

Bloque B

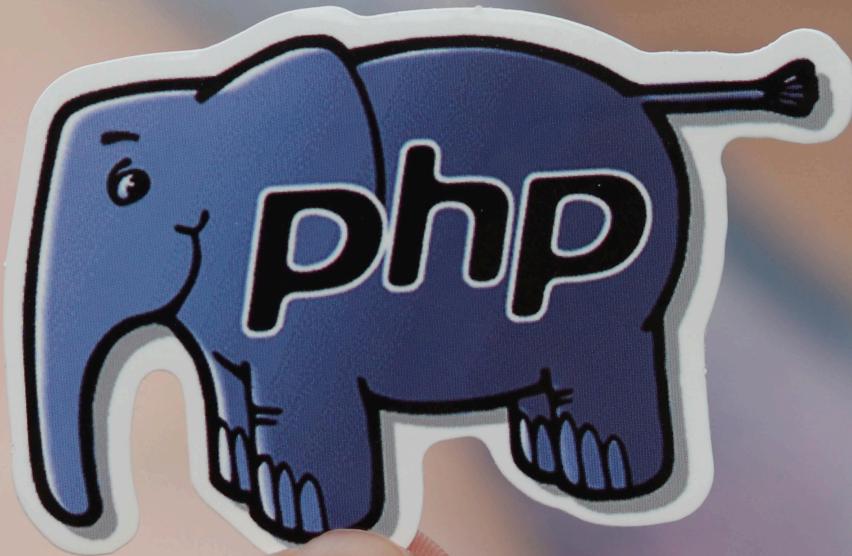
Páginas Web Dinámicas

Unidad 5. Funciones integratedas



Contenidos

1. Introducción
2. Funciones de cadenas
3. Funciones numéricas
4. Funciones de arrays
5. Constantes
6. Función header()
7. Funciones de archivos
8. Resumen
9. Actividad propuesta
10. Referencias



1. Introducción

Introducción

Esta unidad introduce un conjunto de funciones que están incorporadas en el intérprete PHP. Cada función realiza una tarea específica.

Introducción

Las definiciones de las funciones incorporadas están integradas en el intérprete PHP, lo que significa que no necesitan ser incluidas en una página PHP antes de ser llamadas.

Fueron diseñadas para realizar tareas que los desarrolladores web a menudo necesitan realizar al crear páginas web dinámicas y evitarles tener que escribir sus propias funciones para realizar esas tareas.

Introducción

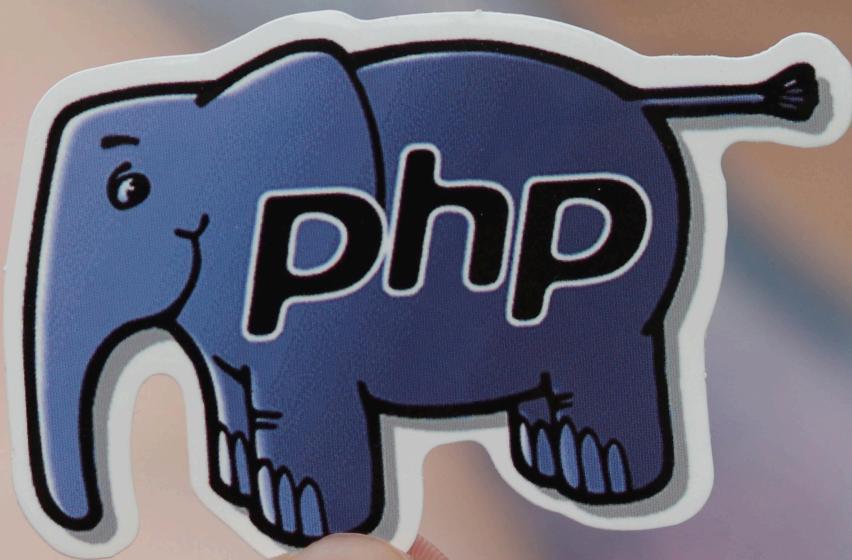
Para llamar a una función incorporada, se necesita saber su nombre, qué parámetros tiene y qué datos devolverá. Por lo tanto, esta unidad contiene varias tablas que muestran los nombres de las funciones y los parámetros, seguidos de descripciones de las funciones, y qué valores devuelven.

Introducción

El primer conjunto de funciones se agrupa en función de los tipos de datos con los que se utiliza para trabajar: cadenas, números y arrays. Más adelante en la unidad, también verás:

- Cómo crear constantes (que son como variables cuyos valores no pueden cambiar una vez establecidos).
- Una función que se utiliza para controlar las cabeceras HTTP que se envían de vuelta al navegador cuando el intérprete PHP devuelve una página que ha sido solicitada.
- Un conjunto de funciones que te permiten obtener información sobre archivos en el servidor.

Las funciones que conozcas en esta unidad serán utilizadas a lo largo del resto de este módulo.



2. Funciones de cadenas

Mayúsculas/minúsculas y comprobación de la longitud

Estas funciones transforman el texto en mayúsculas o minúsculas y cuentan el número de caracteres o palabras de la cadena.

Las siguientes funciones están diseñadas para trabajar con texto (el tipo de datos cadena). Toman una cadena como argumento, la actualizan y devuelven la cadena modificada.

Por ejemplo, la función `strtolower()` toma una cadena y convierte todo el texto a minúsculas. A continuación, devuelve el valor actualizado.

Mayúsculas/minúsculas y comprobación de la longitud

Cambio de mayúsculas y minúsculas

FUNCTION	DESCRIPTION
<code>strtolower(\$string)</code>	Returns a string with all characters in lowercase.
<code>strtoupper(\$string)</code>	Returns a string with all characters in uppercase.
<code>ucwords(\$string)</code>	Returns a string with the first letter of every word in uppercase.

La función `ucwords()` se llama así porque es un acrónimo de "Uppercase Words"

Mayúsculas/minúsculas y comprobación de la longitud

Contar caracteres y palabras

FUNCTION	DESCRIPTION
<code>strlen(\$string)</code>	Returns the number of characters in the string. Spaces and punctuation count as characters. (See also <code>mb_strlen()</code> on p210-211 for multibyte characters.)
<code>str_word_count(\$string)</code>	Returns the number of words in the string.

Ejemplo: conversión mayúsculas/minúsculas y contando caracteres

El siguiente ejemplo realiza varias operaciones sobre la cadena de texto '*Home sweet home*' y luego incluye dos archivos, `header.php` y `footer.php`, para estructurar la salida HTML.

1. Una cadena que dice "Home sweet home" se almacena en una variable llamada `$texto`. Esto se utilizará como argumento cuando se llame a cada una de las funciones.
2. Se llama a la función `strtolower()`. La función convierte el texto a minúsculas y devuelve ese valor. El valor que devuelve la función se escribe en la página utilizando la abreviatura del comando echo.

Ejemplo: conversión mayúsculas/minúsculas y contando caracteres

3. La función `strtoupper()` devuelve la cadena en mayúsculas.
4. La función `ucwords()` devuelve la cadena con la primera letra de cada palabra en mayúsculas.
5. La función `strlen()` cuenta cuántos caracteres hay en la cadena y devuelve ese número.
6. La función `str_word_count()` cuenta el número de palabras de la cadena y devuelve ese número.

Ejemplo: conversión mayúsculas/minúsculas y contando caracteres

```
<?php  
① $text = 'Home sweet home';  
?>  
<?php include 'includes/header.php'; ?>  
<p>  
    <b>Lowercase:</b>  
② <?= strtolower($text) ?><br>  
    <b>Uppercase:</b>  
③ <?= strtoupper($text) ?><br>  
    <b>Uppercase first letter:</b>  
④ <?= ucwords($text) ?><br>  
    <b>Character count:</b>  
⑤ <?= strlen($text) ?><br>  
    <b>Word count:</b>  
⑥ <?= str_word_count($text) ?>  
    </p>  
<?php include 'includes/footer.php'; ?>
```

Ejemplo: conversión mayúsculas/minúsculas y contando caracteres

Lowercase: home sweet home

Uppercase: HOME SWEET HOME

Uppercase first letter: Home Sweet Home

Character count: 15

Word count: 3

Ejemplo: conversión mayúsculas/minúsculas y contando caracteres

Crea un pequeño script que podría ser parte de una aplicación de blog o gestión de contenidos, donde el texto de las entradas de blog necesita ser procesado y analizado antes de mostrarse.

El script transformará y analizará el texto utilizando las funciones `strtoupper()`, `ucwords()`, `strlen()` y `str_word_count()`, y presentará los resultados de una manera que podría ser útil para los administradores del blog.

Ejemplo: conversión mayúsculas/minúsculas y contando caracteres

Pasos a realizar:

1. Define una cadena de texto que simule el contenido de una entrada de blog.
2. Aplica las funciones `strtoupper()`, `ucwords()`, `strlen()` y `str_word_count()` a la cadena.
3. Calcula la longitud del texto sin contar los espacios.

Ejemplo: conversión mayúsculas/minúsculas y contando caracteres

4. Presenta un resumen del análisis que incluya:

- El texto original.
- El texto en mayúsculas.
- El texto con cada palabra capitalizada.
- La longitud del texto en caracteres.
- La longitud del texto en caracteres sin espacios.
- La cantidad de palabras en el texto.

Ejemplo: conversión mayúsculas/minúsculas y contando caracteres

Extensión de la Actividad:

Añade funcionalidades adicionales, como:

- Contar la frecuencia de cada palabra en el texto.
- Detectar y marcar palabras clave predefinidas (por ejemplo, "importante", "procesar").
- Generar un resumen del texto mostrando solo las primeras 50 palabras.

Búsqueda de caracteres en una cadena

Estas funciones buscan uno o varios caracteres en una cadena. Si encuentran una coincidencia, devuelven la posición de ese carácter. Si no encuentran ninguna coincidencia, devuelven *false*.

Búsqueda de caracteres en una cadena

Cada carácter de una cadena tiene una **posición**; un número que empieza por 0.

Así, el primer carácter está en la posición 0, el segundo en la 1, y así sucesivamente.



Búsqueda de caracteres en una cadena

Cuando se busca un conjunto de caracteres dentro de una cadena, esos caracteres se denominan **subcadena**.

Algunas de las funciones **distinguen entre mayúsculas y minúsculas (case-sensitive)**, por lo que sólo se encuentra una coincidencia si la cadena y la subcadena tienen la misma combinación de caracteres en mayúsculas y minúsculas.

Las tres últimas funciones, marcadas con un asterisco fueron añadidas en PHP 8; todas ellas distinguen entre mayúsculas y minúsculas.

NOTA: Los parámetros opcionales se muestran entre corchetes.

Búsqueda de caracteres en una cadena

FUNCTION	DESCRIPTION
<code>strpos(\$string, \$substring[, \$offset])</code>	Returns position of first match for substring (case-sensitive). If offset is used, it only looks <i>after</i> this character position.
<code>stripos(\$string, \$substring[, \$offset])</code>	Case-insensitive version of strpos().
<code>strrpos(\$string, \$substring[, \$offset])</code>	Returns position of last match for substring (case-sensitive).
<code>strripos(\$string, \$substring[, \$offset])</code>	Case-insensitive version of strrpos().
<code>strrstr(\$string, \$substring)</code>	Returns text from first occurrence of a substring (including the substring) to the end of the string (case-sensitive).
<code>stristr(\$string, \$substring)</code>	Case-insensitive version of strrstr().
<code>substr(\$string, \$offset[, \$characters])</code>	Returns characters from the position specified in \$offset to the end of the string. If the \$characters parameter is used, it specifies the number of characters to return after \$offset.
<code>* str_contains(\$string, \$substring)</code>	Checks if a substring is found in a string, returns true/false.
<code>* str_starts_with(\$string, \$substring)</code>	Checks if string starts with substring, returns true/false.
<code>* str_ends_with(\$string, \$substring)</code>	Checks if string ends with substring, returns true/false.

Ejemplo: comprobación de caracteres en una cadena

El siguiente ejemplo define una cadena y usa varias funciones de PHP para buscar subcadenas y extraer partes de la cadena original.

Las funciones incluyen búsquedas sensibles e insensibles a mayúsculas y minúsculas, búsqueda de la primera y última aparición de subcadenas, y extracción de subcadenas.

Los resultados se muestran en una página HTML entre un encabezado y un pie de página incluidos externamente.

Ejemplo: comprobación de caracteres en una cadena

1. La cadena *Home sweet home* se almacena en `$text`.
2. Se llama a la función `strpos()` para buscar la primera vez que aparece la subcadena *ho* en la cadena. Devuelve 11.
3. Se llama a la función `stripos()` para buscar la primera vez que aparece la subcadena *me* después de la posición 5. Devuelve 13.
4. Se llama a la función `strrpos()` para encontrar la última vez que se encuentra la subcadena *Ho*. Devuelve 0 porque distingue entre mayúsculas y minúsculas.

Ejemplo: comprobación de caracteres en una cadena

5. Se llama a la función `stripos()` para encontrar la última vez que se encontró la subcadena *Ho*. Devuelve 11 porque no distingue entre mayúsculas y minúsculas.
6. Se llama a la función `strstr()` para obtener el texto de la primera aparición de la subcadena *ho*. Devuelve *home*.
7. Se llama a la función `stristr()` para obtener el texto de la primera aparición de *ho*. Devuelve *Home sweet home* porque no distingue entre mayúsculas y minúsculas.
8. Se llama a la función `substr()` y devuelve cinco caracteres, empezando por el carácter en la quinta posición.

Ejemplo: comprobación de caracteres en una cadena

```
<?php  
① $text = 'Home sweet home';  
    ?> ...  
    <b>First match (case-sensitive):</b>  
② <?= strpos($text, 'ho') ?><br>  
    <b>First match (not case-sensitive):</b>  
③ <?= stripos($text, 'me', 5) ?><br>  
    <b>Last match (case-sensitive):</b>  
④ <?= strrpos($text, 'Ho') ?><br>  
    <b>Last match (not case-sensitive):</b>  
⑤ <?= strripos($text, 'Ho') ?><br>  
    <b>Text after first match (case-sensitive):</b>  
⑥ <?= strstr($text, 'ho') ?><br>  
    <b>Text after first match (not case-sensitive):</b>  
⑦ <?= striistr($text, 'ho') ?><br>  
    <b>Text between two positions:</b>  
⑧ <?= substr($text, 5, 5) ?><br>
```

Ejemplo: comprobación de caracteres en una cadena

First match (case-sensitive): 11

First match (not case-sensitive): 13

Last match (case-sensitive): 0

Last match (not case-sensitive): 11

Text after first match (case-sensitive): home

Text after first match (not case-sensitive): Home sweet home

Text between two positions: sweet

Ejemplo: comprobación de caracteres en una cadena

Desarrolla un script en PHP que analice y manipule el contenido de un artículo web. El script deberá identificar ciertas subcadenas dentro del texto y realizar diferentes operaciones basadas en la detección de estas subcadenas.

Ejemplo: comprobación de caracteres en una cadena

Instrucciones:

1. Definir el Contenido:

- Define una cadena de texto que simule el contenido de un artículo web.

2. Detección y Análisis de Subcadenas:

- Detectar la primera y última aparición de una palabra específica.
- Comprobar si el artículo contiene ciertas palabras clave.
- Extraer partes del texto basadas en subcadenas específicas.
- Comprobar si el texto comienza o termina con ciertas palabras.

3. Presentación de Resultados:

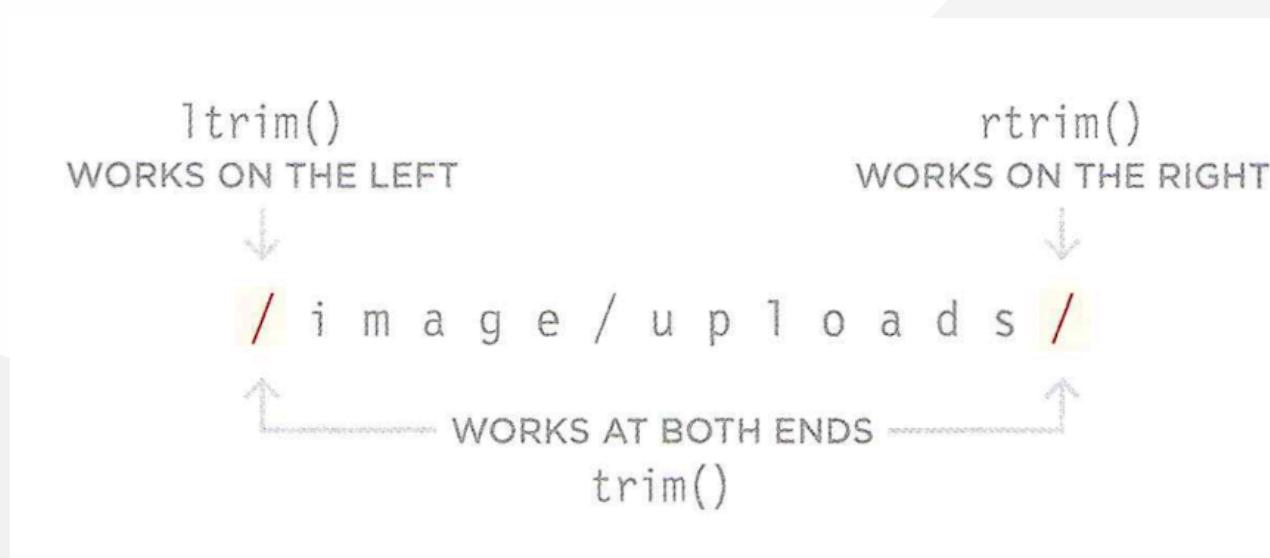
- Mostrar los resultados del análisis de manera clara y ordenada.

Eliminar y sustituir caracteres

Estas funciones pueden eliminar caracteres específicos (incluidos los espacios en blanco), sustituir caracteres (como una herramienta de búsqueda y sustitución) y repetir una cadena un número determinado de veces.

Eliminar y sustituir caracteres

Las funciones de **recorte** (`trim`) eliminan caracteres de una cadena. Pueden buscar caracteres específicos al principio y/o al final de la cadena y eliminarlos si están presentes.



Si no especifica los caracteres que desea eliminar, las funciones de recorte eliminan cualquier espacio en blanco que se encuentre al principio y/o al final de la cadena, incluidos espacios, tabulaciones (`\t`), retornos de carro (`\r`) y saltos de línea (retornos suaves `\n`).

Eliminar y sustituir caracteres

Las funciones de **sustitución** buscan caracteres en una cadena. Si encuentra una coincidencia, sustituye esos caracteres por otros nuevos.

La función **repetir** repite una cadena un número fijo de veces.

Eliminar y sustituir caracteres

FUNCTION	DESCRIPTION
<code>ltrim(\$string[, \$delete])</code>	Remove whitespace from the left-hand side of the string. If specified, \$delete provides a set of characters that should be removed if they are found at the start of the string. It is case-sensitive.
<code>rtrim(\$string[, \$delete])</code>	Removes whitespace from the right-hand side of the string.
<code>trim(\$string[, \$delete])</code>	Removes whitespace from the left and right-hand sides of the string.
<code>str_replace(\$old, \$new, \$string)</code>	Replaces the substring \$old with the one in \$new (case-sensitive).
<code>str_ireplace(\$old, \$new, \$string)</code>	Replaces the substring \$old with the one in \$new (case-insensitive).
<code>str_repeat(\$string, \$repeats)</code>	Repeats the string a specified number of times.

Ejemplo: reemplazando caracteres en una cadena

El siguiente ejemplo manipula una cadena de texto ('/images/uploads/') realizando varias operaciones: elimina caracteres específicos con `trim`, `ltrim`, y `rtrim`, reemplaza subcadenas con `str_replace` y `str_ireplace`, y repite la cadena con `str_repeat`.

Ejemplo: reemplazando caracteres en una cadena

1. La ruta '*/images/uploads/*' es una cadena que se almacena en una variable llamada `$text`.
2. La función `trim()` devuelve la cadena con `/` eliminado de la izquierda y derecha del texto.
3. La función `ltrim()` devuelve la cadena con `/` eliminado de la parte izquierda.
4. La función `rtrim()` devuelve la cadena con `s/` eliminado del lado derecho.

Ejemplo: reemplazando caracteres en una cadena

5. La función `str_replace()` devuelve la cadena con las letras `images` sustituidas por las letras `img`. Distingue entre mayúsculas y minúsculas.
6. La función `str_ireplace()` devuelve la cadena con las letras `IMAGES` sustituidas por las letras `img`. La búsqueda por subcadena distingue entre mayúsculas y minúsculas, por lo que encontraría tanto `IMAGES` como `images` y las sustituiría por `img`.
7. La función `str_repeat()` devuelve la cadena con todos los caracteres repetidos dos veces.

Ejemplo: reemplazando caracteres en una cadena

```
<?php  
① $text = '/images/uploads/';  
    ?> ...  
    <b>Remove '/' from both ends:</b><br>  
② <?= trim($text, '/') ?><br>  
    <b>Remove '/' from the left of the string:</b><br>  
③ <?= ltrim($text, '/') ?><br>  
    <b>Remove 's/' from the right of the string:</b><br>  
④ <?= rtrim($text, 's/') ?><br>  
    <b>Replace 'images' with 'img':</b><br>  
⑤ <?= str_replace('images', 'img', $text) ?><br>  
    <b>As above but case-insensitive:</b><br>  
⑥ <?= str_ireplace('IMAGES', 'img', $text) ?><br>  
    <b>Repeat the string:</b><br>  
⑦ <?= str_repeat($text, 2) ?></p>
```

Ejemplo: reemplazando caracteres en una cadena

Remove '/' from both ends:

images/uploads

Remove '/' from the left of the string:

images/uploads/

Remove 's/' from the right of the string:

/images/upload

Replace 'images' with 'img':

/img/uploads/

As above but case-insensitive:

/img/uploads/

Repeat the string:

/images/uploads//images/uploads/

Ejemplo: reemplazando caracteres en una cadena

Escribe un script PHP que simule la entrada de datos de un usuario y luego utilice las funciones de manipulación de cadenas para limpiar y formatear estas entradas.

Ejemplo: reemplazando caracteres en una cadena

Instrucciones

1. Configura el Script PHP:

- Crea un archivo PHP llamado `procesar_datos.php`.
- Define variables para simular datos de entrada de un usuario: `$nombre`,
`$correo`, y `$mensaje`.

2. Muestra los Datos Originales:

- Utiliza `echo` para mostrar en pantalla los datos originales.

Ejemplo: reemplazando caracteres en una cadena

3. Realiza las Siguientes Operaciones de Manipulación de Cadenas:

- Eliminar Espacios en Blanco Adicionales: Utiliza `trim()` para eliminar los espacios en blanco al inicio y al final de las cadenas.
- Asegurar que el Correo Está en Minúsculas: Aplica `strtolower()` para convertir toda la cadena a minúsculas.
- Reemplazar Ciertas Palabras en el Mensaje: Utiliza `str_replace()` para reemplazar palabras inapropiadas con "*****".
- Reemplazo Insensible a Mayúsculas/Minúsculas: Emplea `str_ireplace()` para reemplazar "urgente" con "Prioridad Alta".
- Repetir una Cadena para Enfatizar: Añade "!!!" al final del mensaje usando `str_repeat()`.

Ejemplo: reemplazando caracteres en una cadena

4. Muestra los Datos Procesados:

- Utiliza echo para mostrar en pantalla los datos después de haber sido procesados.

Funciones de cadena multibyte

Algunas de las funciones de cadena que hemos visto hasta ahora devuelven un resultado erróneo si se utilizan con caracteres multibyte.

Las funciones de cadena multibyte que se muestran a continuación admiten todos los caracteres de UTF-8.

Funciones de cadena multibyte

Cuando el texto se codifica utilizando UTF-8, algunos caracteres utilizan más de un byte de datos. Por ejemplo, el símbolo £ utiliza dos bytes y € utiliza tres.

Si utiliza caracteres multibyte como argumentos para algunas de las funciones de cadena, pueden producir un resultado incorrecto (en el ejemplo a continuación se muestran casos de resultados inexactos).

Funciones de cadena multibyte

Las funciones de cadena multibyte que se muestran a continuación tienen los mismos nombres que las funciones que has conocido hasta ahora en este capítulo, pero van precedidas de los caracteres *mb*.

Algunas funciones de cadena no tienen equivalente multibyte, por ejemplo, `trim()` y `str_replace()`. Estas funcionan con UTF-8 siempre que se haya establecido como codificación de caracteres por defecto en los archivos `php.ini` o `.htaccess`.

Funciones de cadena multibyte

FUNCTION	DESCRIPTION
<code>mb_strtoupper(\$string)</code>	Returns string with all characters in uppercase.
<code>mb_strtolower(\$string)</code>	Returns string with all characters in lowercase.
<code>mb_strlen(\$string)</code>	Returns the number of characters in the string.
<code>mb_strpos(\$string, \$substring[, \$offset])</code>	Returns the position of the first place the substring is found (case-sensitive). If an <code>\$offset</code> is specified, it only looks after this character position.
<code>mb_stripos(\$string, \$substring[, \$offset])</code>	Case-insensitive version of <code>mb_strpos()</code> .
<code>mb_strrpos(\$string, \$substring[, \$offset])</code>	Returns position of last match for substring (case-sensitive).
<code>mb_stripos(\$string, \$substring[, \$offset])</code>	Case-insensitive version of <code>mb_strrpos()</code> .
<code>mb_substr(\$string, \$start[, \$characters])</code>	Returns text from the first occurrence of a substring (including the substring) to the end of the string (case-sensitive).
<code>mb_stristr(\$string, \$substring)</code>	Case-insensitive version of <code>mb_substr()</code> .
<code>mb_substr(\$string, \$start[, \$characters])</code>	Returns characters from position specified in <code>\$start</code> to the end of the string. If <code>\$characters</code> is specified, it returns this number of characters after <code>\$start</code> .

Ejemplo: Uso de funciones de cadenas multibyte

Este ejemplo utiliza funciones de cadena con el símbolo £ , que UTF-8 necesita dos bytes de datos para codificar.

1. Se crea una cadena utilizando el símbolo £ y se almacena en \$text . Tiene una longitud de 11 caracteres.
2. La función strlen() funciona contando el número de bytes necesarios para representar una cadena, no el número de caracteres de la cadena. Por eso dice que hay 12 caracteres en la cadena (no 11).

Ejemplo: Uso de funciones de cadenas multibyte

3. La función `mb_strlen()` tiene en cuenta la codificación que utiliza el intérprete de PHP y muestra el número correcto de caracteres de la cadena como 11.
4. La función `strpos()` encuentra la primera posición de 444. La posición se calcula utilizando el número de bytes antes de encontrar la subcadena (no el número de caracteres). Devuelve 9, en lugar de 8.
5. `mb strpos()` encuentra la primera posición de 444, devolviendo correctamente el número 8.

Ejemplo: Uso de funciones de cadenas multibyte

```
<?php  
① $text = 'Total: £444';  
    ?> ...  
    <b>Character count using <code>strlen()</code>:</b>  
② <?= strlen($text) ?><br>  
    <b>Character count using <code>mb_strlen()</code>:</b>  
③ <?= mb_strlen($text) ?><br>  
    <b>First match of 444 <code>strpos()</code>:</b>  
④ <?= strpos($text, '444') ?><br>  
    <b>First match of 444 <code>mb_strpos()</code>:</b>  
⑤ <?= mb_strpos($text, '444') ?><br>
```

Ejemplo: Uso de funciones de cadenas multibyte

Character count using `strlen()`: 12

Character count using `mb_strlen()`: 11

First match of 444 `strpos()`: 9

First match of 444 `mb_strpos()`: 8

Ejemplo: Uso de funciones de cadenas multibyte

Sobre el ejemplo anterior, realiza las siguientes modificaciones:

- Modifica el texto en la variable `$text` del siguiente código para incluir caracteres de distintos idiomas (por ejemplo, chino, japonés) y explica cómo se comportan las funciones.