

# Generación de código HTML: Formas de que php escriba dentro de HTML



## Formas de que PHP escriba dentro de HTML

Es posible la intercalación de órdenes en lenguaje **PHP**, **alternándolas** dentro de una página escrita en lenguaje **HTML**.

Pueden abrirse y cerrarse las etiquetas de PHP **tantas veces como sea necesario**

# Apertura y cierre de las etiquetas PHP

Las etiquetas PHP:

- Puede abrirse y cerrarse en la misma línea en que abrió, o puede cerrarse en otra línea diferente. Es indistinto.
- Puede intercalarse dentro de etiquetas HTML preexistentes.

```
<!DOCTYPE html>
<html lang="es">

<head>
    <meta charset="UTF-8">
    <title>Escribir en HTML</title>
</head>

<body>
    <h1>Esto fue escrito estáticamente en HTML</h1>
    <?php
print("<h2>Hola mundo! Esto lo escribió el intérprete de PHP</h2>");
?>
    <p>Esto ya estaba escrito en código HTML.</p>
    <?php print("<p>Esto también lo escribió el software intérprete de PHP.</p>"); ?>
    <p>
        <a href="index.php">
            <?php
            print("Volver al index del sitio, escrito por PHP");
            ?>
        </a>
    </p>
</body>

</html>
```

# Formas de que PHP escriba dentro de HTML

## 1. Apertura y cierre estándar:

```
<?php xxxx ?>
```

```
<?php  
xxxx  
?>
```

- Ésta es la única sintaxis universal: funciona siempre. Es la única forma recomendada y la que vamos a usar

# Formas de que PHP escriba dentro de HTML

## 2. Apertura y cierre corto:

```
<? xxxx ?>
```

```
<?  
xxxx  
?>
```

- Esta sintaxis se conoce como **short tags** (etiquetas cortas). Fue muy usada en los primeros años de PHP, **pero no es estándar y no la utilizaremos**
- No todas las configuraciones del intérprete de PHP habilitan su uso, por lo que un código que utilice esta sintaxis puede dejar de funcionar al ser ubicado en un servidor con una configuración más estricta.

# Formas de que PHP escriba dentro de HTML

Generar etiquetas HTML mediante:

**echo**

**print()**

# Función print()

- La función **print()** le indica al software intérprete de PHP que “escriba” en el código fuente de la página que devolverá al navegador aquello que pongamos entre sus paréntesis.
- Si lo que deseamos es que se escriba en el código de la página un texto, literalmente, debemos escribirlo entre comillas dentro de sus paréntesis.

Admite **comillas dobles** y **comillas simples**



Formas de  
que PHP  
escriba  
dentro de  
HTML

**Ejemplo:**

```
<?php  
    print ("hola");  
?>
```

# Formas de que PHP escriba dentro de HTML

- Si sólo tuviéramos que escribir texto y nunca etiquetas HTML, no tendríamos más problemas.
- Pero como debemos encerrar **entre comillas** el texto a mostrar, se nos planteará un problema a la hora de escribir código HTML que, a su vez **tenga comillas dentro**.

- Este ejemplo **generará un error**, pues la comilla ubicada después del signo = está cumpliendo, la función de **cerrar la primera de las comillas**.

- Por lo tanto, el tramo de texto se da por concluido y el resto, el que sigue a esa comilla, el intérprete de PHP no sabe cómo tratarlo, y muestra un mensaje de error.

**Error:** comillas dobles dentro de comillas dobles

```
<?php  
print("<h1 class='portada'> Bienvenidos</h1>");  
?>
```

## Escapar o desactivar las comillas con \

- Una posible solución al problema de las comillas es desactivarlas todas las comillas dobles intermedias, una por una, para que no den por concluida la cadena de texto antes de que lleguemos a la última comilla doble que indica el término de la función print.
- El carácter de escape es la barra invertida \ y sirve para NO ejecutar el carácter que le sigue inmediatamente como si fuera parte de una orden del lenguaje PHP, sino que lo considera como una letra más que debe ser escrita literalmente.

Esto funciona muy bien en frases cortas.

Pero NO cuando tenemos que imprimir largos bloques de código HTML, ya que es muy probable que esos bloques ya los tengamos escritos previamente, generados por nuestro editor de código HTML y que poseerán numerosas comillas dobles

El ejemplo anterior quedará así:

```
<?php  
print("<h1 class=\"portada\"> Bienvenidos</h1>");  
?>
```

## Comillas simples

para delimitar el inicio y  
final del bloque de texto  
a imprimir

```
<?php  
    print('<h1 class="portada"> Bienvenidos</h1>');  
?>
```

¡Y problema solucionado de momento!

# Comando echo

- Este comando también puede utilizar comillas simples o dobles para delimitar lo que **va a imprimir**, de la misma manera que print.
- A diferencia de print, no es habitual envolver entre paréntesis lo que se escribirá.

```
<?php

echo "¡Hola Mundo entre comillas dobles!";

echo '<html>
    <head>
        <title>Envuelvo entre comillas simples</title>
    </head>
    <body>
        "Tiene comillas dobles, "muchas comillas", y no importa"
    </body>
</html>';

?>
```



# Formas de que PHP escriba dentro de HTML

- El código que escribirá puede estar dividido en múltiples líneas.
- PHP ignora tanto los saltos de línea como los espacios en blanco.
- Para dar por terminada una sentencia se agrega un punto y coma al final de la línea.

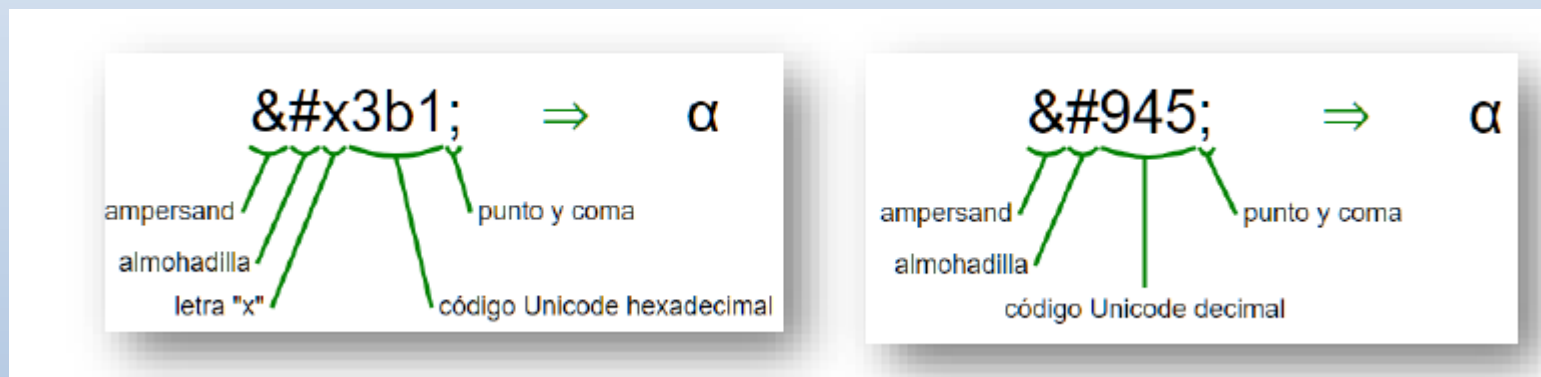
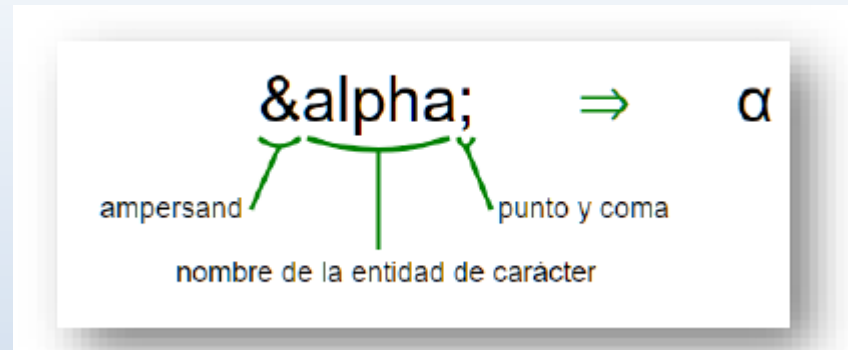
# Entidades HTML

Cuando estemos insertando texto en documentos HTML puede darse el caso de que necesitemos insertar símbolos.

Para ello HTML nos ofrece las entidades. Las entidades son unas estructuras que, mediante el uso de una codificación, nos permiten representar un símbolo.

La estructura de la entidad HTML es un ampersand(&) seguido del código o nombre de la entidad y terminado en un punto y coma.

```
<?php  
echo "varios &nbsp; &nbsp; &nbsp; espacios";  
?>
```



PÁGINAS WEB CON REFERENCIAS DE CARÁCTER Y CÓDIGOS UNICODE

<https://html.spec.whatwg.org/multipage/named-characters.html#named-character-references>

<https://unicode-table.com/es/sets/symbols-for-youtube/#check-marks>

Carácter	Entidad	Nota
&	&amp;	Interpretado como el comienzo de una entidad HTML.
<	&lt;	Interpretado como la apertura de una etiqueta
>	&gt;	Interpretado como el cierre de una etiqueta
"	&quot;	Interpretado como apertura o cierre del valor de un atributo
á	&aacute;	Vocal <i>a</i> acentuada
é	&eacute;	Vocal <i>e</i> acentuada
í	&iacute;	Vocal <i>i</i> acentuada
€	&euro;	Símbolo de la moneda euro
£	&pound;	Símbolo de la moneda libra
©	&copy;	Símbolo de copyright
®	&reg;	Símbolo de marca registrada
<i>espacio</i>	&nbsp;	Interpretado como un espacio en blanco, alternativa <a href="#">&amp;#160;</a>

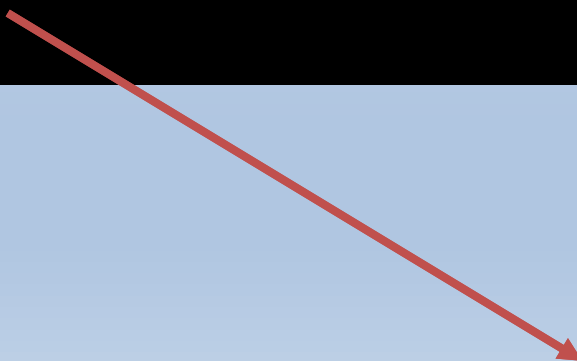
# Grandes bloques: heredoc

- Cuando tenemos necesidad de escribir largos bloques de código HTML, con variables intercaladas, podemos usar la construcción **heredoc** que nos permite escribir grandes cantidades de texto, sin necesidad de escapar caracteres en su interior.
- Además de la facilidad de lectura y pegado, esta sintaxis incrementa la velocidad de interpretación del código en el servidor si la comparamos con el uso de varios echo o print seguidos.

# heredoc

1. Al inicio del bloque de texto, debemos colocar tres veces el signo “menor que”, de esta manera: <<< seguido de varios caracteres alfanuméricos (en el ejemplo que sigue hemos elegido PRUEBA, pero puede ser cualquier otra combinación de letras).
2. Después pegamos el bloque de texto y código HTML que deseemos.
3. Para finalizar, repetimos los mismos tres caracteres que indicaron el inicio del bloque.

```
<?php
echo <<<PRUEBA
<p>Este texto puede tener dentro "comillas" sin necesidad de
escaparlas.</p>
<p>También procesa (reemplaza por su valor) las variables que
hubiera dentro del código</p>
<p>Esta construcción del lenguaje llamada heredoc es ideal para
incluir largos bloques de código HTML.</p>
PRUEBA;
?>
```



**Importante:** debe estar colocado al principio de la línea sin que exista ningún espacio

# heredoc

**Importante:** los caracteres indicadores del inicio y fin del bloque, deben incluirse al comienzo del renglón (sin dejar ni un solo espacio en blanco, ni tabulaciones, sangrado de código, ni comentarios), y tampoco deben contener un salto de línea.



Es posible almacenar la salida de heredoc en una variable

```
<?php
$contenido = <<<PRUEBA
<p>Este texto puede tener dentro "comillas" sin necesidad de
escaparlas.</p>
<p>También procesa (reemplaza por su valor) las variables que
hubiera dentro del código</p>
<p>Esta construcción del lenguaje llamada heredoc es ideal para
incluir largos bloques de código HTML.</p>
PRUEBA;
echo $contenido;
?>
```

# Grandes bloques: nowdoc

- El texto se procesa de igual forma que si fuera una cadena entre comillas simples
- No se realizará ninguna sustitución del valor de las variables.
- Sintaxis: el **identificador de apertura** entre **comillas simples**.

```
echo <<<'MICADENA'
```

```
...
```

```
MICADENA;
```

```
<?php
```

```
//nowdoc: devuelve el texto literalmente, sin interpretar las variables
```

```
$cadena = "Esto es una cadena de texto";
```

```
$contenido = <<<'PRUEBA'
```

```
<p>Este texto puede tener dentro "comillas" sin necesidad de escaparlas.</p>
```

```
<p>También no procesa (no reemplaza por su valor) las variables que hubiera  
dentro del código</p>
```

```
<p>Esta construcción del lenguaje llamada nowdoc es ideal para incluir  
largos bloques de código HTML.</p>
```

```
$cadena
```

```
PRUEBA;
```

```
echo $contenido;
```

```
?>
```

# El problema de las comillas en PHP

¿Cuándo usar comillas simples y cuándo comillas dobles?

- Los valores **alfanuméricos deben envolverse entre comillas**, ya sean simples o dobles;
- Los valores **numéricos no deben envolverse** entre comillas
- Ejemplo

```
$entero=5;  
$nombre="Alumno";
```

# El problema de las comillas en PHP

Para almacenar un dato alfanumérico dentro de una variable, deberemos comprender cómo afecta a esos datos el ser delimitado su inicio y su fin por los dos tipos de comillas existentes:

- **comillas dobles**
- **comillas simples**

El problema  
de las  
comillas en  
PHP

La interpretación de  
variables dentro de  
comillas

# comillas simples

1. Cuando deseamos que el intérprete de PHP escriba un **texto literalmente** usaremos **comillas simples** para delimitar el inicio y el final de ese texto.

## Funcionamiento:

- **no podremos incluir comillas simples dentro de ese texto** (deberemos escaparlas)
- cualquier variable que incluyamos dentro de comillas simples, **no será reemplazada por su valor.**

# comillas dobles

2. Cuando necesitemos que las variables se reemplacen por su valor, usaremos comillas dobles para delimitar el inicio y el final del bloque de texto.

## Funcionamiento:

- no podremos incluir comillas dobles dentro de ese texto (debemos escaparlas)
- cualquier variable que incluyamos dentro de comillas simples, **será reemplazada por su valor.**



```
<?php

$nombre = "Alumno";

$numero = 7;

$comillasDobles = "Texto entre comillas dobles, puede contener 'comillas simples' dentro sin problemas";

$comillasSimples = 'Texto entre comillas simples, puede contener "comillas dobles" dentro sin problemas';

$escapeDoble = "Texto con \"comillas\" dobles escapadas";

$espapeSencillo = 'Texto con \'comillas\' simples escapadas';

$variablesDobles = "Texto con variables como $nombre y $numero intercaladas entre comillas dobles, que se reemplazarán por su valor";

$variablesSimples = 'Texto con variables como $nombre y $numero intercaladas entre comillas simples, que no se reemplazarán por su valor, quedará escrito $nombre y $numero tal cual';

?>
```

# Formas de incluir un salto de línea en HTML

- Pero `\r\n` sólo generará un salto de línea para el código fuente, no para HTML.
- La combinación `\r\n` para el intérprete de PHP no se traduce en una etiqueta `<br>` y por lo tanto, el navegador no pondrá un salto de línea.

# Formas de incluir un salto de línea en HTML

- La forma de generar un salto de línea es utilizar `<br>` que es la etiqueta HTML.

- Ejemplos:

```
echo 'Hola<br>';  
echo "Hola<br>";  
echo "Hola\r\n";  
echo 'Hola\r\n';
```

# Formas de incluir un salto de línea en HTML

- El navegador mostrará:

Hola

Hola

Hola Hola\r\n

- El código del archivo html generado por el intérprete PHP es el siguiente:

Hola<br>Hola<br>Hola

Hola\r\n

## Diferencias entre echo y print

La principal diferencia entre echo y print:  
**echo** admite múltiples parámetros  
mientras que **print** solo admite uno

```
<?php
$a = "Hola mundo!";
echo $a, "<br>", $a, "<br>";
print $a;
print "<br>";
print $a;
?>
```

# Operadores

## Concatenación

.

## Concatenación y asignación

.=

PHP tiene dos operadores para trabajar con cadenas de texto:

- Con el operador de concatenación punto . puedes unir las dos cadenas de texto que le pases como operandos.
- El operador de asignación y concatenación .= concatena al argumento del lado izquierdo la cadena del lado derecho.

```
<?php
$a = "Módulo ";
$b = $a . "DWES"; // ahora $b contiene "Módulo DWES"
$a .= "DWES"; // ahora $a también contiene "Módulo DWES"
echo $a . "<br>";
echo $b . "<br>";
```

//Tanto echo como print admiten el operador de concatenación de cadenas (.)  
//para concatenar varias cadenas en una sola.

//Para imprimir el valor de una variable se puede usar echo o print.  
//Ambas funciones son muy similares, pero echo es ligeramente más rápido.  
//echo no devuelve ningún valor, mientras que print devuelve 1, por lo que  
puede ser utilizado en expresiones.

```
$a = "hola";
$b = "mundo";
```

```
echo "Probando el operador concatenación:<br> ";
echo $a . "<br>" . $b . "<br>";
print($a . "<br>" . $b . "<br>");
```

```
echo "Utilizando comillas dobles:<br>";
echo "$a<br>$b<br>";
print("$a<br>$b<br>");
?>
```

# Short tags

```
<!-- Probando short tags -->
<?php
$variable = "<br>Probando short tags"
?>
<?= $variable ?>
<?php echo $variable; ?>

<?= "<br>Soy una short tag" ?>

<?php echo "<br>NO soy una short tag" ?>
```



# short tags

`<?=$variable ?>`

Esto es idéntico a escribir:

`<?php echo $variable; ?>`

A lo largo del curso discutiremos las ventajas e inconvenientes de su utilización.

## Historial de cambios

Versión	Descripción
7.0.0	Se eliminaron de PHP las etiquetas de ASP <code>&lt;%, %&gt;</code> , <code>&lt;%=</code> , y la etiqueta de script <code>&lt;script language="php"&gt;</code> .
5.4.0	La etiqueta <code>&lt;?=</code> <b>siempre está disponible</b> independientemente del ajuste en el fichero <code>php.ini</code> de la directiva <code>short_open_tag</code> .

# Comentarios en PHP

En una página web los comentarios al **HTML** van entre los delimitadores:

```
<!--      comentario      -->
```

Dentro del código PHP, hay tres formas de poner comentarios:

1. **Comentarios de una línea** utilizando **//** Son comentarios al estilo del lenguaje C. Cuando una línea comienza por los símbolos **//**, toda ella se considera que contiene un comentario, hasta la siguiente línea.
2. **Comentarios de una línea** utilizando **#** Son similares a los anteriores, pero utilizando la sintaxis de los scripts de Linux.
3. **Comentarios de varias líneas.** También iguales a los del lenguaje C. Cuando en una línea aparezcan los caracteres **/\***, se considera que ahí comienza un comentario. El comentario puede extenderse varias líneas, y acabará cuando escribas la combinación de caracteres opuesta: **\*/**

# Comentarios en PHP

- Recuerda que cuando pongas comentarios al código PHP, éstos no figurarán en ningún caso en la página web que se envía al navegador (justo al contrario de lo que sucede con los comentarios a las etiquetas HTML).