Manejo de Cadenas

Programando con PHP

Introducción

- El tratamiento de cadenas es muy importante en PHP.
  - El objetivo final del procesamiento de un fichero PHP está relacionado con la generación de documentos HTML (casi siempre)
- Existe un amplio conjunto de funciones para el manejo de cadenas.
  - Veremos las principales.

Acceso al contenido

Acceso al contenido de una cadena

- Se puede acceder a cada uno de los caracteres que componen una cadena de la misma forma que lo hacemos con los elementos de un array.
  - Haciendo referencia a su posición en la cadena.
- strlen (cadena): devuelve la longitud de la cadena.

Ejercicio

- Escribir una página en PHP que a partir de una cadena de caracteres muestre cada uno de los caracteres en una celda distinta de una tabla. Se debe indicar siempre en qué posición está el carácter en cuestión.

Carácter	Posición
S	0
a	1
l	2
u	3
d	4
o	5
s	6

```php
<center>
    <h2> Funcion <i>strlen<i> </h2>
    <?php
        $cadena = "saludos";
        echo "<table border='1' cellpadding='2' cellspacing='2'>";
        echo "<tr bgcolor='yellow'><td>Carácter</td>";
        echo "<td>Posición</td></tr>";
        for($i=0; $i<strlen($cadena);$i++)
        {
            echo "<tr align='center'><td>".$cadena[$i];
            echo "</td><td>".$i."</td></tr>";
        }
            echo "</table>";
    ?>
</center>
```

Funcion *strlen*

Carácter	Posición
s	0
a	1
l	2
u	3
d	4
o	5
s	6

Búsqueda en cadenas

Búsqueda en cadenas

- strstr (cade, buscar) y strchr (cade, buscar)
  - Buscan la cadena ‘buscar’ dentro de ‘cade’
  - Devuelve la subcadena que va desde que se encuentra ‘buscar’ hasta el final de ‘cade’.
  - Si no la encuentra devuelve una cadena vacía.
  - Diferencia entre mayúsculas y minúsculas.

```php
<?php
    $cadena='En un lugar de la mancha...';
    $buscar='lu';

    $encuentra = strstr($cadena, $buscar);
    if($encuentra != '')
    {
        echo "Cadena encontrada";
    }
    else
    {
        echo "Cadena no encontrada";
    }

?>
```

Cadena encontrada

Búsqueda en cadenas

- strrchr(cadena, carBusc)
  - Busca la **última** aparición de un carácter dentro de una cadena.
  - Aunque en 'carBusc' haya una cadena, sólo se tendrá en cuenta el primer carácter de esta cadena.
  - Devuelve la subcadena que hay entre la última aparición del carácter hasta el final de la cadena.
  - Si no lo encuentra, devuelve una cadena vacía.
  - Diferencia entre mayúsculas y minúsculas

```
<h2> Funcion <i>strchr</i> y <i>strrchr</i> </h2>
<?php
    $cadena = "saludos para todos...";
    $car1 = "do";
    $car2 = "pc";
    echo "<table border='1' cellpadding='2' cellspacing='2'>";
    echo "<tr align='center'><td bgcolor = 'pink'>cadena</td>";
    echo "<td>$cadena</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>strchr(cadena,'$car1')</td>";
    echo "<td>".strchr($cadena, $car1)."</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>strrchr(cadena,'$car1')</td>";
    echo "<td>".strrchr($cadena, $car1)."</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>strchr(cadena,'$car2')</td>";
    echo "<td>".strchr($cadena, $car2)."</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>strchr(cadena,'$car2')</td>";
    echo "<td>".strrchr($cadena, $car2)."</td></tr>";
        echo "</table>";
?>
```

Funcion *strchr* y *strrchr*

cadena	saludos para todos...
strchr(cadena,'do')	dos para todos...
strrchr(cadena,'do')	dos...
strchr(cadena,'pc')	
strchr(cadena,'pc')	para todos...

Búsqueda en cadenas

- stristr (cadena, cadenaBuscar)
  - Igual que strstr() pero no diferencia entre mayúsculas y minúsculas.
- strpos (cade1, cade2 [,despl])
  - Busca la primera 'cade2' dentro de 'cade1' desde la posición que se indique.
  - Si no se indica posición será desde el principio.
  - Devuelve la **posición** de la primera ocurrencia de 'cade2'
  - Diferencia entre mayúsculas y minúsculas.

Búsqueda en cadenas

- strrpos (cadena, caracter [,despl])
  - Busca **posición** de la última aparición del carácter en la cadena.
  - Si no lo encuentra, devuelve false.
  - Diferencia entre mayúsculas y minúsculas.

```
<h2> Funcion <i>strpos</i> y <i>strrpos</i> </h2>
<?php
    $cadena = "saludos para todos...";
    $car = "do";
    echo "<table border='1' cellpadding='2' cellspacing='2'>";
    echo "<tr align='center'><td bgcolor = 'pink'>cadena</td>";
    echo "<td>$cadena</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>strpos(cadena,'$car')</td>";
    echo "<td>".strpos($cadena, $car)."</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>strrpos(cadena,'$car')</td>";
    echo "<td>".strrpos($cadena, $car)."</td></tr>";
        echo "</table>";
?>
```

Funcion *strpos* y *strrpos*

cadena	saludos para todos...
strpos(cadena,'do')	4
strrpos(cadena,'do')	15

Búsqueda en cadenas

- strspn (cadena, máscara)
  - Obtiene la longitud de la subcadena más larga que está formada sólo por caracteres que haya en la máscara.
  - Si encuentra un carácter que no está en la máscara abandona la búsqueda.
  - Diferencia entre mayúsculas y minúsculas.
- strcspn (cadena, máscara)
  - Obtiene la longitud de la subcadena más larga formada sólo por caracteres NO contenidos en la máscara.

```
<h2> Funcion <i>strspn</i> y <i>strcspn</i> </h2>
<?php
    $cadena = "saludos para todos...";
    $car = "do";
    echo "<table border='1' cellpadding='2' cellspacing='2'>";
    echo "<tr align='center'><td bgcolor = 'pink'>cadena</td>";
    echo "<td>$cadena</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>strspn(cadena,'$car')";
    echo "</td><td>".strspn($cadena, $car)."</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>strcspn(cadena,'$car')";
    echo "</td><td>".strcspn($cadena, $car)."</td></tr>";
        echo "</table>";
?>
```

Funcion *strspn* y *strcspn*

cadena	saludos para todos...
strspn(cadena,'do')	0
strcspn(cadena,'do')	4

Búsqueda en cadenas

- strpbrk (cadena, lista_car):
  - Busca cualquier ocurrencia de la lista de caracteres pasada como parámetro dentro de la cadena.
  - Devuelve una cadena que va desde el carácter encontrado o false si no lo encuentra.
  - Diferencia entre mayúsculas y minúsculas.

Expresiones regulares

Expresiones regulares

- Las expresiones regulares son una potente herramienta que nos permite contrastar un texto con un **patrón de búsqueda**.

- Esta tarea resulta fundamental en algunos programas, y en otros puede facilitarnos increíblemente el trabajo.

Expresiones regulares

- Una expresión regular, consiste en **comparar un patrón frente a un texto**, para comprobar si el texto contiene lo especificado en el patrón.
- Ejemplo:
  - Patrón: **in**
  - Coindicen:
    - **in**tensidad
    - **cin**ta
    - **in**terior
  - Patrón: [**mp**]**adre**
  - Coindicen:
    - Mi **madre** se llama Luisa
    - Tu **padre** es jardinero

Expresiones regulares

- **Sintaxis y metacaracteres**
  - **El punto**
    - El punto representa **cualquier carácter**. Escribiendo un punto en un patrón querrás decir que ahí hay un carácter, cualquiera. Desde la A a la Z (en minúscula y mayúscula), del 0 al 9, o algún otro símbolo.
    - **Ejemplos:**
      ca.a coincide con *ca**n**a*, *ca**m**a*, *ca**s**a*, *ca**j**a*, etc…
      No coincide con *ca**st**a* ni *caa*

Expresiones regulares

- **Sintaxis y metacaracteres**
  - **Principio y fin de cadena**
    - Si queremos indicar al patrón qué es el principio de la cadena o qué es el final, debemos hacerlo con **^ para inicio y $ para final**.
  - **Ejemplos:**
    - "^*olivas*" coincide con "***olivas*** *verdes*", pero no con "*quiero olivas*"
    - "olivas$" sin embargo, va a coincidir con "quiero **olivas**" pero no con "olivas verdes"

Expresiones regulares

- **Sintaxis y metacaracteres**
  - **Cuantificadores**
    - Indicar que cierto elemento del patrón va a repetirse un **número indeterminado de veces**,
    - usaremos + o * .
    - Usando + queremos decir que el elemento anterior aparece una o más veces.
    - Usando * queremos decir que el elemento anterior aparece cero o más veces.
  - **Ejemplos:**
    "*gafas+*" coincide con "*gafa**ssss***" pero no con "*gafa*"

Expresiones regulares

- **Sintaxis y metacaracteres**
  - **puede que esté (una vez) o puede que no**: ?
    - "*coches?*" coincide con "*coche*" y con "*coches*"
  - Para definir la **cantidad de veces que se repetirá el elemento**: { }
    - "*abc{4}*" coincide con "*abcccc*", pero no con "*abc*" ni "*abcc*",
    - "*abc{1,3}*" coincide con "*abc*", "*abcc*", "*abccc*", pero no con "*abcccc*"
  - Si un parámetro queda vacío, significa "un **número indeterminado**". Por ejemplo: "*x{5,}*" significa que la x ha de repetirse 5 veces, o más.

Expresiones regulares

- **Sintaxis y metacaracteres**
  - **Rangos**
    - Los corchetes [] permiten especificar el **rango de caracteres**.
    - Basta que exista *cualquiera de ellos* para que se de la condición.
  - **Ejemplos:**
    - "*c[ao]sa*" coincide con "*casa*" y con "*cosa*"
    - "*[a-f]*" coincide con todos los caracteres alfabéticos de la "*a*" a la "*f*"
    - "*[0-9][2-6][ANR]*" coincide con "*12A*", "*35N*", "*84R*",
  - Dentro de los corchetes,
    - el símbolo ^ ya no significa inicio, si no que es un negador.

Expresiones regulares

- **Sintaxis y metacaracteres**
  - **Alternancia**
    - Para alternar entre varias opciones, usaremos el símbolo |
    - Con este mecanismo haremos un disyuntor, que nos permitirá dar varias opciones.
    - Si una de ellas coincide, el patrón será cierto.
    - **Ejemplos:**
      - "*aleman(ia|es)*" coincide con "*aleman**ia***" y con "*aleman**es***"
      - "*(norte|sur|este|oeste)*" coincide con cualquiera de los puntos cardinales.

Expresiones regulares

- **Sintaxis y metacaracteres**
  - **Agrupadores**
    - Los paréntesis nos sirven para agrupar un subconjunto.
    - Es útil para definir la alternancia, pero agrupar un subpatrón nos permite trabajar con él como si fuera un único elemento.
    - **Ejemplos:**
      "*(abc)+*" coincide con "*abc*", "*abcabc*", "*abcabcabc*", etc
      "*ca(sca)?da*" coincide con "*cascada*" y con "*cada*"

Expresiones regulares

- **Sintaxis y metacaracteres**
  - **Escapar caracteres**
    - Si queremos que en el patrón hubiese un punto, o un símbolo asterisco, sin que se interprete como metacarácter, tendremos que “escaparlo”.
    - Esto se hace poniendo una barra invertida justo antes: \. o *
    - Esto puede hacerse con cualquier carácter que quieras introducir de **forma literal**, y no interpretada.

Búsqueda con expresiones regulares

Búsqueda con expresiones regulares

- preg_match(patrón, cadena)
  - Chequea el patrón en una cadena alfanumérica. Devuelve true si coincide, o false en caso contrario. Además captura las coincidencias en un array.
  - Necesita que el patrón empiece y termine por un carácter en concreto. Puede ser cualquiera, pero se aconseja “ ` “
- preg_match_all(patron, cadena)
  - Igual que preg_match, pero almacenando todas las subcadenas que coincidan con el patrón
- preg_replace(patrón, reemplazo, cadena)
  - Nos permite reemplazar textos mediante expresiones regulares. Los argumentos pueden ser arrays, con lo que se realiza más de una sustitución con una sola función.

Ejemplos

- Comprobar si una cadena contiene alguna “r”

```php
<?php
    $cad="riesgo";
        if (preg_match("/r/", $cad)) echo "SI";
        else echo "NO";
?>
```

- Comprobar si una cadena contiene algún número

```php
<?php
    $cad="moto";
        if (preg_match("/[1-9]/", $cad)) echo "SI";
        else echo "NO";
?>
```

Ejemplos

- Comprobar si una cadena tiene dos números

```php
<?php
    $cad="riesgo33";
        if (preg_match("/[0-9]{2}/", $cad)) echo "SI";
        else echo "NO";
?>
```

- Comprobar si una cadena empieza por dos números

```php
<?php
    $cad="riesgo33";
        if (preg_match("/^[0-9]{2}/", $cad)) echo "SI";
        else echo "NO";
?>
```

Ejemplos

- Comprobar si una cadena termina por S

```php
<?php
    $cad="riesgo33S";
        if (preg_match("/S$/", $cad)) echo "SI";
        else echo "NO";
?>
```

- Comprobar que una cadena tenga un número después una letra minúscula y después un número

```php
<?php
    $cad="riesgo3T3S";
        if (preg_match("/[0-9][a-z][0-9]/", $cad)) echo "SI";
        else echo "NO";
?>
```

Ejemplos

- Comprobar que una cadena tenga un número, después una sola letra mayúscula o minúscula y después otro número

```php
<?php
    $cad="riesgo3T3S";
        if (preg_match("/[0-9]([a-z]|[A-Z])[0-9]/", $cad)) echo "SI";
        else echo "NO";
?>
```

- Comprobar que una cadena tenga un número después dos o más letras minúsculas y después la A

```php
<?php
    $cad="e3seA3S";
        if (preg_match("/[0-9][a-z]{2,}A/", $cad)) echo "SI";
        else echo "NO";
?>
```

Comparación de cadenas

Comparación de cadenas

- strcmp (cad1, cad2):
  - < cero, si cad1 es menor que cad2.
  - > cero, si cad1 es mayor que cad2.
  - Cero, si ambas son iguales.
  - Distingue entre mayúsculas y minúsculas.
- strcasecmp (cad1, cad2):
  - Igual que la anterior, pero no diferencia entre mayúsculas y minúsculas.

```php
<?php
    $cad1 = "Saludos";
    $cad2 = "saludos";
    echo "<table border='1' cellpadding='2' cellspacing='2'>";
    echo "<tr align='center'><td bgcolor = 'pink'>cadena 1</td>";
    echo "<td>$cad1</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>cadena 2</td>";
    echo "<td>$cad2</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>strcmp(cad1, cad2)";
    echo "</td><td>".strcmp($cad1, $cad2)."</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>strcasecmp(cad1, cad2)";
    echo "</td><td>".strcasecmp($cad1, $cad2)."</td></tr>";
        echo "</table>";
?>
```

Funcion *strcmp* y *strcasecmp*

cadena 1	Saludos
cadena 2	saludos
strcmp(cad1, cad2)	-1
strcasecmp(cad1, cad2)	0

Comparación de cadenas

- strncmp (cad1, cad2, num):
  - Igual que strcmp() pero sólo compara los 'num' primeros caracteres de cada cadena.

```
$cadena1='pais';
$cadena2='paisaje';

$num1=strncmp($cadena1, $cadena2, 4);
$num2=strncmp($cadena1, $cadena2, 5);
```

caracteres comparados	Resultado
4	0
5	-1

Operar con subcadenas

Operar con subcadenas

- substr (cad, inicio [,tam])
  - Devuelve la subcadena que va desde 'inicio' hasta el fin, o si se indica, el número de caracteres indicados.
- substr_replace(cad1, cad2, inicio [,tam])
  - Devuelve una cadena que es el resultado de sustituir parte cad1 por cad2.
  - La cadena original no sufre modificación.

Operar con subcadenas

- str_replace(cadbus, cadree, cadena)
  - Devuelve una cadena en la que todas las apariciones de ‘cadbus’ se cambian por ‘cadree’ en cadena.
  - La cadena original no sufre cambios.
- strtr (cadena, cadbus, cadree)
  - Igual que str_replace, pero se cambian cada uno de los caracteres de la cadena buscada, por su correspondiente en la cadena de sustitución.

```php
<?php
    $texto="Me gusta este pais";

    $nueva_cadena=str_replace($cadena1, $cadena2, $texto);

    echo "Original: $texto<br>";
    echo "Modificada: $nueva_cadena";
?>
```

Original: Me gusta este pais
Modificada: Me gusta este paisaje

```php
<?php
    $cadena = "abcdefghijkl";
    $cadB = "aei";
    $cadS = "AEI";
    echo "<table border='1' cellpadding='2' cellspacing='2'>";
    echo "<tr align='center'><td bgcolor = 'pink'>cadena</td>";
    echo "<td>$cadena</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>Patrón</td>";
    echo "<td>$cadB</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>Sustitución</td>";
    echo "<td>$cadS</td></tr>";
    echo "<tr align='center'><td bgcolor = 'pink'>";
    echo "strtr(cadena,patrón, sustitucion)";
    echo "</td><td>".strtr($cadena, $cadB, $cadS)."</td></tr>";
        echo "</table>";
?>
```

Funcion *strtr*

cadena	abcdefghijkl
Patrón	aei
Sustitución	AEI
strtr(cadena,patrón, sustitucion)	AbcdEfghIjkl

Operar con subcadenas

- substr_count(cadena, buscar)
  - Devuelve el número de veces que aparece ‘buscar’ en ‘cadena’

Modificación del contenido

Limpieza de cadenas

Limpieza de cadenas

- rtrim(cadena)
  - Devuelve la cadena sin los espacios en blanco y sin los caracteres de fin de línea que haya al final de la cadena.
- ltrim(cadena)
  - Devuelve la cadena pero elimina los espacios en blanco al principio de la cadena.
- trim (cadena)
  - Devuelve la cadena pero elimina los espacios en blanco del principio y del final de dicha cadena.

Modificación del contenido

Relleno de cadenas

Relleno de cadenas

- str_pad (cadena, long, car)
  - Rellena una cadena con un carácter de relleno (por defecto espacio en blanco) hasta que la cadena tenga la longitud deseada.
  - Opcionalmente se puede indicar el modo de relleno:
    - STR_PAD_RIGHT: relleno por la derecha (defecto)
    - STR_PAD_LEFT: relleno por la izquierda.
    - STR_PAD_BOTH: relleno por ambos lados, intenta colocar los mismos caracteres a derecha e izquierda.

```php
<?php
    $texto="Cadena";

    $nueva_cadenaD=str_pad($texto, 20, 'r', STR_PAD_RIGHT);
    $nueva_cadenaI=str_pad($texto, 20, 'r', STR_PAD_LEFT);
    $nueva_cadenaB=str_pad($texto, 20, 'r', STR_PAD_BOTH);

    echo "Original: $texto<br>";
    echo "Modificada Derecha: $nueva_cadenaD<br>";
    echo "Modificada Izquierda: $nueva_cadenaI<br>";
    echo "Modificada Ambos: $nueva_cadenaB<br>";

?>
```

Original: Cadena
Modificada Derecha: Cadenarrrrrrrrrrrrr
Modificada Izquierda: rrrrrrrrrrrrCadena
Modificada Ambos: rrrrrrCadenarrrrrr

Modificación del contenido

Conversión entre mayúsculas y minúsculas

Conversión Mayúsculas Minúsculas

- strtolower(cadena)
  - Convierte una cadena de caracteres a minúsculas.
- strtoupper(cadena)
  - Convierte una cadena de caracteres a mayúsculas.
- ucfirst(cadena)
  - Convierte a mayúsculas el primer carácter de una cadena.
- ucwords(cadena)
  - Convierte a mayúsculas el primer carácter de cada palabra de la cadena.

Modificación del contenido

División de cadenas

División de cadenas

- strtok (cadena, divisor)
  - Divide una cadena en subcadenas.
  - Utiliza ”divisor” como carácter de división.
  - La primera vez que se llama a la función, devuelve el primer trozo obtenido.
  - Las siguientes veces que se llama a la función **NO** hay que pasar de nuevo la cadena a dividir.

```php
<?php
    $patron = " ";
    $cadena = "dato1 dato2 dato3 dato4";
    $datos = strtok($cadena,$patron);
   while ($datos)
    {
        echo "<tr align='center'>";
        echo "<td bgcolor='pink'>subcadena</td>";
        echo "<td>$datos</td></tr>";
        $datos = strtok($patron);
    }
?>
```

División de cadenas

cadena	dato1 dato2 dato3 dato4
strotok(cadena,$patron)	
subcadena	dato1
subcadena	dato2
subcadena	dato3
subcadena	dato4

División de cadenas

- chunk_split(cadena [,longitud] [,separador])
  - Divide una cadena en porciones de menor tamaño.
  - Se puede indicar el tamaño de las subcadenas o el separador que se debe usar.
  - Devuelve un array con las subcadenas obtenidas.
- explode(patron, cadena)
  - Igual que strtok
- implode (nexo, array)
  - Devuelve una cadena resultado de unir todos los elementos de un array. Utiliza 'nexo' como unión.

Funciones relacionadas con HTML

HTML

- Gran parte de las cadenas con las que trabaja PHP tienen como función convertirse en parte de un documento HTML que será enviado al usuario.
- PHP incluye algunas funciones que evitan los problemas de transformación de texto en código PHP.

HTML

- htmlspecialchars(cadena)
  - Convierte los caracteres con un significado especial en su traducción en HTML

Carácter	Traducción HTML
&	&amp;
“	&quot;
<	&lt;
>	&gt;

HTML

- htmlentities (cadena)
  - Igual que la anterior, pero traduce todos los caracteres a su correspondiente HTML.

Otras funciones con cadenas

Otras funciones

- chr(entero)
  - Recibe un codigo ASCII y devuelve el carácter asociado a dicho código.
- count_chars(cadena [,modo])
  - Cuenta el numero de apariciones de cada carácter en la cadena.
  - El modo en que lo devuelve depende del segundo parámetro:

0	**Matriz asociativa**
1	Sólo los caracteres que aparecen más de 0 veces.
2	Sólo los caracteres que no aparecen.
3	Cadena con caracteres que hay
4	Cadena con caracteres que no hay

Otras funciones

- strrev(cadena)
  - Devuelve la cadena invertida.
- str_repeat(cadena, veces)
  - Devuelve una cadena con la cadena que se le pasa repetida tantas veces como se pase en el 2º parámetro.

Manejo de Cadenas

Programando con PHP