DIO
CEZAR
GO

PHP

parte 2

COMANDO EXIT

- comando `exit()` encerra a execução do script php;
- utilizado quando não se deseja continuar exibindo a página.

EXEMPLO EXIT

```
<?php
     $permissao = false;
     if(!$permissao){
          echo “Você não pode acessar essa página!”;
          exit();
     }
?>
```

A FUNÇÃO LIST

- a função `list()` quebra um *array* em vários valores;
- captura o valor do array e os atribui para a lista de variável passada como parâmetro;

EXEMPLO LIST

```
<?php
     $info = array('Café', 'marrom', 'cafeína');
     // Listando todas as variáveis
     list($bebida, $cor, $substancia) = $info;
     echo "$bebida é $cor e $substancia o faz especial.";
?>
```

FUNÇÕES DE CLASSIFICAÇÃO - ASORT

`asort()` → classifica um *array* associativo crescentemente, ordenando pelos valores

EXEMPLO ASORT

```
<?php
     $frutas = array( "d" => "limão",  "a" => "laranja",
                      "b" => "banana", "c" => "maçã");
     asort($frutas);
     foreach($frutas as $chave => $valor)
         echo "$chave = $valor\n";
?>
```

RESULTADO

```
b = banana
a = laranja
d = limão
c = maçã
```

FUNÇÕES DE CLASSIFICAÇÃO - KSORT

`ksort()` → classifica um *array* associativo crescentemente, ordenando pelas chaves

EXEMPLO KSORT

```
<?php
     $frutas = array( "d" => "limão",  "a" => "laranja",
                      "b" => "banana", "c" => "maçã");
     ksort($frutas);
     foreach($frutas as $chave => $valor)
         echo "$chave = $valor\n";
?>
```

RESULTADO

```
a = laranja
b = banana
c = maçã
d = limão
```

FUNÇÕES DE CLASSIFICAÇÃO - RSORT

`rsort()` → classifica um array numérico decrescentemente

EXEMPLO RSORT

```
<?php
      $frutas = array ("limão", "laranja", "banana", "maçã");
      rsort($frutas);
      foreach($frutas as $chave => $valor)
          echo "$chave = $valor\n";
?>
```

RESULTADO

```
0 = maçã
1 = limão
2 = laranja
3 = banana
```

FUNÇÕES DE REORDENAÇÃO - SHUFFLE

shuffle() → ordena aleatoriamente os elementos do array

EXEMPLO SHUFFLE

```
<?php
     $frutas = array( "d" => "limão",  "a" => "laranja",
                      "b" => "banana", "c" => "maçã");
     shuffle($frutas);
     foreach ($frutas as $chave => $valor)
          echo $chave." = ".$valor."<br>";
?>
```

RESULTADO

```
0 = laranja
1 = banana
2 = limão
3 = maça
```

NOTA → esta função define novas chaves para os elementos em array. Ela irá remover qualquer chave que você tenha definido.

FUNÇÕES DE REORDENAÇÃO - ARRAY_REVERSE

`array_reverse()` → cria um novo *array* com o mesmo conteúdo do original só que na ordem inversa

EXEMPLO ARRAY_REVERSE

```
<?php
      $frutas = array( "d" => "limão",  "a" => "laranja",
                       "b" => "banana", "c" => "maçã");
      $frutas_reverse = array_reverse($frutas);
      foreach ($frutas_reverse as $chave => $valor)
            echo $chave." = ".$valor."<br>";
?>
```

RESULTADO

```
c = maça
b = banana
a = laranja
d = limão
```

FORMATANDO STRINGS - TRIM

`trim()` → elimina os espaços em branco do início e do final da string, retornando a string resultante

EXEMPLO TRIM

```
<?php
     $str = “ Testando uma string ”;
     trim($str);
     echo $str; // “Testando uma string”
?>
```

FORMATANDO STRINGS - LTRIM

`ltrim()` → elimina somente os espaços em branco do início

EXEMPLO LTRIM

```
<?php
     $str = “ Testando uma string ”;
     ltrim($str);
     echo $str; // “Testando uma string ”
?>
```

FORMATANDO STRINGS - CHOP

chop() → elimina somente os espaços em branco do final

EXEMPLO CHOP

```
<?php
     $str = “ Testando uma string ”;
     chop($str);
     echo $str; // “ Testando uma string”
?>
```

FORMATANDO STRINGS - NL2BR

`nl2br()` → substitui as novas linhas da string ('/n') pela tag
 do html;

EXEMPLO TRIM

```
<?php
     $texto = "Testando \n uma nova linha";
     echo nl2br($texto);
     // Testando <br> uma nova linha
?>
```

FORMATANDO STRINGS - CAPITALIZAÇÕES

- `strtoupper()` → coloca a string toda em letras maiúsculas;
- `strtolower()` → coloca a string toda em letras minúsculas;
- `ucfirst()` → coloca o primeiro caractere da string em letra maiúscula;
- `ucwords()` → coloca o primeiro caractere de cada palavra em letra maiúscula;

FORMATANDO STRINGS - CAPITALIZAÇÕES

EXEMPLOS CAPITALIZAÇÕES

```php
<?php
     $str = "testando uma string";
     echo strtoupper($str);
     // TESTANDO UMA STRING
     echo strtolower($str);
     // testando uma string
     echo ucfirst($str);
     // Testando uma string
     echo ucwords($str);
     // Testando Uma String
?>
```

FORMATANDO STRINGS - CARACTERES ESPECIAIS

em php quando queremos construir uma *string* com um caractere especial, devemos inserir antes do caractere uma '\' para que o interpretador entenda aquilo como uma string, por exemplo:

EXEMPLO CARACTERES ESPECIAIS

```
<?php
     $str = "testando uma \"string\"";
     echo $str;
     // estando uma "string"
?>
```

FORMATANDO STRINGS - BARRA AUTOMÁTICA

`AddSlashes()` → adiciona automaticamente uma barra invertida (\) antes de caracteres especiais;

`StripSlashes()` → remove as barras invertidas (\) localizadas antes de caracteres especiais;

EXEMPLO CARACTERES ESPECIAIS

```
<?php
     $str = "Seu nome é O'reilly?";
     echo addslashes($str);
     // Seu nome é O\'reilly?
     echo stripslashes($str);
     // Seu nome é O'reilly?
?>
```

NOTA → útil quando precisamos armazenar dados em nossos bancos (que não aceitam caracteres como ‘ ou “)

REUTILIZAÇÃO DE CÓDIGO

- usamos `require()` ou `include()` para inserir um outro arquivo no arquivo corrente;
    - se for um outro arquivo php, pode-se utilizar todos os recurSos oferecidos pelo código;
- a diferença entre include e require é a forma como um erro é tratado;
- require produz um erro `E_COMPILE_ERROR`, o que encerra a execução do script;
- o include apenas produz um *warning;*
- `include_once()` tem a garantia que o arquivo não será incluído novamente se ele já foi incluído antes;
- `require_once` é análogo ao `include_once`;

REUTILIZAÇÃO DE CÓDIGO

my_sum.php

```
<?php
     function my_sum($a, $b){
          return ($a + $b);
     }
?>
```

index.php

```
<?php
     require_once(“my_sum.php”);
     echo my_sum(10, 20);
     // 30
?>
```

CRIANDO SUAS PRÓPRIAS FUNÇÕES

- regras para nomes:
  - distinguem maiúsculas de minúsculas;
  - não pode ter o mesmo nome que uma função pré-existente;
  - só pode conter letras, dígitos e sublinhados;
  - não pode iniciar com um dígito;
- uma função em php pode ter um parâmetro opcional, definido em sua declaração → `function divide($a, $b, $verificaDivZero = true){ ... }`
- passagem por valor ou por referências → mesmo esquema do C;

EXTRAINDO DADOS DE FORMULÁRIOS

- vimos que é possível trafegar informações pelas páginas *web* de 2 formas:
  - método GET (variáveis anexas na URL da página);
  - método POST (Variáveis escritas no cabeçalho do protocolo HTTP);

EXTRAINDO DADOS DE FORMULÁRIOS - POST

- para capturar as variáveis enviadas pelo método POST, a linguagem php define um *array* global:

$phone = $_POST[‘phone’];

- as variáveis enviadas estarão nesse *array* global com sua indexação dada por seu atributo name definido no campo input do HTML;

```
<input type="text" name="phone" id="phone"/>
```

POST EXEMPLO

EXEMPLO POST

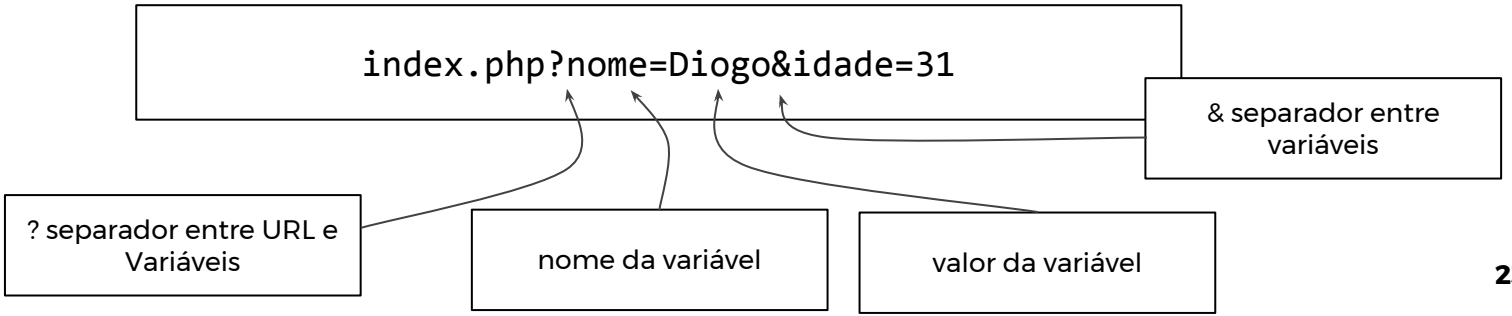
```
…
<?php
   if(!empty($_POST['nome'])){
      echo "<h1>Seu nome é: ".$_POST['nome']."</h1>";
   }
   else{
?>
      <form method="post" action="index.php">
            <label>Nome:</label>
            <input type="text" name="nome" id="nome"/>
            <input type="submit" name="enviar" id="enviar" value="Enviar"/>
      </form>
<?php
   }
?>
```

EXTRAINDO DADOS DE FORMULÁRIOS - POST

- para capturar as variáveis enviadas pelo método GET, a linguagem php define um *array* global:

$name = $_GET[‘name’];

- as variáveis capturadas pelo método GET são passadas por um formulário ou por um link diretamente na URL do arquivo, por exemplo;



GET EXEMPLO

EXEMPLO GET

```
...
<?php
      $nome  = $_GET['nome'];
      $idade = $_GET['idade'];

      if(empty($nome) || empty($idade)){
            echo "Não foram encontradas todas as variáveis de GET, utilize:
index.php?nome=Diogo&idade=23";
      }
      else{
            echo "Olá, $nome. Você tem $idade anos.";
      }
?>
```