
AGENDA 4

PHP: ESTUTURAS DE REPETIÇÃO E FUNÇÕES



GEEaD - Grupo de Estudos de Educação a Distância
Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO MOBILE I

Expediente

Autor:

Paulo Eduardo Cardoso Andrade

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:

Flávio Biazim

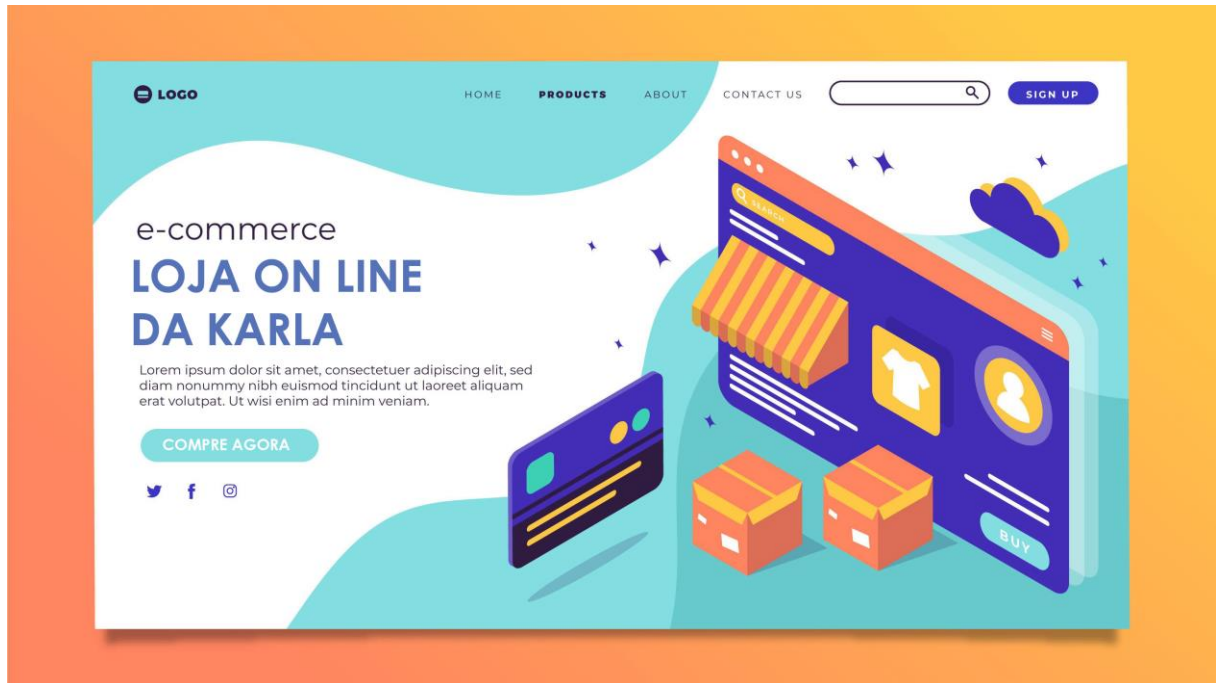


Imagem 3. Web site (FREEPIK)

Uma grande parte de páginas web necessitam de estruturas de repetição e funções no seu desenvolvimento. Para exemplificar, imagine um professor acessando o portal acadêmico de sua escola, para ter acesso às diversas turmas para as quais ministra aula. Ao clicar em um botão dentro dos vários disponíveis, é gerada uma página contendo uma tabela com as respectivas colunas, com o RM (registro de matrícula), Nome do Aluno, Idade e e-mail para contato.

Na sala que ele escolheu existem 30 alunos matriculados, então, a tabela deve possuir 31 tags `<tr>`, sendo a primeira para os títulos das colunas e as demais uma para cada aluno. Até aqui nada de mais; porém, ele clicou para gerar a mesma tabela em uma outra turma que possui 40 alunos matriculados, então, a tabela agora passa a ter 41 tags `<tr>`. Neste exemplo, é possível entender que será executada a mesma tabela, mas ela terá o tamanho diferente. Então, para automatizar e oferecer suporte a todas as composições de turmas, podemos criar a tabela a partir da quantidade de alunos na turma utilizando estruturas de repetição.

Agora, as funções. Considere que você está desenvolvendo este mesmo portal e que, em uma página, você tenha que repetir uma determinada operação de matemática complexa diversas vezes no decorrer do código, esse conceito traz a possibilidade desta operação ser

implementada dentro de uma função que será chamada para realizar essa operação, quantas vezes for necessário.

Funções

Funções são blocos de código com um nome, que têm como objetivo executar uma tarefa específica e, por meio de seu nome ser invocada e, posteriormente executada, em diversas partes do código. Essa é uma das primeiras e mais utilizadas técnicas para reutilização de código.

A declaração de funções no PHP, como em diversas outras linguagens, é iniciada com a palavra reservada “function”. Logo após, é determinado seu nome e, por fim, se houver, um ou mais parâmetros. Para delimitar o que faz parte ou não da função, utilizam-se chaves. Todo o código que estiver alocado entre essas chaves será executado quando a função for chamada. Observe a sintaxe:

```
<?php
function foo ($arg_1, $arg_2, /* ..., */ $arg_n)
{
    Instruções
    return $valor_retornado;
}
?>
```

Obs.: O PHP não diferencia letras maiúsculas de minúsculas para o nome de funções, ou seja, você pode chamar a função teste() como: Teste(), TESTE() e todos serão reconhecidos como a mesma função.

As funções em php podem possuir retorno ou não, ou seja, o php possui funções void que apenas executam as instruções determinadas e não devolve nenhum valor como resultado e, também, possui funções com retorno, que ao serem executadas, devolvem um valor ao fim de sua execução.

Para melhor entendimento, no visual studio Code, crie dois novos arquivos e os salve dentro da pasta root ou Agenda4, o primeiro com o nome de “exemploFuncao” - Neste arquivo, vamos criar um input para o usuário digitar um valor e, ao clicar no botão, ser informado se esse número é par ou ímpar.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <title>Função com e sem retorno.</title>
</head>

<body>
  <div class=" w3-third w3-center w3-animate-top w3-padding">
    <h2 class="w3-container w3-teal ">PAR ou ÍMPAR</h2>
    <form class="w3-
container" method="post" action="exemploFuncaoAction.php">
      <label class="w3-text-teal">Digite Valor</label>
      <input class="w3-input w3-border w3-light-
grey" name="txtValor" type="number" placeholder="Digite ex: 6">
      <br>
      <button class="w3-btn w3-blue-
grey" name="btnCalcular">Verificar</button>
    </form>
  </div>
</body>
</html>
```

O que deve resultar em uma página semelhante à imagem a seguir.

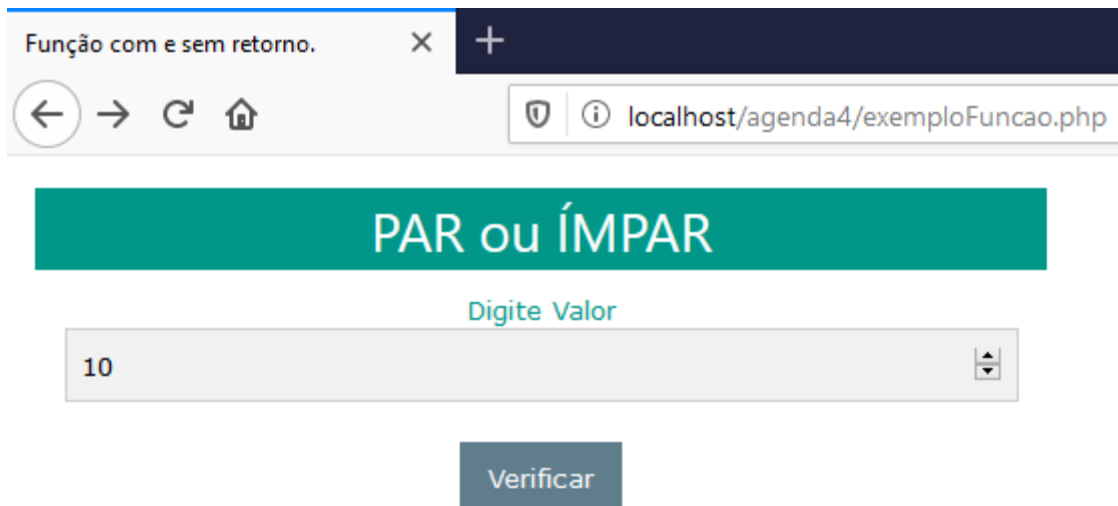


Imagem 4. Imagem do navegador executando exemplo de Função.

Para o segundo arquivo, utilize o nome “exemploFuncaoAction.php” – nele vamos criar duas funções: uma com retorno e outra sem retorno, para observar suas diferenças e os resultados no navegador.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <title>Função com e sem retorno.</title>
</head>
<body>
  <h2 class="w3-container w3-teal ">
    <?php

        function parImparRetorno($valor)
        {
            $resto = $valor % 2;
            if($resto == 0)
                return "PAR";
            else
                return "ÍMPAR";
        }
        function parImparVoid($valor)
        {
            $resto = $valor % 2;
            if($resto == 0)
                echo "PAR";
            else
                echo "ÍMPAR";
        }
    </?php>
  </h2>

```

```

    $t = $_POST['txtValor'];

    //Chamada ou Invocação da Função.
    echo parImparRetorno($t);
    echo "<br>";
    parImparVoid($t);
?>
</h2>
</body>
</html>

```

Resultado no Navegador.

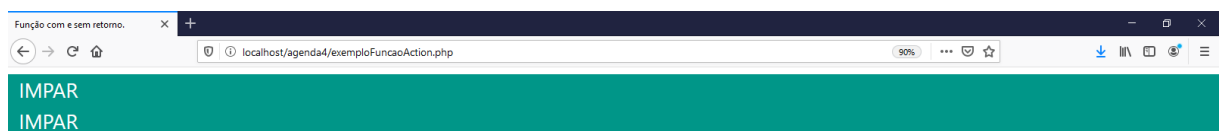


Imagem 4. Resultado no navegador.

É possível notar que o resultado no navegador foi o mesmo para as duas chamadas de função; porém, na primeira, todas as instruções resultaram em uma string, que é retornada direto para um comando echo. Já a segunda função é desenvolvida sem retorno, utilizando o comando echo de forma direta dentro da própria função.

Durante o desenvolvimento surgirão situações em que será necessário o uso de cada uma das opções, portanto, saber utilizar as duas é de grande importância.

A seguir, tire mais algumas dúvidas assistindo ao vídeo.



Fonte: Curso de PHP 7 - Aula 31 - Como criar funções (Node Studio Treinamentos(2020).

Criando funções no PHP. Disponível em: <https://www.youtube.com/watch?v=loNQIP5BUEk>. Acessado em 15/05/2020

Funções nativas

No php há uma lista enorme de funções nativas que podem ser visualizadas através do link: https://www.php.net/manual/pt_BR/indexes.functions.php.

Essas funções são recursos disponíveis que auxiliarão o desenvolvimento dos mais diversos projetos.

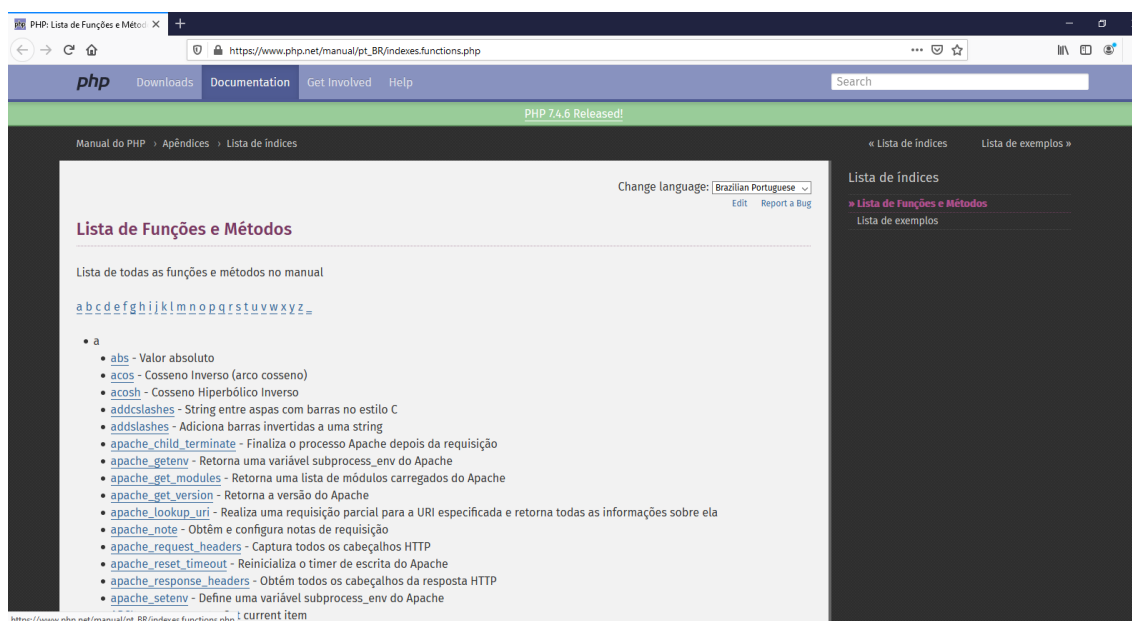


Imagem 5. Imagem da página do manual do php(PHP, 2020).

A imagem anterior mostra uma lista de funções, ao clicar em qualquer dos links, será carregada uma nova página expondo os conceitos, estruturas e uso da respectiva função. Porém, existe um detalhe que requer uma atenção especial localizado imediatamente abaixo do nome da função, essa informação é referente às versões do php que suportam essa função.

(PHP 4, PHP 5, PHP 7)

Imagem 6. Imagem de como estão expostas as versões suportadas pela versão do PHP

Vale a pena conferir os links a seguir que demonstram a utilização de algumas funções nativas.

DIEGO BROCANELLI– PHP e seu vasto arsenal de funções nativas. Acessado em 13/05/2020.

<https://www.diegebocanelli.com.br/php/php-e-seu-vasto-arsenal-de-funcoes-nativas/>

E, também, tire mais algumas dúvidas assistindo ao vídeo.



Fonte: Cuso em Video - Curso PHP Iniciante #16 - Funções String em PHP (Parte 1) (CURSO EM VIDEO, 2020).

Funções String em PHP (Parte 1) - Curso PHP Iniciante #16 - Gustavo Guanabara. Disponível em: <https://www.youtube.com/watch?v=hQLyylI2Lwl>. Acessado em 15/05/2020.

Estruturas de Repetição

No PHP, as estruturas de repetição seguem o mesmo padrão de outras linguagens, são estruturas que oferecem recursos que permitem executar trechos de código repetidas vezes baseado no resultado de uma condição predeterminada.

A primeira dessas estruturas é a “For”, provavelmente a estrutura mais utilizada entre todas, em desenvolvimento web. Sua característica é possibilitar um loop de repetição com início e fim bem definidos, formando sua estrutura básica que é dividida em três expressões, como podemos ver no código a seguir:

Sintaxe:

FOR

```
For ([inicialização];[condição];[incred. ou decem.]) {  
    [instrucoes];  
}
```

Normalmente, a primeira expressão é usada para definirmos variáveis de controle para o valor inicial e para a quantidade de repetições em um determinado bloco de código. A segunda expressão, por sua vez, é específica para definir a condição de repetição. Sendo true

(verdadeiro), o laço repetirá mais de uma vez todas as instruções, caso contrário, o laço será finalizado. Por fim, a terceira expressão define comandos que serão executados toda vez em que o bloco de comando seja executado.

Para melhor entendimento, no Visual Studio Code, crie dois novos arquivos e os salve dentro da pasta root ou Agenda4. O primeiro com o nome de “exemploFor”. Neste arquivo, vamos criar um formulário com dez botões numerados de 0 a 10, para quando o usuário clicar em um botão, apareça a tabuada do respectivo número como resultado. Então codifique:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <title>Estrutura FOR</title>
</head>

<body>
<div class=" w3-third w3-center w3-animate-top w3-padding">
  <h2 class="w3-container w3-
teal ">Escolha qual tabuada você deseja Visualizar</h2>
  <form class="w3-
container" method="post" action="exemploForAction.php">
    <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn7">7</button>
    <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn8">8</button>
    <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn9">9</button>
    <br>
    <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn4">4</button>
    <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn5">5</button>
    <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn6">6</button>
    <br>
    <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn1">1</button>
    <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn2">2</button>
    <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn3">3</button>
    <br>
    <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn0">0</button>
```

```

    </form>
  </div>
</body>
</html>

```

O que deve resultar em uma página semelhante à imagem a seguir.

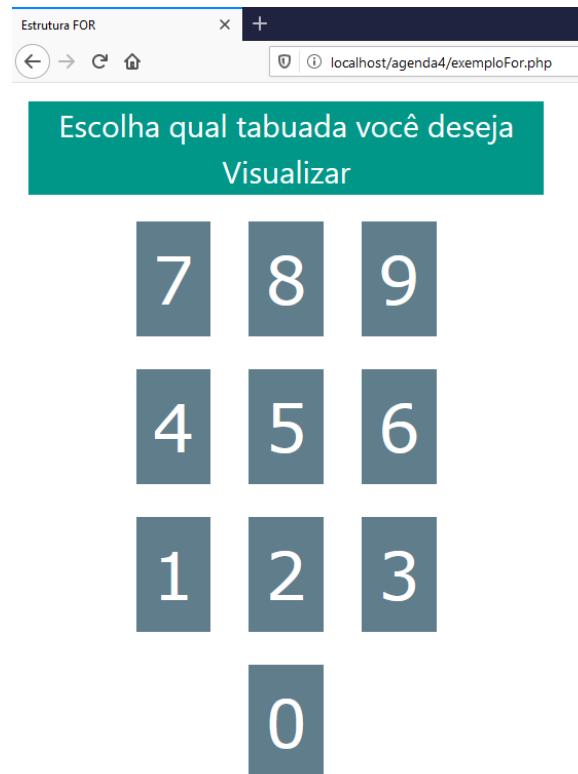


Imagem 7. Imagem do navegador executando exemplo para estrutura for.

Para o segundo arquivo, utilize o nome “exemploForAction.php” – Nele, vamos identificar qual botão foi pressionado pelo usuário, por meio da função nativa do php isset cuja utilização pode ser aplicada para verificar se determinado botão foi pressionado ou não, definindo, assim, qual tabuada deverá ser calculada. Por fim, de acordo com o código a seguir, utilizaremos a estrutura de repetição “for” para criar o resultado dentro de uma tabela.

```

<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <title>Tabuada</title>
</head>

```

```

<body>

    <?php
        echo '<br><a href="exemploFor.php" class="w3-button w3-
teal">Voltar</a><br>';
        $t = -1;
        if(isset($_POST["btn0"]))
            $t = 0;
        elseif(isset($_POST["btn1"]))
            $t = 1;
        elseif(isset($_POST["btn2"]))
            $t = 2;
        elseif(isset($_POST["btn3"]))
            $t = 3;
        elseif(isset($_POST["btn4"]))
            $t = 4;
        elseif(isset($_POST["btn5"]))
            $t = 5;
        elseif(isset($_POST["btn6"]))
            $t = 6;
        elseif(isset($_POST["btn7"]))
            $t = 7;
        elseif(isset($_POST["btn8"]))
            $t = 8;
        elseif(isset($_POST["btn9"]))
            $t = 9;
        echo '<div class="w3-quarter w3-display-middle w3-
responsive w3-teal">';
        echo '<table class="w3-table-all w3-hoverable w3-text-
black">';

        echo '<tr class="w3-teal ">';
        echo '<th class="w3-center"> Tabuada do '.$t.'</th>';
        echo '</tr>';
        for($i = 0; $i<=10; $i++)
        {
            echo '<tr>';
            echo '<td class="w3-
center">'.$t.' X '.$i.' = '.$t*$i.'</td>';
            echo '</tr>';
        }
        echo '</table>';
        echo '</div>';

    ?>
</body>
</html>

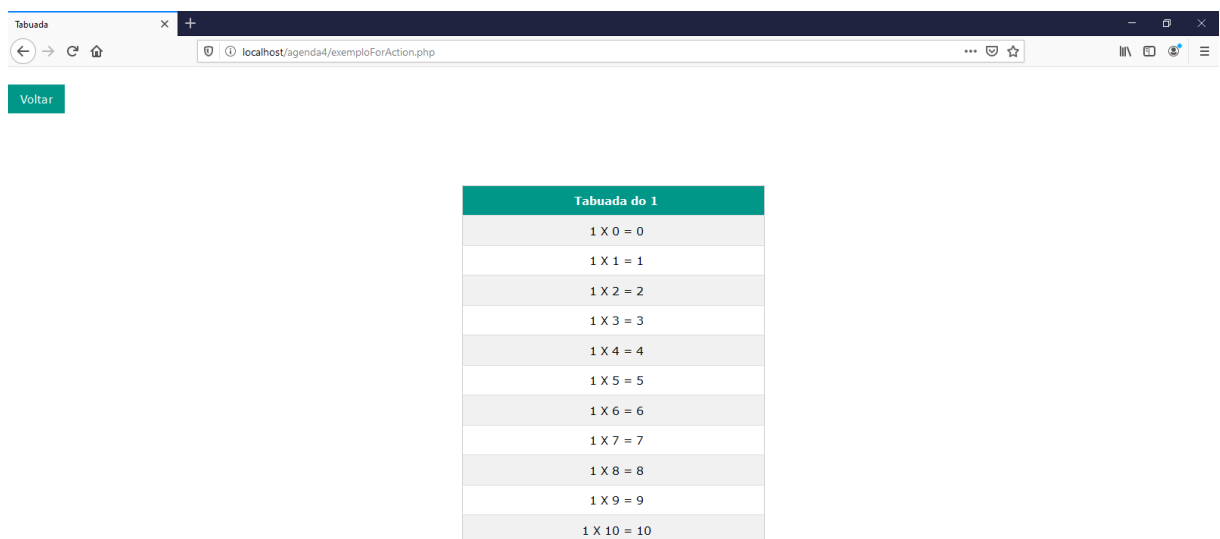
```

É possível notar que foi desenvolvido em comando echo de forma direta na tabela, bem como sua primeira linha, o que resultará em:

Tabuada do 1

Imagem 8. Primeira linha da tabela.

Neste momento, ao invés de fazer o mesmo procedimento para cada uma das respostas da tabuada, o uso da estrutura de repetição `for`, faz com que o processo do cálculo e da criação da tabela seja facilitado, pois, em seu código foi criado um laço de repetição que cria uma linha na tabela para cada cálculo da tabuada, resultando em:



The screenshot shows a web browser window with the address bar displaying `localhost/agenda4/ exemploForAction.php`. Below the address bar is a green button labeled 'Voltar'. The main content area displays a table titled 'Tabuada do 1' with 11 rows, each containing a multiplication result from `1 X 0 = 0` to `1 X 10 = 10`.

Tabuada do 1
1 X 0 = 0
1 X 1 = 1
1 X 2 = 2
1 X 3 = 3
1 X 4 = 4
1 X 5 = 5
1 X 6 = 6
1 X 7 = 7
1 X 8 = 8
1 X 9 = 9
1 X 10 = 10

Imagem 9. Resultado no navegador para quando o usuário escolher o botão com o número 1.

While

A estrutura de repetição `while`, talvez a estrutura mais simples, usa apenas expressão booleana (condição) no início de sua sintaxe. Veja:

WHILE

```
While (expressão) {  
  [instrucoes];  
}
```

Essa instrução expressa que enquanto (while) a condição for verdadeira o bloco de comandos será executado, até que em um determinado momento a condição não seja satisfeita (falsa), finalizando, assim, sua execução.

Para melhor entendimento, no Visual Studio Code, crie dois novos arquivos e os salve dentro da pasta root ou Agenda4, o primeiro com o nome de “estruturaWhile” - Neste arquivo, vamos criar um formulário com um input de texto e um botão para que o usuário digite um valor e, quando clicar no botão, obtenha como resultado a tabuada do número digitado. Então codifique:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <title>Form para tabuada</title>
</head>

<body>
  <div class=" w3-third w3-center w3-animate-top w3-padding">
    <h2 class="w3-container w3-
teal ">Digite qualquer valor para gerar sua tabuada</h2>
    <form class="w3-
container" method="post" action="exemploWhileAction.php">
      <label class="w3-text-
teal"><b>Valor para cálculo da tabuada</b></label>
      <input class="w3-input w3-border w3-light-
grey" name="txtValor" type="number" placeholder="Digite ex: 6">
      <br>
      <button class="w3-btn w3-blue-
grey" name="btnCalcular">Calcular</button>
    </form>
  </div>
</body>
</html>
```

O que deve resultar em:

Form para tabuada

localhost/agenda4/exemploWhile.php

Digite qualquer valor para gerar sua tabuada

Valor para cálculo da tabuada

Digite ex: 6

Calcular

Imagem 10. Resultado no navegador para o arquivo `exemploWhile.php`.

Para o segundo arquivo, utilize o nome “`exemploWhileAction.php`” – Neste arquivo, vamos obter o valor digitado pelo usuário no input por meio da variável global “`POST`”, este valor determinará qual tabuada será calculada e, por fim, a utilização da estrutura de repetição “`while`” para criar o resultado dentro de uma tabela de acordo com o código a seguir.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <title>Tabuada</title>
</head>
<body>

  <?php
    echo '<br><a href="exemploWhile.php" class="w3-button w3-teal">Voltar</a><br>';

    $v = $_POST["txtValor"];
    echo '<div class="w3-quarter w3-display-middle w3-responsive w3-teal">';

    echo '<table class="w3-table-all w3-hoverable w3-text-black">';
    echo '<tr class="w3-teal ">';
    echo '<th class="w3-center"> Tabuada do '.$v.'</th>';
    echo '</tr>';
    $i = 0;
```

```

        while($i<=10)
        {
            echo '<tr>';
            echo '<td class="w3-center">'.$v.' X '.$i.' = '.$v*$i.'</td>';
            echo '</tr>';
            $i++;
        }
        echo '</table>';
        echo '</div>';

    ?>
</body>
</html>

```

Perceba que também foi utilizada a estrutura de repetição para que cada uma das respostas da tabuada apareça dentro de uma linha da tabela, gerando um resultado para o usuário idêntico ao anterior.



Tabuada do 10
10 X 0 = 0
10 X 1 = 10
10 X 2 = 20
10 X 3 = 30
10 X 4 = 40
10 X 5 = 50
10 X 6 = 60
10 X 7 = 70
10 X 8 = 80
10 X 9 = 90
10 X 10 = 100

Imagem 11. Resultado no navegador para quando o usuário digitar o número 10.

DO-WHILE

A estrutura de repetição do-while é provável que seja a última que diz respeito à utilização em desenvolvimento web. Esta estrutura possui uma característica de que a primeira iteração é sempre executada, pois sua expressão booleana (condição) é realizada apenas no fim da estrutura. Após a expressão booleana ser executada, caso o resultado seja verdadeiro, será realizado novamente o bloco de instruções, caso contrário, a execução será finalizada. Segue a sintaxe:

DO..WHILE

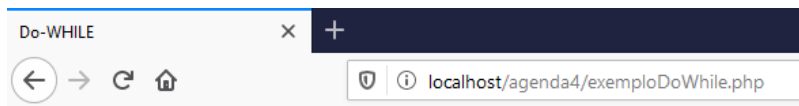
```
do {
    [instrucoes];
} while (expressão);
```

Para melhor entendimento, no visual studio Code, crie dois novos arquivos e os salve dentro da pasta root ou Agenda4, o primeiro com o nome de “estruturasDoWhile” - nele, vamos criar um formulário com apenas um botão para que o usuário, ao clicar gere na sua página todas as tabuadas do 0 ao 10. Para isso codifique:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Do-WHILE</title>
</head>

<body>
    <div class=" w3-third w3-center w3-animate-top w3-padding">
        <h2 class="w3-container w3-
teal ">Clique para Gerar todas as Tabuadas do 0 a 10</h2>
        <form class="w3-
container" method="post" action="exemploDoWhileAction.php">
            <br>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btnGerar">Tabuadas</button>
        </form>
    </div>
</body>
</html>
```

O que deve resultar em:



Tabuadas

Imagem 12. Resultado no navegador para o arquivo exemplo doWhile.php.

Para o segundo arquivo, utilize o nome “exemploDoWhileAction.php” – Neste arquivo, serão executadas duas estruturas de repetição aninhadas: a primeira será responsável por criar cada uma das 11 tabelas e, a segunda, será responsável por cada uma das 11 linhas de cada uma das tabelas. O que resultará no código a seguir:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <title>Tabuada</title>
</head>
<body>
  <?php
    echo '<br><a href="exemploDoWhile.php" class="w3-button w3-teal">Voltar</a><br>';
    $i = 0;
    $j = 0;
    do{
      echo '<div class="w3-quarter w3-responsive w3-teal">';
      echo '<table class="w3-table-all w3-hoverable w3-text-black">';
      echo '<tr class="w3-teal ">';
      echo '<th class="w3-center"> Tabuada do '.$j.'</th>';
      echo '</tr>';
      $i = 0;
      do{
        echo '<tr>';
        echo '<td class="w3-center">'.$j.' X '.$i.' = '.$j*$i.'</td>';
        echo '</tr>';
        $i++;
      }while($i<=10);
      echo '</table>';
      echo '</div>';
      $j++;
    }
```

```

        }while($j<=10);
    ?>
</body>
</html>

```

Perceba que o alinhamento das duas estruturas faz com que o bloco de instruções da segunda estrutura seja executado 10 vezes para cada execução da primeira estrutura. O resultado no navegador dever ser semelhante à imagem a seguir.

Tabuada do 0	Tabuada do 1	Tabuada do 2	Tabuada do 3	Tabuada do 4	Tabuada do 5	Tabuada do 6	Tabuada do 7
0 X 0 = 0	1 X 0 = 0	2 X 0 = 0	3 X 0 = 0	4 X 0 = 0	5 X 0 = 0	6 X 0 = 0	7 X 0 = 0
0 X 1 = 0	1 X 1 = 1	2 X 1 = 2	3 X 1 = 3	4 X 1 = 4	5 X 1 = 5	6 X 1 = 6	7 X 1 = 7
0 X 2 = 0	1 X 2 = 2	2 X 2 = 4	3 X 2 = 6	4 X 2 = 8	5 X 2 = 10	6 X 2 = 12	7 X 2 = 14
0 X 3 = 0	1 X 3 = 3	2 X 3 = 6	3 X 3 = 9	4 X 3 = 12	5 X 3 = 15	6 X 3 = 18	7 X 3 = 21
0 X 4 = 0	1 X 4 = 4	2 X 4 = 8	3 X 4 = 12	4 X 4 = 16	5 X 4 = 20	6 X 4 = 24	7 X 4 = 28
0 X 5 = 0	1 X 5 = 5	2 X 5 = 10	3 X 5 = 15	4 X 5 = 20	5 X 5 = 25	6 X 5 = 30	7 X 5 = 35
0 X 6 = 0	1 X 6 = 6	2 X 6 = 12	3 X 6 = 18	4 X 6 = 24	5 X 6 = 30	6 X 6 = 36	7 X 6 = 42
0 X 7 = 0	1 X 7 = 7	2 X 7 = 14	3 X 7 = 21				
0 X 8 = 0	1 X 8 = 8	2 X 8 = 16	3 X 8 = 24				
0 X 9 = 0	1 X 9 = 9	2 X 9 = 18	3 X 9 = 27				
0 X 10 = 0	1 X 10 = 10	2 X 10 = 20	3 X 10 = 30				

Imagem 13. Resultado no navegador para quando o usuário clicar no botão Gerar Tabuadas.

FOREACH

Como o laço de repetição foreach foi desenvolvido para gerar iterações para o uso em arrays e objetos no PHP, assunto ainda não abordado em php; seu uso ficará condicionado a esses tipos de dados. Para saber mais assista ao vídeo abaixo.



Fonte: Curso de PHP #10 - Loop FOREACH (CFBCURSOS, 2020).

Estrutura de repetição FOREACH bastante interessante, o loop foreach. Disponível em:

<https://www.youtube.com/watch?v=fD0wKp6Cqm0>. Acessado em 15/05/2020.

Obs.: Comando break em laço de repetição.

O comando break em uma estrutura de repetição tem a função de parar a execução de laços de repetição. Sua utilização é muito simples, como podemos ver no exemplo a seguir:

Exemplo:

```
<?php
$i = 0;

while (true) { //Loop infinito.

    if ($i > 10) {
        break;
    }
    echo $i."<br>";
    $i++;
}

?>
```

O resultado é a parada do loop assim que o \$i for maior do que 4.

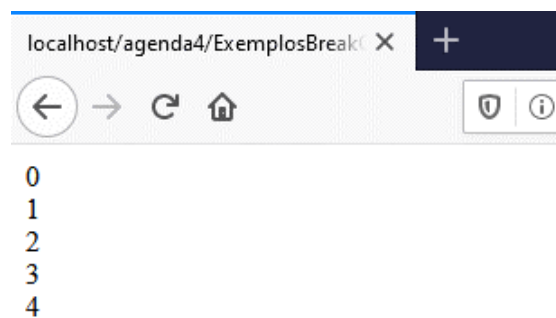


Imagem 14. Resultado no navegador para o uso do comando break.

Obs.: Comando continue em laço de repetição.

O comando continue tem uma função diferente, ele pula a iteração na qual foi executado, passando para a próxima iteração do laço, continuando assim o loop. Seu uso também é bem simples:

Exemplo:

```
<?php

$i = 0;

for($i = 0; $i <=10; $i++)
{
    if ($i == 5) {
        continue;
    }
    echo $i."<br>";
}

?>
```

Observando a imagem a seguir, percebemos que o número 5 não é exibido, então, concluímos que a iteração que mostraria o número 5, foi ignorada pela utilização do comando `continue`.

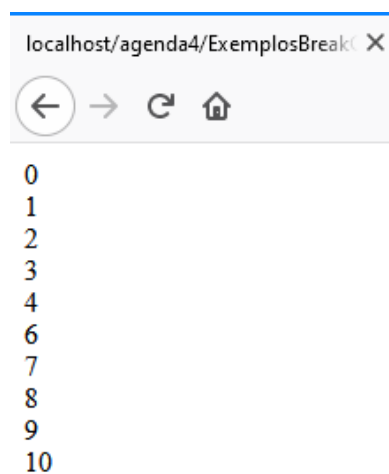


Imagem 15. Resultado no navegador para o uso do comando `continue`.



Utilizando o que foi visto até agora, vamos criar uma página completa para mãe de Karla.

1- Crie um arquivo PHP na pasta root ou Agenda4.

a) Crie uma divisão e coloque a data atual centralizada na Página.

b) Divida a página em três:

- Parte 1: Clicar em um botão para gerar a Tabuada respectiva ao número do botão;
- Parte 2: Digitar o valor e clicar para gerar a tabuada do valor digitado;
- Parte 3: Clicar em um botão para gerar todas as tabuadas;

2- Crie um arquivo PHP para receber todas as ações possíveis do arquivo anterior e exibir as tabuadas requeridas.

Dicas:

- Procure pela função nativa `date()`.
- Utilize o exemplo de cada estrutura de repetição
- Utilize o comando `isset` para verificar qual botão foi acionado.

A seguir, confira se você conseguiu resolver os desafios propostos!

Codificação ArquivoFormulário

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <title>Form para tabuada</title>
</head>
<body>
<div>
  <h2 class="w3-container w3-teal w3-center">Data de Hoje: <?php echo date("d/m/Y");?></h2>
</div>
<div>
<div class=" w3-third w3-center w3-animate-top w3-padding">
```

```

        <h2 class="w3-container w3-
teal ">Escolha qual tabuada você deseja Visualizar</h2>
        <form class="w3-
container" method="post" action="voceNoComandoAction.php">
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn7">7</button>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn8">8</button>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn9">9</button>
            <br>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn4">4</button>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn5">5</button>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn6">6</button>
            <br>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn1">1</button>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn2">2</button>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn3">3</button>
            <br>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btn0">0</button>
        </form>
    </div>
    <div class=" w3-third w3-center w3-animate-top w3-padding">
        <h2 class="w3-container w3-
teal ">Digite qualquer valor para gerar sua tabuada</h2>
        <form class="w3-
container" method="post" action="voceNoComandoAction.php">
            <label class="w3-text-
teal"><b>Valor para cálculo da tabuada</b></label>
            <input class="w3-input w3-border w3-light-
grey" name="txtValor" type="number" placeholder="Digite ex: 6">
            <br>
            <button class="w3-btn w3-blue-
grey" name="btnCalcular">Calcular</button>
        </form>
    </div>
    <div class=" w3-third w3-center w3-animate-top w3-padding">
        <h2 class="w3-container w3-
teal ">Clique para Gerar todas as Tabuadas do 0 a 10</h2>
        <form class="w3-
container" method="post" action="voceNoComandoAction.php">
            <br>
            <button class="w3-btn w3-blue-grey w3-margin w3-
jumbo" name="btnGerar">Tabuadas</button>
        </form>
    </div>

```

```
</div>
</body>
</html>
```

Resultado no Navegador.

Arquivo Formulário

Imagem 16 –Possível resultado no navegador do Exercício Você no Comando.

Codificação Arquivo de Ação

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <title>Mensagem</title>
</head>
<body>

  <?php
    echo '<br><a href="voceNoComando.php" class="w3-button w3-teal">Voltar</a><br>';

    if(isset($_POST["btnCalcular"]))
    {
      $v = $_POST["txtValor"];
      echo '<div class="w3-quarter w3-display-middle w3-responsive w3-teal">';

      echo '<table class="w3-table-all w3-hoverable w3-text-black">';
      echo '<tr class="w3-teal ">';
      echo '<th class="w3-center"> Tabuada do '.$v.'</th>';
      echo '</tr>';
```



```

        $i = 0;
        while($i<=10)
        {
            echo '<tr>';
            echo '<td class="w3-center">'.$v.' X '.$i.' = '.$v*$i.'</td>';
            echo '</tr>';
            $i++;
        }
        echo '</table>';
        echo '</div>';
    }
    else
    {
        if(isset($_POST["btnGerar"]))
        {
            $i = 0;
            $j = 0;
            do{
                echo '<div class="w3-quarter w3-responsive w3-teal">';
                echo '<table class="w3-table-all w3-hoverable w3-text-';
                echo '<tr class="w3-teal ">';
                echo '<th class="w3-center"> Tabuada do '.$j.'</th>';
                echo '</tr>';
                $i = 0;
                do{
                    echo '<tr>';
                    echo '<td class="w3-';
                    echo '<tr>';
                    echo '</tr>';
                    $i++;
                }while($i<=10);
                echo '</table>';
                echo '</div>';
                $j++;
            }while($j<=10);
        }
        else
        {
            $t = -1;
            if(isset($_POST["btn0"]))
                $t = 0;
            elseif(isset($_POST["btn1"]))
                $t = 1;
            elseif(isset($_POST["btn2"]))
                $t = 2;
            elseif(isset($_POST["btn3"]))
                $t = 3;
            elseif(isset($_POST["btn4"]))
                $t = 4;
            elseif(isset($_POST["btn5"]))
                $t = 5;
            elseif(isset($_POST["btn6"]))

```

```

        $t = 6;
    elseif(isset($_POST["btn7"]))
        $t = 7;
    elseif(isset($_POST["btn8"]))
        $t = 8;
    elseif(isset($_POST["btn9"]))
        $t = 9;
    echo '<div class="w3-quarter w3-display-middle w3-responsive w3-teal">';
    echo '<table class="w3-table-all w3-hoverable w3-text-black">';

    echo '<tr class="w3-teal ">';
    echo '<th class="w3-center"> Tabuada do '.$t.'</th>';
    echo '</tr>';
    for($i = 0; $i<=10; $i++)
    {
        echo '<tr>';
        echo '<td class="w3-center">'.$t.' X '.$i.' = '.$t*$i.'</td>';
        echo '</tr>';
    }
    echo '</table>';
    echo '</div>';
}

?>
</body>
</html>

```

Resultado no Navegador.

Parte 1 e Parte 2



Tabuada do 8
8 X 0 = 0
8 X 1 = 8
8 X 2 = 16
8 X 3 = 24
8 X 4 = 32
8 X 5 = 40
8 X 6 = 48
8 X 7 = 56
8 X 8 = 64
8 X 9 = 72
8 X 10 = 80

Imagem 17. Possível resultado no navegador do Exercício Você no Comando.

Parte 3

Mensagem x +

localhost/agenda4/voceNoComandoAction.php 90%

Voltar

Tabuada do 0	Tabuada do 1	Tabuada do 2	Tabuada do 3
0 X 0 = 0	1 X 0 = 0	2 X 0 = 0	3 X 0 = 0
0 X 1 = 0	1 X 1 = 1	2 X 1 = 2	3 X 1 = 3
0 X 2 = 0	1 X 2 = 2	2 X 2 = 4	3 X 2 = 6
0 X 3 = 0	1 X 3 = 3	2 X 3 = 6	3 X 3 = 9
0 X 4 = 0	1 X 4 = 4	2 X 4 = 8	3 X 4 = 12
0 X 5 = 0	1 X 5 = 5	2 X 5 = 10	3 X 5 = 15
0 X 6 = 0	1 X 6 = 6	2 X 6 = 12	3 X 6 = 18
0 X 7 = 0	1 X 7 = 7	2 X 7 = 14	3 X 7 = 21
0 X 8 = 0	1 X 8 = 8	2 X 8 = 16	3 X 8 = 24
0 X 9 = 0	1 X 9 = 9	2 X 9 = 18	3 X 9 = 27
0 X 10 = 0	1 X 10 = 10	2 X 10 = 20	3 X 10 = 30
Tabuada do 4	Tabuada do 5	Tabuada do 6	Tabuada do 7
4 X 0 = 0	5 X 0 = 0	6 X 0 = 0	7 X 0 = 0
4 X 1 = 4	5 X 1 = 5	6 X 1 = 6	7 X 1 = 7
4 X 2 = 8	5 X 2 = 10	6 X 2 = 12	7 X 2 = 14
4 X 3 = 12	5 X 3 = 15	6 X 3 = 18	7 X 3 = 21
4 X 4 = 16	5 X 4 = 20	6 X 4 = 24	7 X 4 = 28
4 X 5 = 20	5 X 5 = 25	6 X 5 = 30	7 X 5 = 35
4 X 6 = 24	5 X 6 = 30	6 X 6 = 36	7 X 6 = 42

Imagem 18. Possível resultado no navegador do Exercício Você no Comando.