Universidade do Estado de Santa Catarina
Departamento de Sistemas de Informação

6WEB103 - Desenvolvimento de Aplicações para a Web I

PHP – Strings e expressões regulares

Prof. Mário Ezequiel

Strings e Expressões Regulares

Com frequência precisamos limpar ou formatar a entrada de dados fornecida pelo usuário nos formulários (validação de dados, inserção em BD). Para isso, PHP tem algumas funções de manipulação de strings.

Aparando strings

- Função **trim**( ) elimina alguns caracteres do início e final da linha: retornos de carro (\n e \r), tabulações (\t e \v), caracteres de final de string (\0) e espaços;
- Funções **ltrim**( ) e **chop**( ) fazem o mesmo, porém **ltrim**( ) remove somente do início da string e **chop**( ) remove somente do final do string.

Exemplos:

```
$name = trim ($name);
$email = chop($email);
```

Formatando strings para apresentação

- Função **nl2br**( ) recebe uma string como parâmetro e insere a tag
 onde encontrar quebras de linha (Enters).

Exemplo:

```
<?php
    $str = “primeira linha,
              segunda linha.”;
   echo nl2br ($str); ?>
```

Envia para o navegador:

```
   primeira linha, <br />
   segunda linha.
```

Formatando strings para impressão

As funções **echo** e **print** fazem a mesma coisa, imprimem texto. Porém, **print** retorna um valor verdadeiro ou falso, indicando sucesso. As funções **printf** ( ) e **sprintf** ( ) fazem o mesmo que **print** ( ), porém **printf** ( ) imprime uma string formatada para o navegador e **sprintf** ( ) retorna uma string formatada. Exemplos:

```
echo “Valor total é $total”;
printf (“Valor total é %s”, $total);
printf (“Valor total é %.2f”, $total);
$valor = sprintf (“%.2f reais”, $total);
```

Alterando a caixa de strings

- strtoupper ( ) coloca a string toda em letras maiúsculas;
- strtolower ( ) coloca a string toda em letras minúsculas;
- ucfirst ( ) coloca a primeira letra da string em maiúsculo;
- ucwords ( ) coloca a primeira letra de cada palavra em maiúsculo.

Exemplos: sendo $str = “desenvolvimento Web um”:

strtoupper ($str) retorna “DESENVOLVIMENTO WEB UM”

strtolower ($str) retorna “desenvolvimento web um”

ucfirst ($str) retorna “Desenvolvimento Web um”

ucwords ($str) retorna “Desenvolvimento Web Um”

Formatando para banco de dados

Alguns caracteres geram problema se tentarmos armazenar em um banco de dados: “ e ‘ (aspas duplas e simples), \ (barra invertida) e NULL (nulo). Para escapar esses caracteres podemos inserir manualmente uma barra invertida antes de cada caracter problemático ou usar as funções **AddSlashes** ( ) e **StripSlashes** ( ).

Exemplos:

$texto = AddSlashes ($texto); → coloca ‘\’ antes de cada caracter problemático

$texto = StripSlashes ($texto); → retira as ‘\’

Unindo e dividindo strings

- **explode** ( ) divide uma string usando um caracter como separador e retorna um vetor com as partes. Exemplo:

$email = “bill@microsoft.com”;

$email_partes = explode (‘@’, $email); → $email_partes[0] conterá “bill” e $email_partes[1] conterá “microsoft.com”

- **implode** ( ) e **join** ( ) fazem o inverso de **explode** ( ), ou seja, junta as partes. Exemplo:

$new_mail = implode (‘@’, $email_partes); → $new_mail conterá “bill@microsoft.com”

Unindo e dividindo strings

- **strtok** ( ) obtém partes de uma string, uma por vez. Exemplo:

$email = “bill@microsoft.com”;

$parte = strtok ($email, ‘@’); → $parte recebe “bill”

$parte = strtok (‘@’); → agora $parte recebe “microsoft.com”

Comprimento de strings

A função **strlen** ( ) recebe uma string como argumento e retorna o tamanho desta string. Exemplo:

```
if ( strlen($email) < 6 ) {
    echo ‘Email inválido’;
    exit; }
```

Comparando strings

Podemos comparar strings de duas formas: usando a função **strcmp**( ) ou usando operadores relacionais.

- função **strcmp**(str1, str2): retorna um valor segundo a tabela abaixo:

Valor de retorno	Significado
-1	str1 precede alfabeticamente str2
0	str1 é igual a str2
1	str1 sucede str2

Exemplos:

strcmp (“braço”, “perna”) → retorna -1

strcmp (“joelho”, “joelho”) → retorna 0

strcmp (“pé”, “cabeça”) → retorna 1

Comparando strings

- Usando operadores relacionais: segundo a tabela abaixo:

Comparação	Retorna TRUE se
str1 = = str2	str1 é igual a str2
str1 != str2	str1 é diferente de str2
str1 <= str2	str1 é igual ou precede str2
str1 < str2	str1 precede str2
str1 > str2	str1 sucede str2
str1 >= str2	str1 é igual ou sucede str2

Comparando strings

Para comparações mais avançadas, utilizamos a função **preg_match**( ). Esta função recebe como primeiro parâmetro uma expressão a ser pesquisada no segundo parâmetro do tipo string, e retorna ‘1’ se encontrar o padrão, ‘0’ se não encontrar. Exemplo:

```
$str = “Agora é a hora”;
if ( preg_match(“/hora/”, $str) )    // a expressão inicia e termina com ‘/’
    print (“A string hora foi encontrada em $str”);
```

A expressão pode estar contida em uma variável. Exemplo:

```
$expr = “/hora/”;
$str = “Agora é a hora”;
if ( preg_match($expr, $str) )
    print (“A string hora foi encontrada em $str”);
```

Comparando strings

A função **preg_match**( ) entrou no lugar da função **ereg**( ) e **eregi**( ) a partir do PHP versão 5.3.

Usando-se a função **ereg**( ) ou **eregi**( ), a seguinte mensagem de erro aparece no navegador:

Deprecated: Function ereg() is deprecated

Comparando strings

Outras expressões:

Expressão	Significado
^Agor	procura “Agor” no início da string
ora$	procura “ora” no final da string
[a-zA-Z]	qualquer letra minúscula ou maiúscula
[a-z]*	zero ou mais letras minúsculas
\b	início e final de palavra
\b[a-zA-Z]*ora\b	qualquer palavra que termina em “ora”
[a-z]{n}	de ‘a’ a ‘z’ exatamente **n** vezes
[a-z]{n,m}	de **n** a **m** vezes
[[:alpha:]]	qualquer alfanumérico
0[1-9] | [12][0-9] | 3[01]	qualquer número entre 01 e 31

Comparando strings

Colocando a letra ‘i’ no final da expressão regular, a função **preg_match**( ) não faz distinção entre maiúsculas e minúsculas. Por exemplo:

$str = “Agora é a hora”;

preg_match (“/ago/ i ”, $str); → encontra “Ago”

A função **preg_replace** ( ) substitui strings. Exemplo:

$str = preg_replace (“/a/”, “A”, $str);

O exemplo acima substitui as substrings “a” por “A” na string $str

Validando email e URL

A função **filter_var( )** pode ser usada para validar e-mails, URLs entre outros. Exemplo:

```
$email = “bill@microsoft.com”;
if ( filter_var($email, FILTER_VALIDATE_EMAIL)) {
        echo “Endereço válido!”;  }

$url = “http://www.udesc.br”;
if ( filter_var($url, FILTER_VALIDATE_URL)) {
        echo “URL válida!”;  }
```

Outros filtros:
https://www.php.net/manual/en/filter.constants.php#constant.filter-validate-bool

Exercícios PHP - strings

1) Inicializar uma string com quebras de linha (ENTER) no meio do texto. Imprimir esta string sem e com a função nl2br( ).
2) Inicializar uma string com uma frase e imprimir usando as funções strtoupper( ), strtolower( ), ucfirst( ) e ucwords( ).
3) Inicializar uma string com caracteres especiais (' \ ', ' ~ ') entre o texto, aplicar AddSlashes( ) e imprimir.
4) Inicializar uma string email= "ana.paula@financeiro.com.br".

Separar por ' @ ' e imprimir as partes, separar 2º parte por ' . ' e imprimir as partes.

Referências

- Deitel, cap. 29
- Thomson, cap. 4
- https://www.php.net/manual/en/function.preg-match.php
- https://www.php.net/manual/en/function.filter-var.php