Introdução ao PHP - Parte 1

Bacharelado em Sistemas de Informação - IFCE Campus Cedro

Professor Zé Olinda (@joseolinda)

jose.olinda@ifce.edu.br

BSI

Bacharelado em Sistemas de Informação

IFCE Campus Cedro

S5 - PWEBI

Programação para a Web I

Professor Zé Olinda

Conteúdos

- Sintaxe
- Variáveis e constantes
- Echo e Print
- Operadores
- Strings
- IF..ELSE..ELSEIF

Conteúdos

(continuação)

- Switch
- For
- While
- Function
- Array
- Super globais

O que é PHP?

PHP é um *acrônimo* para **PHP: Hypertext Preprocessor**.

O que é um arquivo .php?

Um arquivo *.php* pode conter **texto**, **HTML**, **CSS**, **JavaScript** e código **PHP**.

O que o PHP pode fazer?

- PHP pode gerar páginas e conteúdo dinâmico
- PHP pode crear, abrir, ler, escrever, apagar e fechar arquivos
- PHP pode coletar dados de um formulário
- PHP pode criar **cookies**
- PHP pode adicionar, apagar e modificar dados em seu banco de dados
- PHP pode gerenciar acesso de usuários
- PHP pode criptografar dados

Sintaxe

Sintaxe básica

Todo código entre as tags `<?php` e `?>` será enterpretado como PHP.

```php
<?php
    // Comentário de uma linha
    # Outra forma de comentar uma linha
    /*
        Bloco de várias
        linhas comentadas
    */
    echo "Olá mundo PHP!";
?>
```

> *Fora da tag do PHP, todo o código restante será entendido como HTML.*

Variáveis e constantes

Variáveis

Em PHP, uma variável começa **sempre** com o símbolo $ (cifrão), seguido pelo nome da variável.

```php
$minhaPrimeiraVariavel = 10;
```

Os nomes das variáveis podem ser curtos ou descritivos

```php
$x = 37;
$y = "Variável curta";
$listaDeUsuarioDoSistema = new Users::listarTodos();
```

Variáveis: Regras de Nomes

1. Começa **sempre** com o símbolo $ (cifrão), seguido pelo nome da variável
2. Sempre deve começar com letra ou underline
3. Nunca começa com número
4. Pode conter apenas caracteres alfanuméricos e underlines (**A-z**, **0-9** e _)
5. São *case-sensitive* (diferencia maniúsculas e minúsculas)

Constantes

Em PHP, uma constante é um identificador (um nome) para um valor simples. **Nunca** deve começarcom o símbolo $ (cifrão).

```
define(name, value, case_insensitive);
```

```php
define("SAUDACAO", "Bem-vindo ao site!");
// A variável deve ser usada tal qual foi escrita,
// pois case_insensitive é falso por padrão
echo SAUDACAO;

define("DESPEDIDA", "Até mais!");
// A variável pode ser usada em minúscula ou maúsculo,
// pois foi definaida como case insensitive
echo despedida;
```

ECHO vs PRINT

echo

É uma função usada para mostrar uma saída de dados na tela. Pode ser usada com ou sem parenteses. Não retorna **nada** ao ser executada.

```php
// Com parênteses
echo ("Bom dia alunos!");
// Sem parênteses
echo "Bom dia alunos!";
```

> *Pode ser usado para mostrar texto, variável, expressão ou código HTML.*

`print`

É uma função usada para mostrar uma saída de dados na tela. Pode ser usada com ou sem parenteses. Retorna o valor **1** ao ser executada.

```php
// Com parênteses
echo ("Bom dia alunos!");
// Sem parênteses
echo "Bom dia alunos!";
```

> *Pode ser usado para mostrar texto, variável, expressão ou código HTML.*

Operadores

Grupo de Operadores

Os operadores são divididos em:

- Aritméticos
- De atribuição
- De comparação
- Incremento/Decremento
- Lógicos
- De strings

Operadores Aritméticos

Operador	Função	Exemplo
+	Somar	`$x + $y;` `//Soma de $x mais $y`
-	Subtrair	`$x - $y;` `// Diferença de $x menos $y`
*	Multiplicar	`$x * $y;` `// Produto de $x vezes $y`
/	Dividir	`$x / $y;` `// Quociente de $x dividido por $y`
%	Modulo	`$x % $y;` `// Resto de $x dividido por $y`
* *	Potência	`$x ** $y;` `// Resultado de $x elevado a $y`

Operadores de Atribuição

Operação	Equivalência	Significado
$x = $y	$x = $y	Atribuição de valor
$x += $y	$x = $x + $y	Somar o valor atual de $x com $y
$x -= $y	$x = $x - $y	Subtrair o valor atual de $x com $y
$x *= $y	$x = $x * $y	Multiplicar o valor atual de $x com $y
$x /= $y	$x = $x / $y	Multiplicar o valor atual de $x com $y
$x %= $y	$x = $x % $y	Calcular o resto de $x dividido por $y

Operadores de Comparação

Operador	Nome	Exemplo	Retorno
==	Igual	$x == $y	True se $x é igual a $y
===	Idêntico	$x == $y	True se $x é igual e do mesmo tipo de $y
!= <>	Diferente	$x != $y $x <> $y	True se $x é diferente $y
!==	Não idêntico	$x !== $y	True se $x é diferente ou **não** possuir o mesmo tipo de $y
>	Maior que	$x > $y	True se $x é maior que $y
<	Menor que	$x < $y	True se $x é menor que $y
>=	Maior ou igual	$x >= $y	True se $x é maior ou igual a $y
<=	Menor ou igual	$x <= $y	True se $x é menor ou igual a $y

Operadores de Incremento/Decremento

Operador	Nome	Efeito
++$x	Pré-incremento	Aumenta o valor de $x em uma unidade. **Depois**, retorna o valor de $x
$x++	Pós-incremento	Retorna o valor de $x. **Depois**, aumenta o valor de $x em uma unidade
++$y	Pré-decremento	Diminui o valor de $y em uma unidade. **Depois**, retorna o valor de $y
$y++	Pós-decremento	Retorna o valor de $y. **Depois**, diminui o valor de $x em uma unidade

Operadores Lógicos

Operador	Nome	Exemplo	Retorno
and **&&**	e	$x and $y $x && $y	True se ambos $x e $y são verdadeiros
or **||**	ou	$x or $y $x || $y	True se $x ou $y são verdadeiros
xor	ou exclusivo	$x xor $y	True se apenas ou $x ou $y são verdadeiros, mas nunca os dois ao mesmo tempo. Retorna **False** se ambos forem falso ou se ambos forem verdadeiros.
!	não	!$x !$y	Muda o valor lógico da variável. Se o valor for True, muda para **False**. E vice-versa.

Operadores de strings

Operador	Nome	Exemplo	Retorno
.	Concatenação	$x . $y	Concatena (une) os valores de $x e $y em uma única string
+=	Atribuição e Concatenação	$x += $y	Atribui a $x uma concatenaão do valor atual de $x com o valor de $y

Strings

Definição de strings

Uma string é uma série de caracteres, onde um caractere é o mesmo que um byte. O PHP possui suporte a um conjunto de apenas 256 caracteres.

Sintaxe

Uma string literal pode ser especificada dusando:

- aspas simples
- aspas duplas
- sintaxe heredoc
- sintaxe nowdoc (desde o PHP 5.3.0)

aspas simples

```php
<?php
echo 'isto é uma string comum';

echo 'Você pode incluir novas linhas em strings,
dessa maneira que estará
tudo bem';

// Imprime: Arnold disse uma vez: "I'll be back"
echo 'Arnold disse uma vez: "I\'ll be back"';
// Imprime: Você tem certeza em apagar C:\*.*?
echo 'Você tem certeza em apagar C:\\*.*?';
// Imprime: Você tem certeza em apagar C:\*.*?
echo 'Você tem certeza em apagar C:\*.*?';
// Imprime: Isto não será substituido: \n uma nova linha
echo 'Isto não será substituido: \n uma nova linha';
// Imprime: Variáveis $também não $expandem
echo 'Variáveis $também não $expandem';
?>
```

aspas duplas

Se a string for delimitada entre aspas duplas ("), o PHP interpretará caracteres **escapados** e variáveis:

```php
<?php
$nome = "Maria";
// Imprime: Meu nome é Maria"
echo "Meu nome é $nome";
echo "Meu nome é {$nome}";
// Imprime: Você tem conseguir
//          quebrar a linha
echo "Você pode conseguir\nquebrar a linha?";
// Imprime: Sou um programador "experiente"
echo "Sou um programador \"experiente\"";
// Imprime: Variáveis expandem: Maria
echo "Variáveis expandem: $nome";
?>
```

heredoc

```php
<?php

$str = <<<EOD
Exemplo de uma string
distribuída em várias linhas
utilizando a sintaxe heredoc.
EOD;

echo $str;

?>
```

Mais detalhes em:
PHP.net -> Strings -> heredoc

nowdoc

```php
<?php

$str = <<<'EOD'
Example of string
spanning multiple lines
using nowdoc syntax.
EOD;

echo $str;

?>
```

Mais detalhes em:
PHP.net -> Strings -> nowdoc

Funções básica para strings

```php
<?php
// Comprimento da string
echo strlen("Hello world!"); // Imprime: 12
// Conta palavras
echo str_word_count("Hello world!"); // Imprime: 2
// Inverter string
echo strrev("Hello world!"); // Imprime: !dlrow olleH
// Procurar onde a palavra começa na string
echo strpos("Hello world!", "world"); // Imprime: 6
?>
```

Mais detalhes em:
PHP.net -> Funções para Strings

Estruturas condicionais

`if` `else` `elseif`

if

*if (**teste lógico**) {*

    // Código para o resultado verdadeiro do teste

}

```php
<?php
$dia = date("j");

if ($dia == "5") {
    echo "Pagamento em conta :D";
}

?>
```

else

*if (**teste lógico**) {*

    // Código para o resultado verdadeiro do teste

*} **else** {*

    // Código para o resultado falso do teste

}

```php
<?php
$dia = date("j");

if ($dia == "5") {
    echo "Pagamento em conta :D";
} else {
        echo "Acaba a grana, mês ainda tem. :'(";
}
?>
```

elseif

*if (**teste lógico**) {*

    // Código para o resultado verdadeiro do teste

*} **elseif (outro teste lógico)** {*

    // Código para o resultado verdadeiro do novo teste lógico

}

```php
<?php
$dia = date("j");

if ($dia == "5") {
    echo "Pagamento em conta :D";
} elseif ($dia >= "1") {
    echo "Não vejo a hora de gastar este salário!";
}
```

Estrutura de escolha

`switch`

`switch`

Use a estrutura `switch` para selecionar **um entre vários valores** possíveis para uma variável ou expressão.

```
switch (variavel ou expressão) {
    case valor1:
        Código a ser executado de variavel ou expressão = valor1;
        break;
    case valor2:
        Código a ser executado de variavel ou expressão = valor2;
        break;
    default:
        Código a ser executado de nenhum caso for executado;
}
```

switch

```php
<?php
$cor = "vermelho";

switch ($cor) {
    case "vermelho":
        echo "Sua cor favorita é vermelho";
        break;
    case "azul":
        echo "Sua cor favorita é azul";
        break;
    case "verde":
        echo "Sua cor favorita é verde";
        break;
    default:
        echo "Sua cor favorita é preto!";
}
?>
```

Estruturas de Repetiçao op

`switch`

Use a estrutura `switch` para selecionar **um entre vários valores** possíveis para uma variável ou expressão.

```
switch (variavel ou expressão) {
    case valor1:
        Código a ser executado de variavel ou expressão = valor1;
        break;
    case valor2:
        Código a ser executado de variavel ou expressão = valor2;
        break;
    default:
        Código a ser executado de nenhum caso for executado;
}
```