**Padrão para Nomenclatura de PHP**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sumário de Informações do Documento** | | | |
| Versão | Data | Mudanças | Autor |
| 0.1 | 31/05/2013 | Fonte: <http://www.webartz.com.br/desenvolvimento-web/php/php-orientado-a-objetos-conceitos-nomenclatura/> | Roberto Soares Bochnia |

Sumário

[Apresentação 2](#_Toc357758760)

[Introdução 2](#_Toc357758761)

[Nomenclatura de Estruturas 2](#_Toc357758762)

[Importante 2](#_Toc357758763)

[Regras de nomenclatura 2](#_Toc357758764)

[Classes 3](#_Toc357758765)

[Arquivos de Classes 4](#_Toc357758766)

[Atributos 4](#_Toc357758767)

[Constantes 4](#_Toc357758768)

[Métodos 5](#_Toc357758769)

[Objetos 5](#_Toc357758770)

# Apresentação

## Introdução

Quando programamos em PHP, devemos manter um mesmo padrão de nomenclatura durante todo nosso projeto, para que possamos identificar, a qualquer hora, a estrutura que queremos. Existe uma convenção, na sequência abordada, que é aconselhável que seja seguida.

# Nomenclatura de Estruturas

## Importante

É de extrema importância lembrar que o PHP é uma linguagem casesensitive. O que isto significa? Significa que a linguagem diferencia estruturas escritas com letras maiúsculas e minúsculas. Por exemplo:

|  |  |
| --- | --- |
| 1 | **class** minhaclasse {} |

É diferente de:

|  |  |
| --- | --- |
| 1 | **class** MinhaClasse {} |

Para o PHP, as classes minhaclasse e MinhaClasse  são completamente diferentes e serão tratadas como tal. Então tome cuidado de sempre utilizar o mesmo nome para a mesma estrutura.

## Regras de nomenclatura

#### 1 – Não utilize espaços em branco

A linguagem PHP não suporta espaços em branco nos nomes de suas estruturas.

|  |  |
| --- | --- |
| 1 2 3 4 5 | *// Isto é um nome válido* **class** MinhaClasse {}  *// Isto não é um nome válido* **class** Minha Classe {} |

#### 2 – Não utilize acentos

O PHP suporta apenas caracteres da tabela ASCII e, assim, não suporta os acentos utilizados no nosso idioma. Desta forma, vê-se programadores comumente utilizando os nomes de suas estruturas em Inglês.

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 | *// Isto é um nome válido* **class** Paginacao {}  *// Isto não é um nome válido* **class** Paginação {}  *// Isto é um nome válido* **class** Pagination {} |

#### 3 – Inicie o nome com caracteres alfabéticos ou \_

Na nomenclatura das estruturas, é possível apenas utilizar caracteres alfanuméricos e o caracter “\_”, porém, aceitasse apenas caracteres alfabéticos (de A a Z, maiúsculas ou minúsculas) e o caracter “\_” como inicial do nome, não aceitando números. Números são válidos quando utilizados no meio ou no final do nome.

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 | *// Isto é um nome válido* **class** Classe {}  *// Isto é um nome válido* **class** \_classe {}  *// Isto não é um nome válido* **class** 1classe {}  *// Isto não é um nome válido* **class** *#classe {}*  *// Isto não é um nome válido* **class** .classe {}  *// Isto não é um nome válido* **class** @classe {}  *// Isto é um nome válido* **class** doc2pdf {} |

## Classes

A nomenclatura das classes, no padrão citado, segue o conceito de UpperCamelCase. Isto é, os nomes das classes possuem todas as letras minúsculas (caixa baixa), exceto a letra inicial, que é maiúscula (caixa alta). Quando são nomes compostos, são concatenados entre si, com a inicial de cada nome maiúscula.

Exemplos:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 | **class** MinhaClasse {} **class** Mysql {} **class** Logger {} **class** Clientes {} **class** Matematica {} **class** ContaCorrente {} **class** ClientesDoBanco {} |

### Arquivos de Classes

Para salvarmos as nossas classes, comumente utilizamos a extensão .class.php e utilizamos o próprio nome da classe como nome do arquivo.

Para as classes supracitadas, teríamos os arquivos:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 | MinhaClasse.class.php Mysql.class.php Logger.class.php Clientes.class.php Matematica.class.php ContaCorrente.class.php ClientesDoBanco.class.php |

## Atributos

A nomeação de atributos, tanto de instância como de classe, é dada seguindo o conceito lowerCamelCase, iniciando com uma letra minúscula e nomes compostos iniciados com letras maiúsculas, concatenados entre si.

Exemplos:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 | **class** MinhaClasse {    **public** $nome;    **public** $nomeDoCliente;    **public** $saldoDoCaixa;    **public** $populacaoDaCidade;    **public** $valorDoProduto;    **public** $produto; } |

## Constantes

Os nomes das constantes de classes seguem o conceito UpperCase, tendo seu nome inteiro em caixa alta (letras maiúsculas).

Exemplos:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 | **class** Matematica {    **const** PI = 3.14159;    **const** E = 2.71828;    **const** AVOG = 6.02e23; } |

## Métodos

Nomes dos métodos de uma classe também seguem o conceito lowerCamelCase, tendo seu nome iniciado por letra minúscula e as inicias dos nomes compostos maiúsculas, concatenados entre si.

Veja alguns exemplos:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 | **class** Calculadora {    **function** soma($x, $y){ return $x+$y; }    **function** subtrai($x, $y) { return $x-$y; }    **function** calculaRaiz($x) { return sqrt($x); }    **function** resolvePitagoras($a, $b, $c) { return (-$b + sqrt(exp($b, 2) - 4\*$a\*$c)) / (2\*$a) } } |

## Objetos

Por fim, os nomes dos objetos também seguem o conceito de lowerCamelCase, iniciando com letra minúscula e as iniciais de nomes compostos maiúsculas, concatenados entre si.

Exemplos:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 | $minhaClasse = **new** MinhaClasse; $mysql = **new** Mysql; $logger = **new** Logger; $clientes = **new** Clientes; $matematica = **new** Matematica; $contaCorrente = **new** ContaCorrente; $clientesDoBanco = **new** ClientesDoBanco; |

Perceba que é natural que nomearmos os objetos com o mesmo nome da classe que instanciaremos. Isto facilita a identificação de nossos objetos, conhecendo os atributos e métodos que cada um acessa.