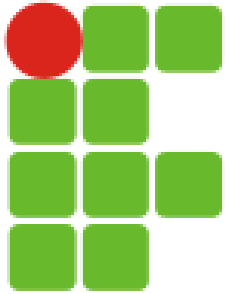


INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
FLUMINENSE

Laboratório Java

Implementação dos Conceitos Básicos da Orientação a Objetos

Renata Mesquita (renatames@gmail.com)

Agenda

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
FLUMINENSE

- Polimorfismo

Polimorfismo

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA FLUMINENSE

- Em primeiro lugar, a palavra grega "Polimorfismo" significa várias formas em português.

- Em outras palavras, o conceito de polimorfismo indica as várias formas que vários objetos se comportam mesmo sendo todos eles oriundo de uma mesma classe, resumindo uma única classe pode dar origem a vários objetos cujos os comportamentos (métodos) podem ter implementações diferentes.

Polimorfismo

- Parece estranho criar um gerente e referenciá-lo como apenas um funcionário. Por que faríamos isso? Na verdade, a situação que costuma aparecer é a que temos um método que recebe um argumento do tipo Funcionário:

```
class ControleDeBonificacoes {
    private double totalDeBonificacoes = 0;

    public void registra(Funcionario funcionario) {
        this.totalDeBonificacoes += funcionario.getBonificacao();
    }

    public double getTotalDeBonificacoes() {
        return this.totalDeBonificacoes;
    }
}
```

Polimorfismo

- Na herança, vimos que todo Gerente é um Funcionário, pois é uma extensão deste.

- Podemos nos referir a um Gerente como sendo um Funcionário.

- Se alguém precisa falar com um Funcionário do banco, pode falar com um Gerente! Porque? Pois Gerente **é um** Funcionário.

- Essa é a semântica da herança.

Polimorfismo

- Polimorfismo é a capacidade de um objeto poder ser referenciado de várias formas.

IMPORTANTE: polimorfismo não quer dizer que o objeto fica se transformando, muito pelo contrário, um objeto nasce de um tipo e morre daquele tipo, o que pode mudar é a maneira como nos referimos a ele

Polimorfismo

- Polimorfismo é a capacidade de um objeto adquirir diversas formas
- Esta capacidade decorre do mecanismo de herança
- Ao se estender uma classe, não se perde a compatibilidade com a superclasse

Polimorfismo

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA FLUMINENSE

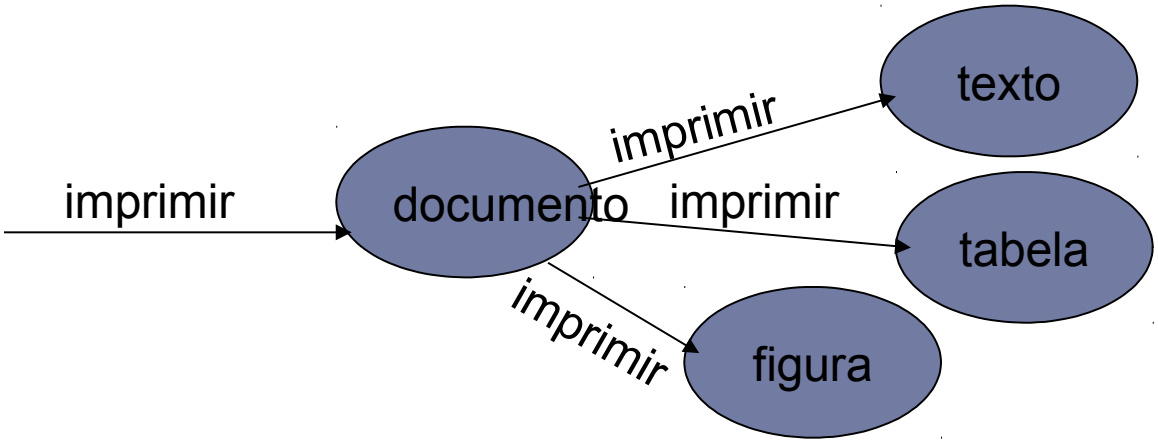
- Permite que dois ou mais objetos possam responder a uma mesma mensagem.

- Permite que um objeto comunique-se com outros através do envio de mensagens de forma consistente, sem se preocupar com as diferentes implementações da mensagem.

- Exemplo de operações polimórficas:
  - Área (de polígonos)
  - Ações como abrir, fechar, ligar
  - Imprimir

Polimorfismo: Aplicações

- Comportamento de aluno ao ouvir o toque do sinal
  - Ao tocar o sinal, alguns alunos irão embora, outros irão para a biblioteca, e outro irão para a próxima aula.
- Um documento composto de texto, tabelas, figuras e gráficos
  - Sendo comportamento ***imprimir*** polimórfico, diversos objetos saberão como imprimir-se, assim quando o usuário solicitar a impressão do documento ele simplesmente solicitará a cada um dos objetos que ele contem (texto, tabela, figura, etc) que se imprima, sem se preocupar em como o processo de impressão se dará.



Polimorfismo


```
ControleDeBonificacoes controle = new ControleDeBonificacoes();

  Gerente funcionario1 = new Gerente();
  funcionario1.setSalario(5000.0);
  controle.registra(funcionario1);

  Funcionario funcionario2 = new Funcionario();
  funcionario2.setSalario(1000.0);
  controle.registra(funcionario2);

  System.out.println(controle.getTotalDeBonificacoes());
```

Polimorfismo

- Repare que conseguimos passar um Gerente para um método que recebe um Funcionário como argumento.

- Pense como numa porta na agência bancária com o seguinte aviso: “Permitida a entrada apenas de Funcionários”.

Polimorfismo

- Quando ele receber um objeto que realmente é um Gerente, o seu método reescrito será invocado.

- Reafirmando: não importa como nos referenciamos a um objeto, o método que será invocado é sempre o que é dele.

- No dia em que criarmos uma classe Secretaria, por exemplo, que é filha de Funcionario, precisaremos mudar a classe de ControleDeBonificacoes? Não. Basta a classe Secretaria reescrever os métodos que lhe parecerem necessários.

Polimorfismo

- É exatamente esse o poder do polimorfismo, juntamente com a herança e reescrita de método: diminuir acoplamento entre as classes, para evitar que novos códigos resultem em modificações em inúmeros lugares.

- Repare que quem criou ControleDeBonificacoes pode nunca ter imaginado a criação da classe Secretaria ou Engenheiro.

- Isso traz um reaproveitamento enorme de código.

Polimorfismo
Exemplo

- Imagine que vamos modelar um sistema para a faculdade que controle as despesas com funcionários e professores. Nosso funcionário fica assim:

```
class EmpregadoDaFaculdade {
    private String nome;
    private double salario;
    double getGastos() {
        return this.salario;
    }
    String getInfo() {
        return "nome: " + this.nome + " com salário " + this.salario;
    }
     // métodos de get, set e outros
}
```

Polimorfismo
Exemplo

- O gasto que temos com o professor não é apenas seu salário. Temos de somar um bônus de 10 reais por hora/aula.

- O que fazemos então? Reescrevemos o método. Assim como o getGastos é diferente, o getInfo também será, pois temos de mostrar as horas/aula também.

Polimorfismo
Exemplo


```
class ProfessorDaFaculdade extends EmpregadoDaFaculdade {
    private int horasDeAula;
    double getGastos() {

        return this.getSalario() + this.horasDeAula * 10;
    }
    String getInfo() {
        String informacaoBasica = super.getInfo();
        String informacao = informacaoBasica + " horas de aula: " + this.horasDeAula;
        return informacao;
    }
    // métodos de get, set e outros
}
```

Polimorfismo
Exemplo

- Uso da palavra chave super.

- Apesar do método ter sido reescrito, gostaríamos de acessar o método da classe mãe, para não ter de copiar e colocar o conteúdo desse método e depois concatenar com a informação das horas de aula.

Polimorfismo
Exemplo

- Como tiramos proveito do polimorfismo? Imagine que temos uma classe de relatório:

```
class GeradorDeRelatorio {
    public void adiciona(EmpregadoDaFaculdade f) {
        System.out.println(f.getInfo());
        System.out.println(f.getGastos());
    }
}
```

- Podemos passar para nossa classe qualquer EmpregadoDaFaculdade! Vai funcionar tanto para professor, quanto para funcionário comum.

Polimorfismo
Exemplo

- Um certo dia, muito depois de terminar essa classe de relatório, resolvemos aumentar nosso sistema, e colocar uma classe nova, que representa o Reitor.

- Como ele também é um EmpregadoDaFaculdade, será que vamos precisar alterar alguma coisa na nossa classe de Relatorio? Não. Essa é a idéia!

- Quem programou a classe GeradorDeRelatorio nunca imaginou que existiria uma classe Reitor e, mesmo assim, o sistema funciona.

Polimorfismo
Exemplo

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA FLUMINENSE

```
class Reitor extends ProfessorDaFaculdade {
      // informações extras
     String getInfo() {
          return super.getInfo() + " e ele é um reitor";
     }
      // não sobreescrevemos o getGastos!!!
}
```

Bibliografia

- JAVA e Orientação a Objetos – Caelum Ensino e Soluções em JAVA