



Universidade Campus Joinville - SC

Análise e Desenvolvimento de Sistemas

Polimorfismo e Herança

Profº Diego Madureira e Silvio Souza

Aluno: Nivea Souza

Joinville - SC
2025

Herança

A herança, na Programação Orientada a Objetos, permite que uma classe reutilize características e comportamentos de outra classe já existente. Dessa forma, o código é centralizado, evita duplicações e facilita a manutenção, já que atributos e métodos comuns ficam definidos em um único lugar.

```
4  class Pessoa
5  {
6
7      public $nome;
8      public $idade;
9      public $altura;
10
11     function falar() {}
12     function andar() {}
13     function respirar() {}
14 }
15
16 $pessoa1 = new Pessoa();
17 $pessoa1 -> nome = "nivea";
18 $pessoa1 -> idade = 24;
19 $pessoa1 -> altura = 1.55;
20
21
22 var_dump($pessoa1);
23
24 class Aluno extends Pessoa{
25     public $turma;
26
27     function estudar(){
28         echo "estudando...";
29     }
30 }
```

Figura 1 – Captura de tela representando Herança. Nivea (2025)

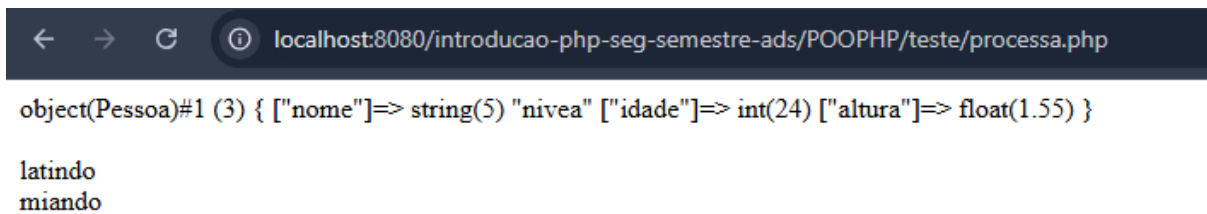
Neste exemplo, a partir da classe Pessoa foi criada a classe Aluno, que é uma extensão de Pessoa. Isso significa que Aluno herda os atributos e métodos da classe Pessoa, mas também pode ter novos métodos próprios, que não são herdados pela classe original.

Polimorfismo

O polimorfismo, na programação orientada a objetos, permite que métodos com o mesmo nome se comportem de formas diferentes em classes distintas. Dessa forma, é possível adaptar o comportamento de um método herdado para se encaixar no contexto específico de cada objeto.

```
37  class Cachorro{
38      public function som(){
39          echo "latindo";
40      }
41  }
42  class Gato extends Cachorro{
43      public function som(){
44          echo "miando";
45      }
46  }
47  $cachorro1 = new Cachorro();
48  $gato1 = new Gato();
49
50
51  echo "<br>";
52  echo "<br>";
53  $cachorro1->som();
54  echo "<br>";
55  $gato1->som();
56
```

Figura 7 – Captura de tela representando polimorfismo. Nivea (2025)



A screenshot of a web browser window. The address bar shows the URL: localhost:8080/introducao-php-seg-semester-ads/POOPHP/teste/processa.php. The main content area displays the output of a PHP script. The first line is a PHP var_dump output for an object: object(Pessoa)#1 (3) { ["nome"]=> string(5) "nivea" ["idade"]=> int(24) ["altura"]=> float(1.55) }. Below this, the method latindo is called, and then the method miando is called.

```
object(Pessoa)#1 (3) { ["nome"]=> string(5) "nivea" ["idade"]=> int(24) ["altura"]=> float(1.55) }  
  
latindo  
miando
```

Figura 2 – Captura de tela com o resultado do exemplo de polimorfismo em POO PHP. Nivea (2025)

Na sobreposição de métodos, um método da classe filha tem o mesmo nome, os mesmos parâmetros e os mesmos tipos de retorno que o método da classe pai, permitindo que o comportamento seja adaptado ao contexto da classe filha.