

PÓS-GRADUAÇÃO

Programação PHP



PÓS-GRADUAÇÃO

Estruturas de controle

Bloco 1


Thiago Salhab Alves





► Estruturas de controle

Objetivos:

- Aprender sobre estruturas de controle no PHP.
 - Aprender sobre estruturas de repetição no PHP.
 - Aprender sobre estruturas *break*, *continue*, *return* e *goto* no PHP.
- 



► Estruturas de controle

- Segundo Soares (2013), as linguagens de programação necessitam de mecanismos para realizar o controle do fluxo do programa.
- As estruturas de controle permitem a realização, dentro dos *scripts*, de determinadas tarefas, como executar um grupo de instruções no caso de uma expressão ser verdadeira ou repetir um grupo de instruções uma quantidade finita de vezes.

► Estruturas de controle

If.

- O comando *if* permite que um grupo de instruções (delimitadas por chaves {}) possam ser executadas conforme o resultado de uma expressão ou de múltiplas expressões, ou seja, se o resultado da avaliação da expressão for verdadeiro, o bloco será executado e, caso contrário, o fluxo seguirá em frente.

► Estruturas de controle

Figura 1 – Estrutura de controle *if*

```
<?php
$valor = 100;
if($valor<=10) {
    echo "Valor menor que 10";
}
else if($valor >10 && $valor<=150) {
    echo "Valor entre 10 e 150";
}
else if($valor>150 && $valor<=1500) {
    echo "Valor maior que 150 e menor ou igual
a 1500";
} else
    echo "Valor acima de 1000";
?>
```

Fonte: elaborada pelo autor.

► Estruturas de repetição

while e *do.. while*

- As estruturas ***while*** e ***do.. while*** executam um grupo de comandos, repetidas vezes, enquanto uma determinada condição for verdadeira, ou seja, a cada interação do *loop* a expressão fornecida é avaliada e caso, seja verdadeira, um novo *loop* é executado, caso contrário (avaliado como falso), a repetição é interrompida.

► Estruturas de repetição

**Figura 2 – Estrutura de repetição
while e *do.. while***

```
<?php
$i=10;
echo "Primeiro while: ";
while($i>0) {
    echo "$i ... ";
    $i--;
}
$k=10;
echo "<br>Primeiro do..while: ";
do {
    echo "$k ... ";
    $k--;
} while($k>0);
echo "<br>Segundo while (não teremos nenhuma iteração): ";
while($i>0) {
    echo "$i / ";
    $i--;
}
echo "<br>Segundo do..while (1 iteração): ";
do {
    echo "$k ... ";
    $k--;
} while($k>0);
?>
```

Fonte: Soares (2013, p. 87).

► Estruturas de repetição

For.

- A estrutura de repetição ***for*** realiza repetições de estruturas complexas, facilitando o trabalho do desenvolvedor PHP para criar lógicas mais sofisticadas e complexas. Sua sintaxe é apresentada a seguir:

```
for (expressão_1; expressão_2; expressão_3) {  
    grupo de comandos  
}
```

► Estruturas de repetição

Figura 3 – Escopo de repetição *for*

```
<?php
for($i=0;$i<=15;$i++) {
    echo "$i ... ";
}
for(print "<br>";$i>=0;$i--) {
    echo "$i ... ";
}
?>
```

Fonte: elaborada pelo autor.

PÓS-GRADUAÇÃO

Comandos no PHP

Bloco 2

Thiago Salhab Alves



► Comandos no PHP

Break

- O comando ***break*** afeta a execução dos comandos *for*, *while*, *do.. while*, permitindo que possam ser encerrados em qualquer lugar do grupo de instruções, podendo avaliar uma expressão e conforme o resultado, encerrar o *loop*.

► Comandos no PHP

Figura 4 – Uso do comando *break*

```
for(;;) {  
    if($i>50){  
        break;  
    }  
    echo "$i ... ";  
    $i++;  
}
```

Fonte: elaborada pelo autor.



► Comandos no PHP

Continue.

- A instrução ***continue*** permite que a execução do *loop* seja alterada, mas diferentemente de *break*, não encerramos o *loop*, apenas informamos ao PHP para encerrar a iteração atual e iniciar a próxima.

► Comandos no PHP

Figura 5 – Uso do comando *continue*

```
<?php
for ($i = 0; $i < 5; ++$i) {
    if ($i == 2)
        continue
    print "$i\n";
}
?>
```

Fonte: elaborada pelo autor.



► Comandos no PHP

Return.

- O ***return*** pode ser usado dentro de funções do PHP e seu efeito é encerrar a função atual ou o programa. O valor informado (ou expressão) na instrução ***return*** é retornado para o chamador.

► Operadores no PHP

Figura 6 – Uso do comando *return*

```
<?php
function soma ($valor) {
return $valor + 5;
}
if ($valor==0) {
return -1;
}
?>
```

Fonte: elaborada pelo autor.

PÓS-GRADUAÇÃO

Teoria em Prática

Bloco 3

Thiago Salhab Alves





► Teoria em prática

Um programador está desenvolvendo uma interface para cadastro de clientes em que se deve preencher os dados, e se deparou com a necessidade de utilizar laços de repetição. Ao procurar na literatura sobre o uso de laços de repetição no PHP, observou que os laços *while*, *do.. while* e *for* são os utilizados pelos programadores. Porém o programador ficou em dúvida de quando utilizar cada tipo de laço de programação e se o resultado seria o mesmo.

Como podemos auxiliar o programador a determinar quando utilizar cada tipo de laço de repetição para melhor aproveitamento do recurso?



► Teoria em prática

- Os laços *while* e *do.. while* devem ser usados quando não se sabe a quantidade de vezes que um determinado trecho de código será repetido. O *while* faz o teste no início e o *do.. while* faz o teste no final.
- O laço *for* deve ser usado quando se sabe a quantidade de vezes que um determinado trecho de código será repetido e também permite criar lógicas mais sofisticadas e complexas.

Dica do Professor

Bloco 4

Thiago Salhab Alves





► Dica do professor

Leitura de capítulo de livro:

- Leitura do capítulo 6 (*Estruturas de Controle*). SOARES, Walace. **PHP 5: Conceitos, Programação e Integração com Banco de Dados**. 7. ed. São Paulo: Érica, 2013.
- Leitura do capítulo (*Linguagem PHP – Estruturas de Controle*). SARAIVA, Maurício de O.; BARRETO, Jeanine dos S. **Desenvolvimento de Sistemas como PHP**. Porto Alegre: SAGAH, 2018.



► Referências

SOARES, Walace. **PHP 5: Conceitos, Programação e Integração com Banco de Dados**. 7. ed. São Paulo: Érica, 2013.

SARAIVA, Maurício de O.; BARRETO, Jeanine dos S. **Desenvolvimento de Sistemas como PHP**. Porto Alegre: SAGAH, 2018.

