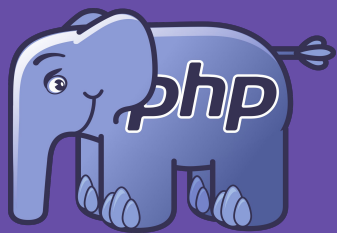


# Curso PHP e MySQL

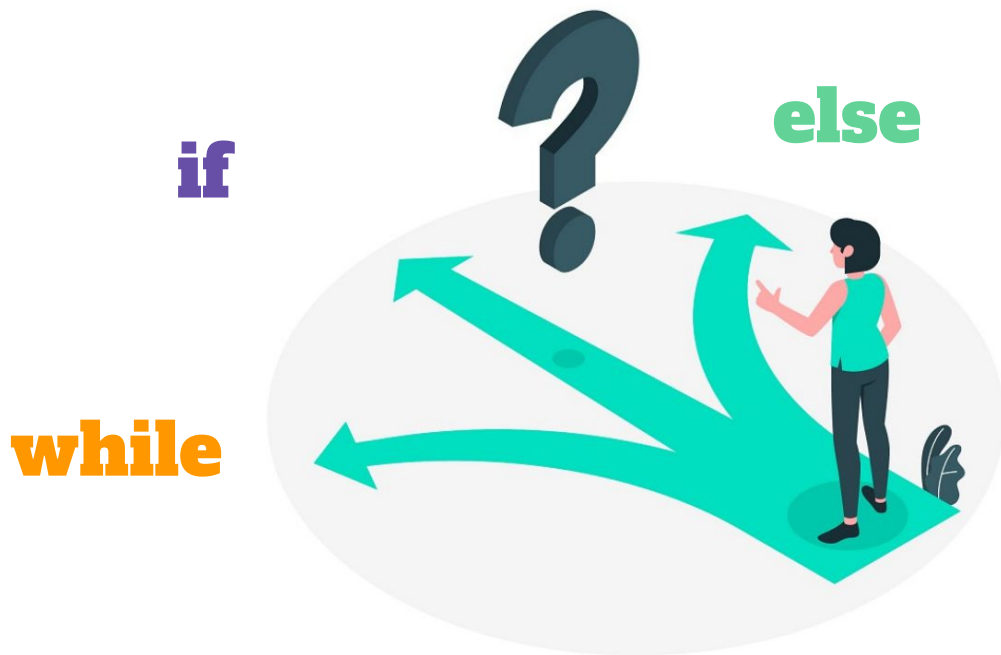
## \_Estruturas de Controle em PHP

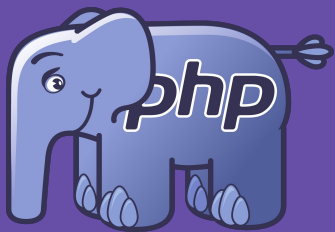
Expressões, operadores de comparação, operadores lógicos, If, else, while, do-while, switch e sintaxe alternativa para estruturas de controle.



## \_Estruturas de controle

Um script PHP é construído por **uma série de instruções**. Uma instrução pode ser uma **atribuição**, uma **chamada de função**, um **laço de repetição**, uma **instrução condicional**, ou mesmo uma instrução que não faz nada (um comando vazio). Instruções podem ser agrupadas através do encapsulamento de um grupo de comandos com chaves.





## \_expressões

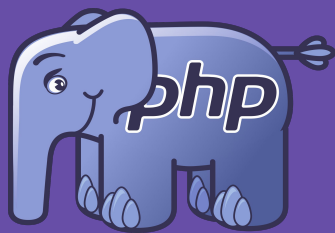
**Expressões** são os blocos de construção mais importantes do PHP. No PHP, quase tudo o que você escreve são expressões. A maneira mais simples e ainda mais precisa de definir uma expressão é "**tudo o que tem um valor**". As formas mais básicas de expressões são **constantes e variáveis**. Exemplo:

```
$a = 5;
```

Depois desta atribuição, você pode esperar que o valor de \$a seja 5 também, assim se você escrever \$b = \$a, você pode esperar que ele se comporte da mesma forma que se você escrevesse \$b = 5.

```
$b = $a = 5;
```

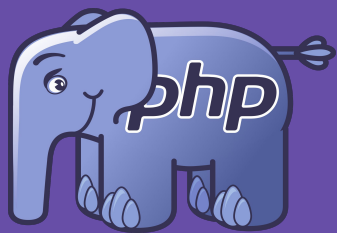
```
/* atribui o valor cinco às variáveis $a e $b */
```



## \_Operadores de Comparação

Operadores de comparação, como os seus nomes implicam, **permitem que você compare dois valores.**

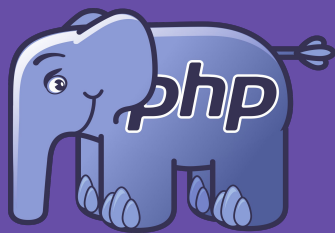
Exemplo	Nome	Resultado
<code>\$a == \$b</code>	Igual	Verdadeiro ( <b>true</b> ) se <code>\$a</code> é igual a <code>\$b</code> .
<code>\$a === \$b</code>	Idêntico	Verdadeiro ( <b>true</b> ) se <code>\$a</code> é igual a <code>\$b</code> , e eles são do mesmo tipo.
<code>\$a != \$b</code>	Diferente	Verdadeiro se <code>\$a</code> não é igual a <code>\$b</code> .
<code>\$a &lt;&gt; \$b</code>	Diferente	Verdadeiro se <code>\$a</code> não é igual a <code>\$b</code> .
<code>\$a !== \$b</code>	Não idêntico	Verdadeiro se <code>\$a</code> não é igual a <code>\$b</code> , ou eles não são do mesmo tipo (introduzido no PHP4).
<code>\$a &lt; \$b</code>	Menor que	Verdadeiro se <code>\$a</code> é estritamente menor que <code>\$b</code> .
<code>\$a &gt; \$b</code>	Maior que	Verdadeiro se <code>\$a</code> é estritamente maior que <code>\$b</code> .
<code>\$a &lt;= \$b</code>	Menor ou igual	Verdadeiro se <code>\$a</code> é menor ou igual a <code>\$b</code> .
<code>\$a &gt;= \$b</code>	Maior ou igual	Verdadeiro se <code>\$a</code> é maior ou igual a <code>\$b</code> .
<code>\$a &lt;=&gt; \$b</code>	Spaceship (nave espacial)	Um integer menor que, igual a ou maior que zero quando <code>\$a</code> é, respectivamente, menor que, igual a ou maior que <code>\$b</code> . Disponível a partir do PHP 7.



## \_Operadores lógicos

Os operadores lógicos são utilizados quando precisamos realizar operações sobre uma expressão e retornar um valor booleano como verdadeiro ou falso.

Exemplo	Nome	Resultado
<code>\$a and \$b</code>	E	Verdadeiro ( <b>true</b> ) se tanto <code>\$a</code> quanto <code>\$b</code> são verdadeiros.
<code>\$a or \$b</code>	OU	Verdadeiro se <code>\$a</code> ou <code>\$b</code> são verdadeiros.
<code>\$a xor \$b</code>	XOR	Verdadeiro se <code>\$a</code> ou <code>\$b</code> são verdadeiros, mas não ambos.
<code>! \$a</code>	NÃO	Verdadeiro se <code>\$a</code> não é verdadeiro.
<code>\$a &amp;&amp; \$b</code>	E	Verdadeiro se tanto <code>\$a</code> quanto <code>\$b</code> são verdadeiros.
<code>\$a    \$b</code>	OU	Verdadeiro se <code>\$a</code> ou <code>\$b</code> são verdadeiros.



\_if

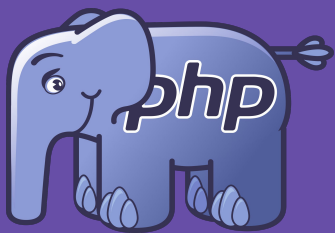
A estrutura condicional **if** é um dos recursos mais importantes nas linguagens de programação, pois **permite a execução condicional de fragmentos de código**.

```
if(expressao)  
    fragmento_codigo
```

As expressões são avaliadas por seus valores Booleanos. Se uma expressão for avaliada como **true** (verdadeiro), o PHP vai executar a declaração, e se avaliá-la como **false** (falso) - irá ignorá-la.

Neste exemplo a seguir, será exibido 'a is bigger than b' se \$a for maior que \$b

```
<?php  
if ($a > $b)  
    echo "a is bigger than b";  
?>
```



\_if

Para utilizar mais de uma declaração após a condicional ser executada, pode-se agrupar várias declarações utilizando parênteses “{ }” sinalizando a abertura e o fechamento do bloco condicional.

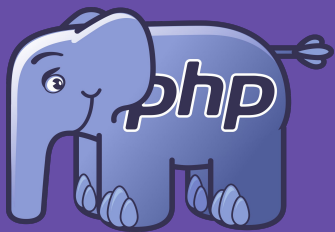
Por exemplo, o código abaixo exibirá ‘a is bigger than b’ se \$a for maior que \$b, e na linha seguinte **atribuirá o valor de \$a em \$b**:

```
<?php
if ( $a > $b ) {
    echo "a is bigger than b";
    $b = $a;
}
?>
```

```
<?php
if( $a == 1 || $a == 2 ) {
    if( $b == 3 || $b == 4 ) {
        if( $c == 5 || $d == 6 ) {
            //Do something here.
        }
    }
}
?>
```

#### Observação:

A declaração If pode ser aninhada indefinidamente dentro de outras declarações if, fornecendo completa flexibilidade para execução condicional de várias partes do programa.



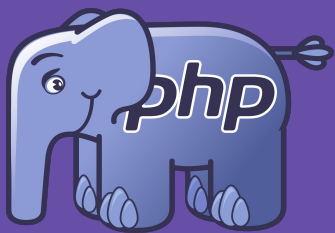
\_else

O else estende a instrução if para executar outra caso a expressão no if retornar false. Por exemplo, o código a seguir exibirá a is greater than b se \$a for maior que \$b, e a is NOT greater than b caso contrário:

```
<?php
if ($a > $b) {
    echo "a is greater than b";
} else {
    echo "a is NOT greater than b";
}
?>
```

A instrução **else** só é executada **se a expressão de avaliação do if for avaliada como false**.





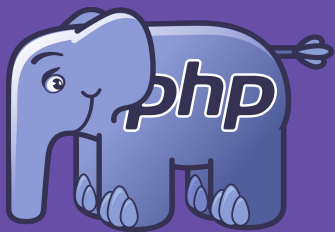
## \_while

Laços while são os mais simples tipos de laços do PHP. O formato básico de uma declaração while é:

```
while (expressao)  
    fragmento_codigo
```

O propósito da declaração while é simples. Ele dirá ao PHP para executar as declarações aninhadas repetidamente, enquanto a expressão do while for avaliada como true.

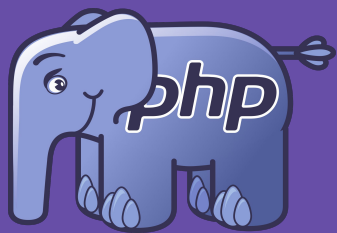
```
$i = 1;  
while ($i <= 10) {  
    echo $i++; /* the printed value would be  
               $i before the increment  
               (post-increment) */  
}
```



\_do while

O laço do-while é similar ao laço while, com exceção que a expressão de avaliação é verificada ao final de cada iteração em vez de no começo. **A maior diferença para o laço while é que a primeira iteração do laço do-while sempre é executada.**

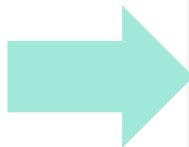
```
<?php
$i = 0;
do {
    echo $i;
} while ($i > 0);
?>
```



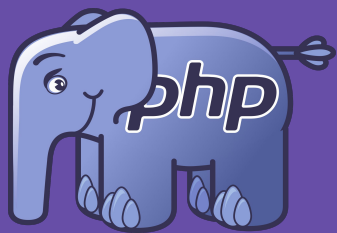
## \_switch

A declaração switch **é similar a uma série de declarações if** na mesma expressão. Em muitos casos, se deseja comparar as mesmas variáveis (ou expressões), com diferentes valores, e executar pedaços diferentes de código dependendo de qual valor ela é igual. Esta é exatamente a serventia da declaração switch.

```
<?php
if ($i == 0) {
    echo "i é igual a 0";
} elseif ($i == 1) {
    echo "i é igual a 1";
} elseif ($i == 2) {
    echo "i é igual a 2";
}
```



```
switch ($i) {
    case 0:
        echo "i é igual a 0";
        break;
    case 1:
        echo "i é igual a 1";
        break;
    case 2:
        echo "i é igual a 2";
        break;
}
?>
```



\_sintaxe  
alternativa

O PHP oferece uma sintaxe alternativa para algumas estruturas de controle; a saber, if, while, for, foreach, e switch. Em cada caso, basicamente a sintaxe alternativa é trocar a chave de abertura por dois pontos (:) e a chave de fechamento por endif;, endwhile;, endfor;, endforeach;, ou endswitch;, respectivamente.

```
<?php if ($a == 5): ?>
```

A é igual a 5

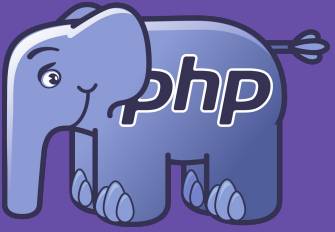
```
<?php endif; ?>
```

```
<?php switch ($foo): ?>
```

```
    <?php case 1: ?>
```

...

```
<?php endswitch ?>
```



\_atividades de  
fixação

1. E



Professor:  
Adriel Sales

# Fonte? Vá na fonte!

A screenshot of the official PHP website (php.net). The page has a dark blue header with navigation links: "php", "Downloads", "Documentation", "Get Involved", "Help", and "php2.1". A search bar is on the right. The main content area is dark blue with the "php" logo in white. Below the logo, it says "A popular general-purpose scripting language that is especially suited to web development. Fast, flexible and pragmatic, PHP powers everything from your blog to the most popular websites in the world." There are two buttons: "What's new in 8.1" and "Download". Below these are links for "8.1.9 - Changelog - Upgrading", "8.0.22 - Changelog - Upgrading", and "7.4.30 - Changelog - Upgrading". A white box highlights a news item: "PHP 8.2.0 Beta 3 available for testing »" dated "18 Aug 2022". The text in the box says: "The PHP team is pleased to announce the third beta release of PHP 8.2.0, Beta 3. This continues the PHP 8.2 release cycle, the rough outline of which is specified in the [PHP Wiki](#). For source downloads of PHP 8.2.0 Beta 3 please visit the [download page](#). Please carefully test this version and report any issues found in the [bug reporting system](#). Please DO NOT use this version in production, it is an early test version. For more information on the new features and other changes, you can read the [NEWS](#) file, or the [UPGRADING](#) file for a complete list of upgrading notes. These files can also be found in the release archive." On the right side, there are sections for "Upcoming conferences" (listing International PHP Conference Munich 2022, SymfonyWorld Online 2022 Winter Edition, SymfonyCon Disneyland Paris 2022, and CakeFest 2022: The Official CakePHP Conference), "User Group Events", "Special Thanks", and "Social media" (with a link to @official\_php).

<https://www.php.net>