Programação Orientada a Objetos
Polimorfismo

Alexandre Mello

Fatec Campinas

2024

Roteiro

Introdução

- Polimorfismo significa "muitas formas"
- Permite ao programador usar o mesmo elemento de formas diferentes

- Polimorfismo denota uma situação na qual um objeto pode se comportar de maneiras diferentes ao receber uma mensagem

Exemplo

```
public class Figura {
   protected double area;

   public Figura() {
      area = 0;
   }

   public void calcularArea() {
   }
}
```

Exemplo

```
public class Retangulo extends Figura {
   private double largura, altura;

   public Retangulo(double l, double a) {
      super();
      largura = l;
      altura = a;
   }

   public void calcularArea() {
      area = largura * altura
      System.out.println("Área do retângulo = " + area);
   }
}
```

Exemplo

```
public class Circulo extends Figura {
   private double raio;

   public Circulo(double r) {
      super();
      raio = r;
   }

   public void calcularArea() {
      area = 3.14 * raio * raio;
      System.out.println("Área do circulo = " + area);
   }
}
```

Exemplo

```
public class Exemplo {
   public statis void main(String[] args) {
      Circulo c = new Circulo(5);
      c.calcularArea();

      Retangulo r = new Retangulo(6, 13);
      r.calcularArea();
   }
}
```

O método `calcularArea()` é polimórfico

Exemplo

```
public class Exemplo {
   public statis void main(String[] args) {
      Figura f;

      f = new Circulo(5);
      f.calcularArea();

      f = new Retangulo(6, 13);
      f.calcularArea();
   }
}
```

A variável f é polimórfica

Exemplo

```
public class Desenho {
   public void usaFiguras(Figura fig) {
      fig.calcularArea()
   }
}

public class Exemplo {
   public statis void main(String[] args) {
      Circulo c = new Circulo(5)
      c.calcularArea();

      Retangulo r = new Retangulo(6, 13);
      r.calcularArea();

      Desenho d = new Desenho();
      d.usaFiguras(c);
      d.usaFiguras(r);
   }
}
```

Exemplo

```
public class Exemplo {
   public statis void main(String[] args) {
      Figura figs[] = new Figura[2];

      figs[0] = new Circulo(5);
      figs[1] = new Retangulo(6, 13);

      for(Figuras f : figs) {
         System.out.println("Area = " + f.calcularArea());
      }
   }
}
```

Exercício

Complete o código Java abaixo adicionando os elementos na lista `animais` (um item para cada tipo de animal) e escreva as classes para os animais Arara, Urso e Cobra.

```
public class Animal {
   public void andar() {
      System.out.println("Eu ando como um Animal genérico.");
   }

   public static void main(String[] args) {
      List<Animal> animais = new LinkedList<Animal>();
      . . .
      for (Animal a : animais) {
         a.andar();
      }
   }
}
```

Estático versus Dinâmico

Polimorfismo estático

- Relacionado à sobrecarga de um método
- Erros são resolvidos em tempo de compilação

```
public class Estatico {
   public int soma(int a, int b) { return a + b; }

   public float soma(float a, float b) { return a + b; }

   public int soma(int a, int b, int c) { return a + b + c; }
}
```

Estático versus Dinâmico

Polimorfismo dinâmico

- Relacionado à sobrescrita de um método
- É determinado qual método será chamado em tempo de execução

```
class X {
   public int soma(int a, int b) { return a + b; }
}

class Y extends X{
   public int soma(int a, int b) { return a + b + 1; }
}
```

Exercício

Lista 3 – Polimorfismo