

PHP

Aula 6



INSTITUTO
FEDERAL
São Paulo

Conteúdo da aula

Nesta aula serão abordadas as estruturas de repetição (*loops*).

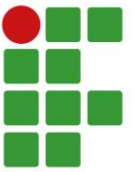
Estruturas de repetição

WHILE, FOR, FOREACH



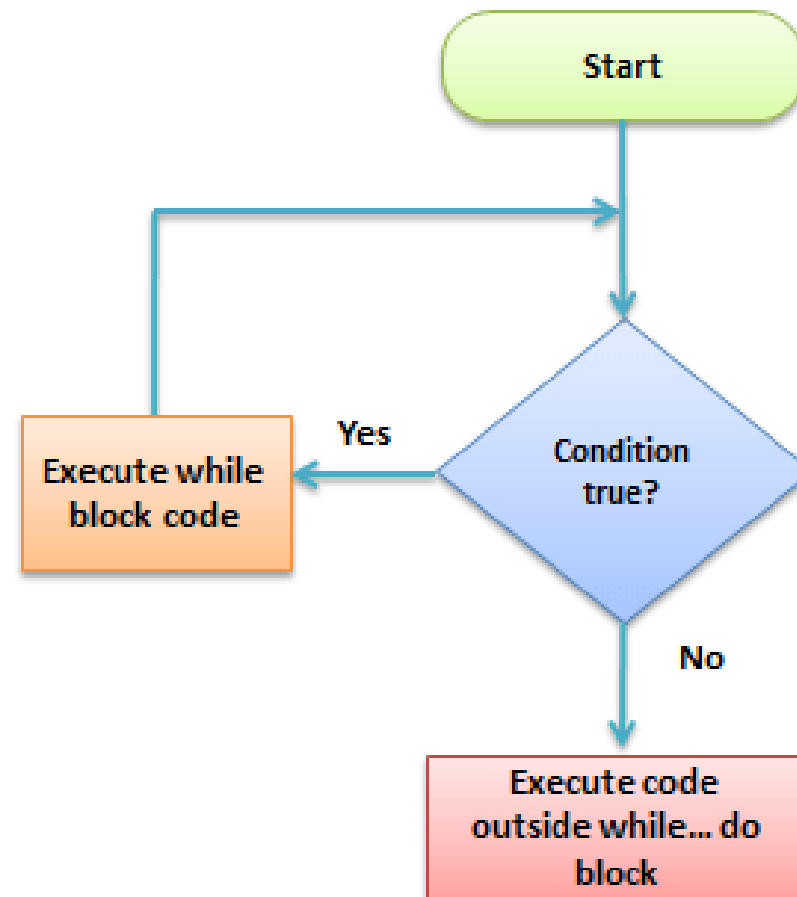
Estruturas de repetição

Os *loops* são usados para executar o mesmo bloco de código repetidamente, desde que uma determinada condição seja verdadeira.



Estruturas de repetição

Diagrama de um *loop*





Estruturas de repetição

Muitas vezes precisamos que o mesmo bloco de código seja executado **repetidamente** por um determinado número de vezes. Assim, em vez de adicionar várias linhas de código quase iguais em um script, podemos usar ***loops***.



Estruturas de repetição

PHP suporta as seguintes tipos de *loops*:

- ***while*** - percorre um bloco de código **enquanto** a condição especificada for verdadeira
- ***do...while*** - percorre um bloco de código **uma vez** e, em seguida, repete o *loop* **enquanto** a condição especificada for verdadeira



Estruturas de repetição

- ***for*** - percorre um bloco de código um número especificado de vezes
- ***foreach*** - percorre um bloco de código para cada elemento em uma **matriz**



Estruturas de repetição

O *loop* **while** executa um bloco de código enquanto a condição especificada for verdadeira.

```
while (condition is true) {  
    code to be executed;  
}
```



Estruturas de repetição

```
<?php
$x = 1;

while($x <= 5) {
    echo "The number is: $x <br>";
    $x++;
}
?>
```



Estruturas de repetição

O *loop* **do-while** sempre executará o bloco de código uma vez, verificará a condição e repetirá o loop enquanto a condição especificada for verdadeira.

```
do {  
    code to be executed;  
} while (condition is true);
```



Estruturas de repetição

```
<?php
$x = 1;

do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);
?>
```



Estruturas de repetição

Em um *loop* ***do-while***, a condição é testada **APÓS** a execução das instruções dentro do *loop*. Isso significa que o ***do-while*** executará suas instruções pelo menos uma vez, mesmo que a condição seja falsa.



Estruturas de repetição

```
<?php
$x = 6;

do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);
?>
```



**INSTITUTO
FEDERAL**
São Paulo

Estruturas de repetição

Exemplo:

https://www.w3schools.com/php/phptryit.asp?filename=tryphp_loop_do_while2



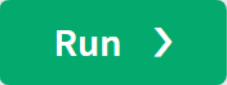




Estruturas de repetição

Este comportamento é diferente do *loop* **while**, que testa a condição **ANTES** da execução das instruções dentro do *loop*, sendo assim ele não executará nenhuma vez caso a condição seja falsa.



Estruturas de repetição

Observe, [link](#)



Result Size: 544 x 399

Get your own website

```
<!DOCTYPE html>
<html>
<body>

<?php
$x = 6;

do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);
?>






</body>
</html>
```

The number is: 6



Estruturas de repetição

Observe, [link](#)



Result Size: 544 x 399

Get your own website

```
<!DOCTYPE html>
<html>
<body>

<?php
$x = 6;

while ($x <= 5) {
    echo "The number is: $x <br>";
    $x++;
}
?>

</body>
</html>
```



Estruturas de repetição

O *loop* **for** percorre um bloco de código um número especificado de vezes.

```
for (init counter; test counter; increment counter) {  
    code to be executed for each iteration;  
}
```



Estruturas de repetição

O *loop* **for** é mais usado/indicado quando você sabe com antecedência quantas vezes o script deve ser executado.



Estruturas de repetição

Exemplo,

```
<?php
for ($x = 0; $x <= 10; $x++) {
    echo "The number is: $x <br>";
}
?>
```



Estruturas de repetição

É possível criar um *loop* **for** com incremento (ou decremento) diferente de 1.

```
<?php
for ($x = 0; $x <= 100; $x+=10) {
    echo "The number is: $x <br>";
}
?>
```



Estruturas de repetição

O *loop* ***foreach*** percorre um bloco de código para cada elemento em uma **matriz**.

```
foreach ($array as $value) {  
    code to be executed;  
}
```



Estruturas de repetição

O *loop* **foreach** funciona apenas em *arrays* e é usado para percorrer cada par chave/valor em um *array*.

Para cada iteração do *loop*, o valor do elemento atual do *array* é atribuído a **\$value** e o ponteiro do **array** é movido por um, até atingir o último elemento do **array**.



Estruturas de repetição

Exemplo (apresentando somente valores)

```
<?php
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $value) {
    echo "$value <br>";
}
?>
```



Estruturas de repetição

Exemplo (apresentando somente valores)

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");

foreach($age as $x => $val) {
    echo "$x = $val<br>";
}
?>
```



Estruturas de repetição

Referências e leitura complementar:

https://www.w3schools.com/php/php_looping.asp

https://www.w3schools.com/php/php_looping_while.asp

https://www.w3schools.com/php/php_looping_do_while.asp

https://www.w3schools.com/php/php_looping_for.asp

https://www.w3schools.com/php/php_looping_foreach.asp

Praticando



**INSTITUTO
FEDERAL**
São Paulo

Praticando

Vamos agora praticar um pouco com um exercício.

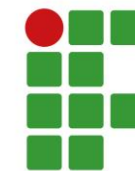




Praticando

Praticando 1 – Tabuada form GET

Crie uma página contendo um formulário com apenas um campo, um valor. Esse formulário deverá enviar as informações para a mesma página e utilizando o método GET.

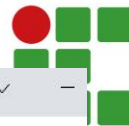


**INSTITUTO
FEDERAL**
São Paulo

Praticando

Veja:

The screenshot shows a web browser window with a single tab titled 'Tabuada'. The address bar displays 'localhost/Aula5/tabuada.php'. The page content features the title 'Tabuada' in a large, bold font. Below the title is a horizontal line. Underneath the line, the label 'Número:' is positioned to the left of a text input field. Below the input field are two buttons: a green button labeled 'Enviar' and a yellow button labeled 'Limpar'.



Tabuada

Número:

Enviar

Limpar

Tabuada do 2

$$2 \times 1 = 2$$

$$2 \times 2 = 4$$

$$2 \times 3 = 6$$

$$2 \times 4 = 8$$

$$2 \times 5 = 10$$

$$2 \times 6 = 12$$

$$2 \times 7 = 14$$

$$2 \times 8 = 16$$

$$2 \times 9 = 18$$

$$2 \times 10 = 20$$

Tabuada

Número:

Enviar

Limpar

Tabuada do 9

$$9 \times 1 = 9$$

$$9 \times 2 = 18$$

$$9 \times 3 = 27$$

$$9 \times 4 = 36$$

$$9 \times 5 = 45$$

$$9 \times 6 = 54$$

$$9 \times 7 = 63$$

$$9 \times 8 = 72$$

$$9 \times 9 = 81$$

$$9 \times 10 = 90$$

Tabuada

Número:

Enviar

Limpar

Tabuada do 34

$$34 \times 1 = 34$$

$$34 \times 2 = 68$$

$$34 \times 3 = 102$$

$$34 \times 4 = 136$$

$$34 \times 5 = 170$$

$$34 \times 6 = 204$$

$$34 \times 7 = 238$$

$$34 \times 8 = 272$$

$$34 \times 9 = 306$$

$$34 \times 10 = 340$$

TO
AL
O



Praticando

Praticando 2 – Contador

Crie uma página contendo um formulário com três campos:

- Início
- Final
- Incremento



Praticando

Praticando 2 – Contador

As informações deverão ser enviadas por POST.

Na página destino deverá ser apresentada uma contagem de início até o valor de final incrementando pelo valor definido em incremento.



**INSTITUTO
FEDERAL**
São Paulo

Praticando

Praticando 2 – Contador

Veja alguns exemplos:



Contador

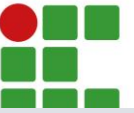
Início:

Final:

Incremento:

Calcular média

Limpar



Contador



localhost/Aula5/destino-contador.php

Contador

Parâmetros informados:

Início: 1

Final: 10

Incremento: 1

1 2 3 4 5 6 7 8 9 10



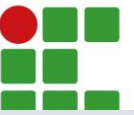
Contador

Início:

Final:

Incremento:

Calcular médiaLimpar



Contador



localhost/Aula5/destino-contador.php

Contador

Parâmetros informados:

Início: 25

Final: 1

Incremento: 2

25 23 21 19 17 15 13 11 9 7 5 3 1



Contador

Início: 100

Final: 1

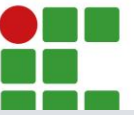
Incremento:

5



Calcular média

Limpar



Contador



localhost/Aula5/destino-contador.php

Contador

Parâmetros informados:

Início: 100

Final: 1

Incremento: 5

100 95 90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10 5



**INSTITUTO
FEDERAL**
São Paulo

Fim!



**INSTITUTO
FEDERAL**
São Paulo

Fim!

Dúvidas?

Perguntas?

Sugestões?