context, misturando a implementação do algoritmo com o a Context.
  - Não pode variar dinamicamente.

INSTITUTO FEDERAL
Santa Catarina

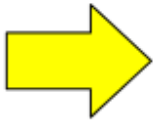
Consequências

- Os clientes escolhem entre diferentes estratégias com diferentes complexidades.
- Os clientes devem conhecer diferentes Strategies.
- Custo de comunicação entre o Strategy e o Context:
  - A interface Strategy é compartilhada por todos os ConcreteStrategy, sejam eles triviais ou complexos.
  - O Context pode criar e inicializar parâmetros que nunca serão utilizados.

Problema

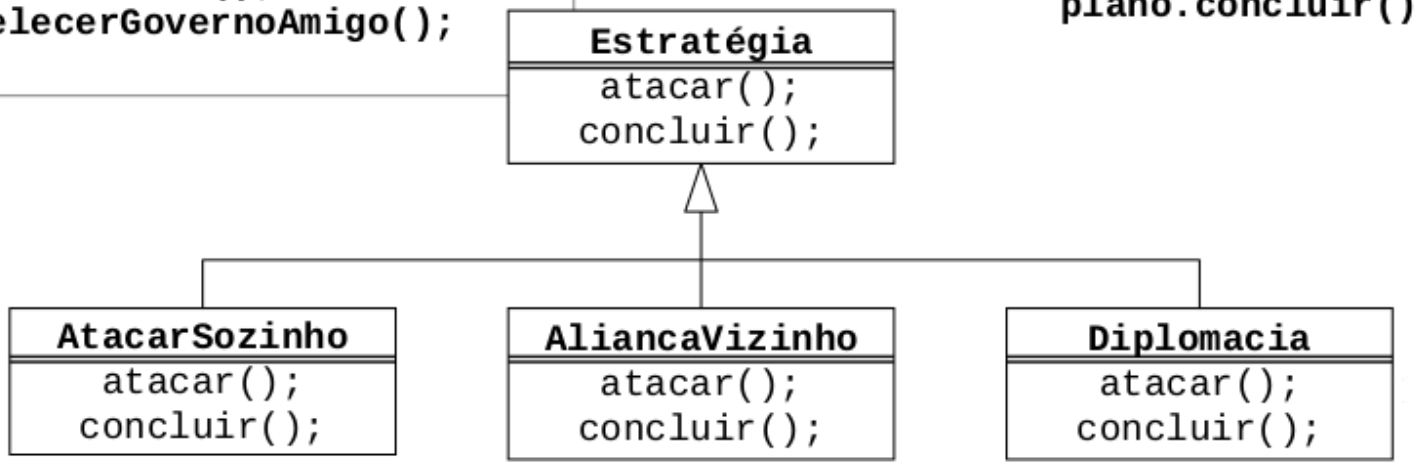
Várias estratégias, escolhidas de acordo com opções ou condições

```
if (inimigo.exercito() > 10000) {
     fazerAlianca();
     vizinhoAtacaPeloNorte();
     nosAtacamosPeloSul();
     dividirBeneficios(...);
     dividirReconstrução(...);
} else if (inimigo.isNuclear()) {
     recuarTropas();
     proporCooperacaoEconomica();
     desarmarInimigo();
} else if (inimigo.hasNoChance()) {
     plantarEvidenciasFalsas();
     soltarBombas();
     derrubarGoverno();
     estabelecerGovernoAmigo();
}
```



```
if (inimigo.exercito() > 10000) {
   plano = new AliancaVizinho();
} else if (inimigo.isNuclear()) {
   plano = new Diplomacia();
} else if (inimigo.hasNoChance())
   plano = new AtacarSozinho();
}
```

```
plano.atacar();
plano.concluir();
```



Em Java

```
public class Guerra {
   Estrategia acao;
   public void definirEstrategia() {
     if (inimigo.exercito() > 10000) {
        acao = new AliancaVizinho();
     } else if (inimigo.isNuclear()) {
        acao = new Diplomacia();
     } else if (inimigo.hasNoChance()) {
        acao = new AtacarSozinho();
     }
   }
   public void declararGuerra() {
      acao.atacar();
   }
   public void encerrarGuerra() {
      acao.concluir();
   }
}
```

```
public interface Estrategia {
   public void atacar();
   public void concluir();
}
```

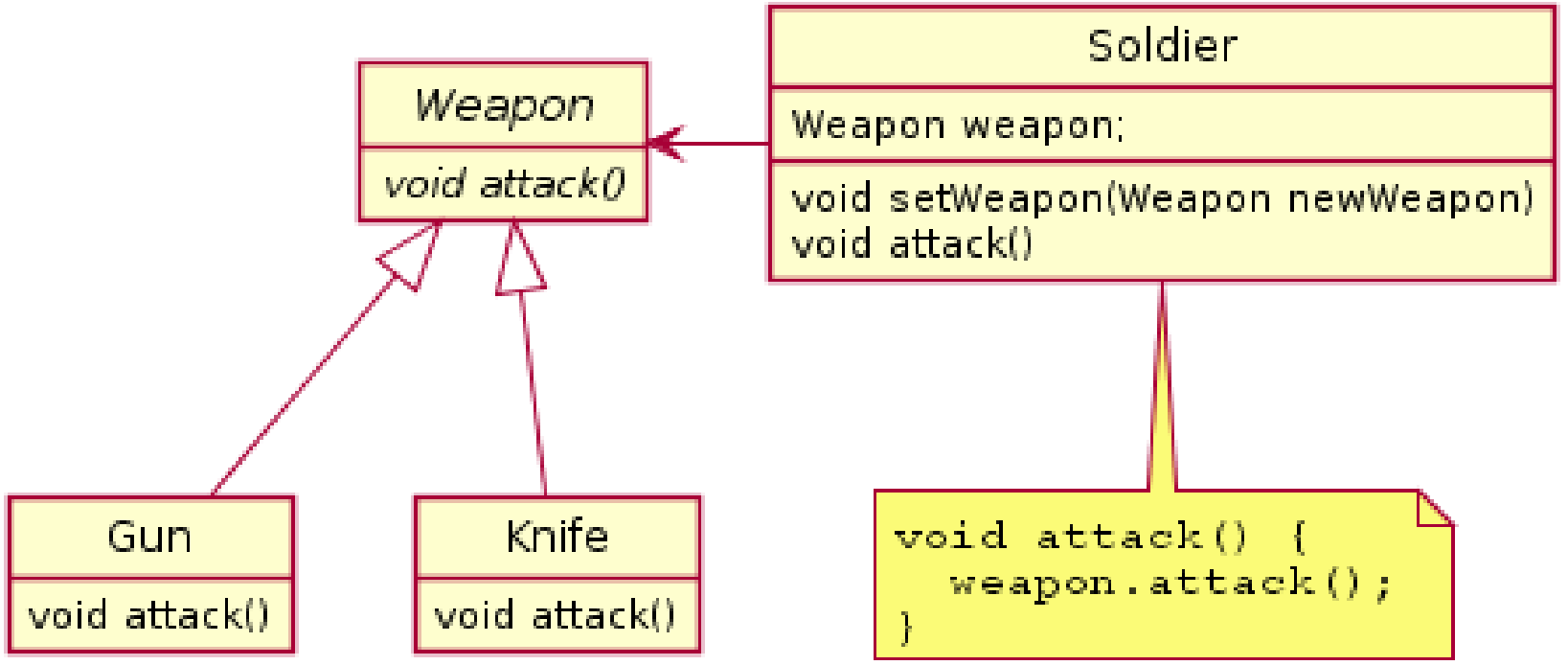
```
public class AtacarSozinho
        implements Estrategia {
   public void atacar() {
      plantarEvidenciasFalsas();
      soltarBombas();
      derrubarGoverno();
   }
   public void concluir() {
      estabelecerGovernoAmigo();
   }
}
```

```
public class AliancaVizinho
        implements Estrategia {
   public void atacar() {
      vizinhoAtacaPeloNorte();
      nosAtacamosPeloSul();
      ...
   }
   public void concluir() {
      dividirBeneficios(...);
      dividirReconstrução(...);
   }
}
```

```
public class Diplomacia
        implements Estrategia {
   public void atacar() {
      recuarTropas();
      proporCooperacaoEconomica();
      ...
   }
   public void concluir() {
      desarmarInimigo();
   }
}
```

INSTITUTO FEDERAL
Santa Catarina

Implementação



INSTITUTO FEDERAL
Santa Catarina

Implementação

```
public class Soldier{
    Weapon weapon;
    void attack(){
        weapon.attack();
}
    void setWeapon(Weapon
newWeapon){
        weapon = newWeapon;
    }
}
```

```
public interface Weapon{
    public void attack();
}
class Gun implements Weapon{
    public void attack(){   }
}
class Knife implements Weapon{
    public void attack(){   }
}
```

INSTITUTO FEDERAL
Santa Catarina

Exercício 1

Uma empresa possui um conjunto de cargos, assim para cada cargo existem regras de cálculo de imposto. Essas regras determinam a porcentagem do salário que deve ser retirada de acordo com o salário base do funcionário. As regras são:

- O Desenvolvedor deve ter um imposto de 15% caso seu salário seja maior que R$ 2000,00 e 10% caso contrário;
- O Gerente deve ter um imposto de 20% caso seu salário seja maior que R$ 3500,00 e 15% caso contrário;

Faça uma programa que calcule o valor do salário com imposto.

INSTITUTO FEDERAL
Santa Catarina

Exercício 1

```
public class Main {
    public static void main(String[] args) {
        Funcionario Funcionario1 = new Funcionario(Funcionario.DESENVOLVEDOR,2100);
        System.out.println(Funcionario1.calcularSalarioComImposto());
        Funcionario Funcionario2 = new Funcionario(Funcionario.DESENVOLVEDOR,1700);
        System.out.println(Funcionario2.calcularSalarioComImposto());
        Funcionario Funcionario3 = new Funcionario(Funcionario.GERENTE,3600);
        System.out.println(Funcionario3.calcularSalarioComImposto());
        Funcionario Funcionario4 = new Funcionario(Funcionario.GERENTE,3000);
        System.out.println(Funcionario4.calcularSalarioComImposto());
    }
}
```

INSTITUTO FEDERAL
Santa Catarina

Saída Esperada

1785.0

1530.0

2880.0

2550.0

INSTITUTO FEDERAL
Santa Catarina

Exercício 2

Faça um programa que simule um carrinho de compras com diferentes formas de pagamentos.

INSTITUTO FEDERAL
Santa Catarina

Exercício 2

```
public class Main {
    public static void main(String[] args) {
        CarrinhoDeCompras carrinho = new CarrinhoDeCompras();
        Item item1 = new Item("1234",10);
        Item item2 = new Item("5678",40);
        carrinho.addItem(item1);
        carrinho.addItem(item2);
        carrinho.comprar(new Paypal("email@examplo.com", "senha"));
        carrinho.comprar(new CartaoDeCredito("Joao", "1234567890123456", "786",
        "12/21"));
    }
}
```

INSTITUTO FEDERAL
Santa Catarina

Saída Esperada

Valor Total = 50 pago com Paypal.

Valor Total = 50 pago com cartao de credito.

INSTITUTO FEDERAL
Santa Catarina

Exercício 3

Considere o sistema de um estacionamento que precisa utilizar diversos critérios para calcular o valor que deve ser cobrado de seus clientes.

Para um veículo de passeio, o valor deve ser calculado como R$2,00 por hora, porém, caso o tempo seja maior do que 12 horas, será cobrada uma taxa diária de R$20,00 para cada dia.

Além disso, pode-se implementar regras diferentes para mensalistas, caminhões, que dependem do número de eixos e do valor da carga carregada, e para veículos para muitos passageiros, como ônibus e vans. Faça um programa para fazer o cálculo do estacionamento para os veículos de passeio.

INSTITUTO FEDERAL
Santa Catarina

Exercício 3

```
public class Main {
    public static void main(String[] args) {
        ContaEstacionamento contaEstacionamento;
        Veiculo veiculo = new Veiculo("ABC-1234",Veiculo.PASSEIO);
        contaEstacionamento = new
        ContaEstacionamento(veiculo, "08-03-2018 18:30:00", "08-03-2018 22:30:00");
        contaEstacionamento.imprimirConta();
        contaEstacionamento = new
        ContaEstacionamento(veiculo, "08-03-2018 18:30:00", "10-03-2018 19:30:00");
        contaEstacionamento.imprimirConta();
    }
}
```

INSTITUTO FEDERAL
Santa Catarina

Saída Esperada

Placa: ABC-1234

Entrada: 08-03-2018 18:30:00

Saída: 08-03-2018 22:30:00

Tempo Estacionado: 4 horas

Valor Pago: R$ 8

Placa: ABC-1234

Entrada: 08-03-2018 18:30:00

Saída: 10-03-2018 19:30:00

Tempo Estacionado: 49 horas

Valor Pago: R$ 42