

Orientação a Objetos com PHP

Wendell Bento Geraldes

IFGO

6 de novembro de 2018

Sumário

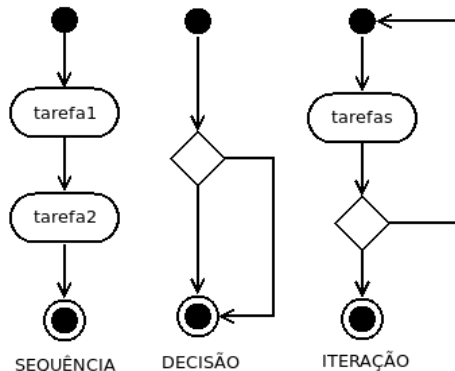
- 1 Programação Estruturada
- 2 Orientação a Objetos
- 3 Construtores e destrutores
- 4 Herança
- 5 Polimorfismo
- 6 Referências bibliográficas

Orientação a Objetos com PHP

Programação Estruturada

A programação estruturada é um paradigma de programação que introduziu uma série de conceitos importantes, é baseada fortemente na modularização, cuja idéia é dividir o programa em unidades menores conhecidas por procedimentos ou funções. Na programação estruturada, as unidades do código (funções) se interligam por meio de três mecanismos básicos: sequência, decisão e iteração. (DALL'OGGIO, 2007)

Figura: Programção Estruturada



Orientação a Objetos com PHP

Orientação a Objetos

Ao trabalharmos com orientação a objetos é fundamental entender o conceito de classes e objetos. Uma classe é uma estrutura que define um tipo de dados, podendo conter atributos (variáveis) e também funções (métodos) para manipular esses atributos. (DALL'OGGIO, 2007)

Código-Fonte 1: Produto.class.php

```
<?php
class Produto {
    var $Codigo;
    var $Descricao;
    var $Preco;
    var $Quantidade;
}
?>
```

Orientação a Objetos com PHP

Orientação a Objetos

Um objeto contém exatamente a mesma estrutura e as propriedades de uma classe, no entanto sua estrutura é dinâmica, seus atributos podem mudar de valor durante a execução do programa e podemos declarar diversos objetos oriundos de uma mesma classe. (DALL'OGGIO, 2007)

Código-Fonte 2: objeto.php

```
<?php
include_once 'classes/Produto.class.php'; //insere a classe
$produto = new Produto; //cria um objeto
$produto->Codigo = 4001; //atribuir valores
$produto->Descricao = 'CD - The Best of Eric Clapton';
echo $produto;
?>
```

Orientação a Objetos com PHP

Orientação a Objetos

Um objeto não foi feito para ser impresso diretamente, mas suas propriedades sim. Entretanto se tentarmos imprimir um objeto diretamente, o PHP retornará o identificador interno deste objeto na memória. Reescreveremos, então, a classe **Produto** para adicionar uma funcionalidade ou um método, que é uma função declarada dentro da estrutura da classe, agindo dentro deste escopo. O método criado, **ImprimeEtiqueta()**, trabalha com as propriedades do objeto. Afim de diferenciar as propriedades de um objeto de variáveis locais, utiliza-se a pseudovariável **\$this** para representar o objeto atual e acessar suas propriedades. (DALL'OGGIO, 2007)

Orientação a Objetos com PHP

Orientação a Objetos

Código-Fonte 3: Produto.class.php

```
<?php
class Produto {
    var $Codigo;
    var $Descricao;
    var $Preco;
    var $Quantidade;
    function ImprimirEtiqueta() {
        print 'Codigo: ' . $this->Codigo."\\n";
        print 'Descricao: ' . $this->Descricao."\\n";
    }
}
?>
```

Orientação a Objetos com PHP

Orientação a Objetos

Código-Fonte 4: objeto.php

```
<?php
include_once 'classes/Produto.class.php'; //insere a classe
$produto1 = new Produto; //cria um objeto
$produto2 = new Produto; //cria outro objeto
$produto1->Codigo = 4001; //atribuir valores
$produto1->Descricao = 'CD - The Best of Eric Clapton';
$produto2->Codigo = 4002; //atribui valores
$produto2->Descricao = 'CD - The Best of Black Sabbath';
$produto1->ImprimeEtiqueta(); //imprime informacoes da etiqueta
$produto2->ImprimeEtiqueta();
?>
```


Orientação a Objetos com PHP

Construtores e destrutores

Um construtor é um método especial utilizado para definir o comportamento inicial de um objeto, ou seja, o comportamento no momento de sua criação. O método construtor é executado automaticamente no momento em que instanciamos um objeto por meio do operador `new`. (DALL'OGGIO, 2007)

Um destrutor ou finalizador é um método especial executado automaticamente quando o objeto é deslocado da memória, quando atribuímos o valor `NULL` ao objeto, quando utilizamos a função `unset()` sobre o mesmo ou, ainda quando o programa é finalizado. (DALL'OGGIO, 2007)

Orientação a Objetos com PHP

Construtores e destrutores

Código-Fonte 5: Pessoa.class.php

```
<?php
class Pessoa {
    var $Nome;
    var $Idade;
    function __construct($Nome, $Idade) { //construtor
        $this->Nome = $Nome;
        $this->Idade = $Idade;
    }
    function __destruct () { //destrutor
        echo "Objeto {$this->Nome} finalizado...\n";
    }
}

?>
```

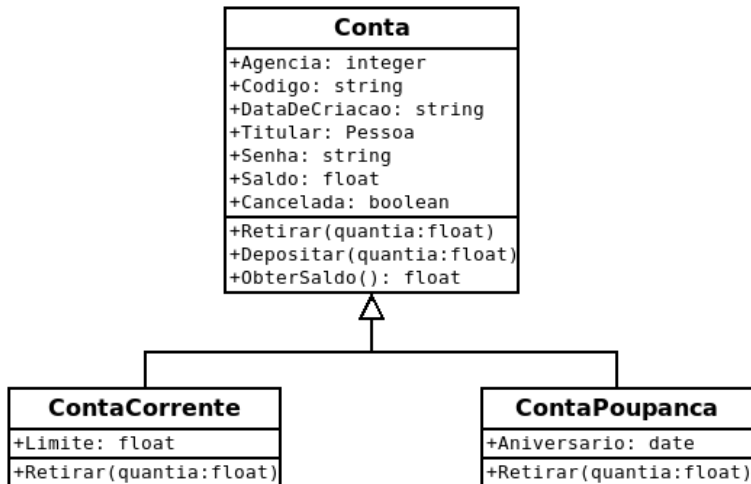
Orientação a Objetos com PHP

Herança

O que devemos levar em consideração sobre herança em orientação a objetos é o compartilhamento de atributos e comportamentos entre classes de uma mesma hierarquia (árvore). As classes inferiores da hierarquia automaticamente herdam todas as propriedades e os métodos das classes superiores, chamadas de superclasses. Utilizando a herança, em vez de criarmos uma estrutura totalmente nova (uma classe), podemos reaproveitar uma estrutura já existente que nos forneça uma base abstrata para o desenvolvimento, provendo recursos básicos e comuns. (DALL'OGGIO, 2007)

Diagrama de Classes

Exemplo de Herança



Orientação a Objetos com PHP

Herança

Código-Fonte 6: Conta.class.php

```
<?php
    //SuperClasse ou Classe Mae
    class Conta
    {
        //Atributos
        //Construtor
        //Metodos
    }
?>
```

Orientação a Objetos com PHP

Herança

Código-Fonte 7: ContaPoupanca.class.php

```
<?php
    //SubClasses ou Classes filhas
    class ContaPoupanca extends Conta
    {
        //Atributos
        //Construtor
        //Metodos
    }
?>
```

Orientação a Objetos com PHP

Herança


Código-Fonte 8: ContaCorrente.class.php

```
<?php
    //SubClasses ou Classes filhas
    class ContaCorrente extends Conta
    {
        //Atributos
        //Construtor
        //Metodos
    }
?>
```

Orientação a Objetos com PHP

Polimorfismo

O significado da palavra polimorfismo nos remete a "muitas formas". Polimorfismo em orientação a objetos é o princípio que permite que classes derivadas de uma mesma superclasse tenham métodos iguais (com a mesma nomenclatura e parâmetros), mas comportamentos diferentes, redefinidos em cada uma das classes. (DALL'OGGIO, 2007)

 DALL'OGGIO, P. **PHP: programando com orientação a objetos**. 1. ed. São Paulo: Novatec Editora, 2007. 580 p.

 NIEDERAUER, J. **Desenvolvendo Websites com PHP**. 2. ed. São Paulo: Novatec Editora, 2011. 301 p.