Arrays e formulários

Manipulação de arrays e strings
Interação com o usuário através de formulários html

Material Didático organizado pela Profª. Aline de Campos

Arrays

Os arrays, também chamados de vetores, são variáveis que podem armazenar vários valores ao mesmo tempo.

Além do identificador habitual há um índice associado indicando a posição de memória em que fica armazenado o elemento do array.

Na inserção o interpretador procurará pelo último índice utilizado e o incrementará.

Arrays

Os arrays são bastante utilizados em situações onde existe uma grande quantidade de dados a serem automatizados nos scripts.

Por exemplo, uma lista de 50 nomes. Seria complicado atribuir variáveis diferentes para cada um dos nomes. Sendo assim, armazenado estes nomes em um array, com os índices de 0 a 49 e utilizando um laço de repetição poderíamos ler todos estes nomes de maneira facilitada.

Arrays

Caso não seja atribuído um índice o indexador

```php
<?php
  $vetor[] = "Zero";   // Armazenado no índice 0
  $vetor[] = "Um";     // Armazenado no índice 1
  $vetor[] = "Dois";   // Armazenado no índice 2
?>
```

O índice também pode ser representado por texto e neste caso recebe o nome de chave associativa.

```php
<?php
  $vetor["numero"] = "Zero";
?>
```

Arrays

```php
<?php
  $vetor[0] = "Zero";
  $vetor[1] = "Um";
  $vetor[2] = "Dois";
?>
```

ATENÇÃO:
Em arrays os índices sempre começam de 0.

```php
<?php
  $vetor = Array("Zero", "Um", "Dois");
  echo $vetor[1];    // Imprimirá o valor "Um";

  $vetor = Array(1,2,3, "nome"=>"João");
  echo $vetor[0];          // Imprimirá o valor 1
  echo $vetor["nome"]; // Imprimirá o valor "João"
?>
```

Arrays multidimensionais

Quando for necessário armazenar dados ainda mais complexos, podem-se utilizar os arrays multidimensionais, totalmente suportados pelo PHP. Tomemos por exemplo um edifício no qual teremos andares e em cada andar vários apartamentos. Como podemos armazenar o nome de cada morador ? Se utilizarmos o conceito de arrays multidimensionais, esta tarefa fica simples e bem organizada.

```php
<?php
        $andar[1][101] = "José da Silva";
        $andar[1][105] = "Ana Maria";
        $andar[2][210] = "Mara Soares";
        $andar[2][211] = "Darci Fernandes";
        $andar[3][301] = "Finck";

        echo $andar[1][101]."<br>";
        echo $andar[2][211]."<br>";
?>
```

Arrays multidimensionais

```php
<?php
    $produtos = array();
    $produtos[] = array(0=>'001',1=>'Arroz');
    $produtos[] = array(0=>'002',1=>'Feijão');
    $produtos[] = array(0=>'003',1=>'Farinha');
    $produtos[] = array(0=>'004',1=>'Açúcar');
    $produtos[] = array(0=>'005',1=>'Sal');

    echo $produtos[1][0]."-".$produtos[1][1]."<br/>";
    echo $produtos[3][0]."-".$produtos[3][1];
?>
```

Y \ X	0	1
0	001	Arroz
1	002	Feijão
2	003	Farinha
3	004	Açúcar
4	005	Sal

Manipulação de arrays

SINTAXE	DESCRIÇÃO
array_values (exemplo_array)	Lista os valores contidos em “exemplo_array”
asort(exemplo_array) e arsort(exemplo_array)	Ordena por ordem alfabética direta ou inversa em função dos valores
count(exemplo_array)	Retorna o número de elementos do array
ksort(exemplo_array) e krsort(exemplo_array)	Ordena por ordem alfabética direta ou inversa em função das chaves
list ($variavel1, $variavel2...) = exemplo_array	Atribui cada variável a cada um dos valores do array
next(exemplo_array), prev(exemplo_array), reset(exemplo_array) y end(exemplo_array)	Possibilita mover por dentro do array com um ponteiro para frente, para trás, início e ao fim.
each(exemplo_array)	Retorna o valor e a chave do elemento no qual nos encontramos e mexe o ponteiro ao elemento seguinte.

Manipulação de strings

Funções que são relacionadas com HTML

SINTAXE	DESCRIÇÃO
htmlspecialchars(string str);	Devolve a string fornecida, substituindo os seguintes caracteres: & para '&', " para '"', < para '<' > para '>'
htmlentities(string str);	Converte todos os caracteres da string que possuem uma representação especial em html.
nl2br(string str);	Devolve a string fornecida substituindo todas as quebras de linha ("\n") por quebras de linhas em html (" ").
get_meta_tags(string ficheiro);	Abre um ficheiro html e percorre o cabeçalho em busca de "meta" tags, Devolvendo num array todos os valores encontrados.
strip_tags(string str);	Devolve a string fornecida, retirando todas as tags html e/ou PHP encontradas.
urlencode(string str);	Devolve a string fornecida, convertida para o formato urlencode. Esta função é útil para passar variáveis para uma próxima página.
urldecode(string str);	Funciona de maneira inversa a urlencode, desta vez decodificando a string fornecida do formato urlencode para texto normal.

Manipulação de strings

Funções que são relacionadas a Arrays

SINTAXE	DESCRIÇÃO
implode(string separador, array partes); **join(string separador, array partes);**	Devolvem uma string contendo todos os elementos do array fornecido separados pela string também fornecida.
split(string padrao, string str, int [limite]);	Devolve um array contendo partes da string fornecida separadas pelo padrão fornecido, podendo limitar o número de elementos do array.
explode(string padrao, string str);	Funciona de maneira bastante semelhante à função split, com a diferença que não é possível estabelecer um limite para o número de elementos do array.

Manipulação de strings

explode(string padrao, string str);

Devolve um array contendo partes da string fornecida separadas pelo padrão fornecido.

```php
<?php
// Example 1
$pizza  = "piece1 piece2 piece3 piece4 piece5 piece6";
$pieces = explode(" ", $pizza);
echo $pieces[0]; // piece1
echo $pieces[1]; // piece2

// Example 2
$data = "foo:*:1023:1000::/home/foo:/bin/sh";
list($user, $pass, $uid, $gid, $gecos, $home, $shell) =
                                        explode(":", $data);
echo $user; // foo
echo $pass; // *
?>
```

Manipulação de strings

implode(string separador, array partes);
join(string separador, array partes);
Devolvem uma string contendo todos os elementos do array fornecido separados pela string também fornecida.

```php
<?php

$pieces = array("piece1", "piece2", "piece3", "piece4",
"piece5", "piece6");

$pizza = implode("*", $pieces);

echo $pizza;
// piece1*piece2*piece3*piece4*piece5*piece6
?>
```

Manipulação de strings

Funções para edição de Strings

SINTAXE	DESCRIÇÃO
chop(string str);	Retira espaços e linhas em branco do final da string fornecida.
ltrim(string str);	Retira espaços e linhas em branco do final da string fornecida.
trim(string str);	Retira espaços e linhas em branco do início e do final da string fornecida.
strrev(string str);	Devolve a string fornecida invertida.
strtolower(string str);	Devolve a string fornecida com todas as letras minúsculas.
strtoupper(string str);	Devolve a string fornecida com todas as letras maiúsculas.
ucfirst(string str);	Devolve a string fornecida com o primeiro caracter convertido para letra maiúscula.
ucwords(string str);	Devolve a string fornecida com todas as palavras iniciadas por letras maiúsculas.

SINTAXE	DESCRIÇÃO
str_replace(string str1, string str2, string str3);	Altera todas as ocorrências de str1 em str3 pela string str2.
str_pad(string, quant, caracter);	Preenche uma string para um certo tamanho com outra string
str_repeat(caracteres, quant);	Repete uma string
str_shuffle(string);	Mistura uma string aleatoriamente
chr(int ascii);	Devolve o caracter correspondente ao código ASCII fornecido.
ord(string string);	Devolve o código ASCII correspondente ao caracter fornecido.
strlen(string str);	Devolve o tamanho da string fornecida.

Manipulação de strings

Funções para comparação de Strings

SINTAXE	DESCRIÇÃO
similar_text(string str1, string str2, double [porcentagem]);	Compara as duas strings fornecidas e devolve o número de caracteres coincidentes. Opcionalmente pode ser fornecida uma variável, passada por referência (ver tópico sobre funções), que receberá o valor percentual de igualdade entre as strings. Esta função é case sensitive, ou seja, maiúsculas e minúsculas são tratadas como diferentes.
strcasecmp(string str1, string str2);	Compara as duas strings e Devolve 0 (zero) se forem iguais, um valor maior que zero se str1 > str2, e um valor menor que zero se str1 < str2. Esta função é case insensitive, ou seja, maiúsculas e minúsculas são tratadas como iguais.
strcasecmp(string str1, string str2);	Funciona de maneira semelhante à função strcasecmp, com a diferença que esta é case sensitive, ou seja, maiúsculas e minúsculas são tratadas como diferentes.

SINTAXE	DESCRIÇÃO
strstr(string str1, string str2); **strchr(string str1, string str2);**	As duas funções são idênticas. Procura a primeira ocorrência de str2 em str1. Se não encontrar, Devolve uma string vazia, e se encontrar Devolve todos os caracteres de str1 a partir desse ponto.
strstr(string str1, string str2);	Funciona de maneira semelhante à função strstr, com a diferença que esta é case insensitive, ou seja, maiúsculas e minúsculas são tratadas como iguais.
strpos(string str1, string str2, int [offset]);	Devolve a posição da primeira ocorrência de str2 em str1, ou zero se não houver. O parâmetro opcional offset determina a partir de qual caracter de str1 será efetuada a busca. Mesmo utilizando o offset, o valor de retorno é referente ao início de str1.
strrpos(string haystack, char needle);	Devolve a posição da última ocorrência de str2 em str1, ou zero se não houver.

Manipulação de strings

strlen()

Retorna o numero de caracteres de uma string.

```php
<?php
$str = "abcdef"; echo strlen($str); // Retorna 6
$str = " ab cd "; echo strlen($str); // Retorna 7
?>
```

```php
<?php
/* Validar se um campo tem o número necessário de
caracteres */
$nome = $_POST["comentario"];
if (strlen($nome) > 200) {
        echo "O comentário deve ter no máximo 200
                caracteres.";
}
?>
```

Manipulação de strings

substr()

Retorna uma parte de uma string.

```php
<?php
    $str = "abcdef";          // Positivo
    substr($str, 1);          // Retorna "bcdef"
    substr($str, 1, 3);       // Retorna "bcd"

    // Negativo
    substr($str, -2);         // Retorna "ef"
    substr($str, -3, 1);      // Retorna "d"

    // Acessar através de chave
    echo $str{0};             // Retorna "a"
?>
```

Manipulação de strings

```php
<?php
    // Exibir os 100 primeiros caracteres de um texto
    $texto = "Lorem ipsum dolor sit amet, consectetur
    adipiscing elit. Duis ac felis. Nulla elementum tortor
    nec libero. Vestibulum tincidunt nisi quis arcu. Nunc
    feugiat justo euismod quam. Proin tortor turpis,
    vulputate eu, ullamcorper in, euismod nec, eros.
    Pellentesque et nisl. Curabitur ultrices justo vitae
    lacus. Etiam vehicula, nulla vel faucibus eleifend,
    turpis erat auctor metus, et faucibus nibh quam id
    nisl. Mauris vestibulum. Nam at tellus sed neque
    viverra posuere. Nullam id erat at arcu tristique
    lacinia. Integer eu nibh. Cras augue velit, rutrum
    sed, posuere ut, volutpat quis, dolor. In sed ipsum
    non massa commodo bibendum. ";
    echo substr($texto, 0, 100);
?>
```

Manipulação de strings

ucfirst()

Converte para maiúsculo o primeiro caractere da string.

```php
<?php
    $str = "hello world";
    echo ucfirst($str); // Retorna "Hello world"
?>
```

strtoupper()

Converte uma string para maiúsculas.

```php
<?php
    $str = "eu sou uma string";
    echo strtoupper($str);  // Retorna "EU SOU UMA STRING"
?>
```

Manipulação de strings

strtolower()

Converte uma string para minúsculas.

```php
<?php
    $str = "EU SOU UMA STRING";
    echo strtolower($str);          // Retorna "eu sou uma string"
?>
```

Manipulação de strings

str_replace()

Substituição de caracteres em uma string.

```php
<?php
    $texto = "programadores são ******.";
    echo str_replace("*****", "felizes", $texto);
    // Retorna "Programadores são felizes.";

    $vogais = array("a", "e", "i", "o", "u");
    $apenasConsoantes =
        str_replace($vogais, "", "Hello World of PHP");
    // Retorna "Hll Wrld f PHP"

    // Resumindo
    str_replace("Substituído", "Substituição", "Onde");
?>
```

Manipulação de strings

```php
<?php
    // Filtrar um texto de palavrões
    $texto = "Nunca da certo pqp";
    $palavroes = array("fdp", "vsf", "pqp", "droga");
    echo str_replace($palavroes, "***", $texto);
    // Retorna "Nunca da certo ***"
?>
```

Manipulação de strings

strip_tags()

Retorna um string, retirando as tags HTML e/ou PHP.

```php
<?php
    $texto = "<font color='#000'>String</font>";
    echo strip_tags($texto);
    // Retorna apenas "String"
?>
```

Métodos GET e POST

```
<html>
 <body>
  <!-- USANDO MÉTODO GET -->
  <form action="pagina.php" method="GET">
   <input name="arg" type="text" value="valor">
   <input name="enviar" type="submit" value="enviar">
  </form>
 </body>
</html>
```

Métodos GET e POST

```
<html>
 <body>
  <!-- USANDO MÉTODO POST -->
  <form action="pagina.php" method="POST">
   <input name="arg" type="text" value="valor"/>
   <input name="enviar" type="submit“ value="enviar"/>
  </form>
 </body>
</html>
```

Métodos GET e POST

Após acionar o botão *submit* do primeiro formulário:

Usando o método **GET** a URL fica assim:

```
http://localhost/pagina.php?arg=valor&enviar=enviar
```

Usando o método **POST** a URL fica assim:

```
http://localhost/pagina.php
```

Recuperação de dados

Em ambos os casos os valores passados pelo formulário podem ser recuperados usando:

`$_POST` para método POST

`$_GET` para método GET

`$_REQUEST` recupera os valores em ambos os métodos e também em cookies.

Recuperação de dados

```php
<?php
// Recuperando do Form com método GET
$arg = $_GET[“argumento”];

// Recuperando do Form com método GET
$arg = $_POST[“argumento”];

// Recuperando tanto com método GET
quanto POST
$arg = $_REQUEST[“argumento”];
?>
```

Atividade prática

Criar um formulário com os seguintes itens:

- Um *formulário*

- Três campos de *texto* (nome completo, curso e instituição)

- Um campo do tipo *radio* (sexo: feminino ou masculino)

- Um campo do tipo *checkbox* (turnos de estudo: manhã, tarde ou noite)

- Um campo de *texto ampliado* (descrição das atividades)

- Um campo do tipo *select* (Idade: menos de 20 anos, entre 21 e 29 anos, mais de 30 anos)