Iniciación a PHP

Estructura de una página PHP

Una página PHP es un archivo de texto que contiene uno o varios fragmentos de código PHP y que también puede contener fragmentos de código HTML.

Fragmentos PHP

Los fragmentos de código PHP están delimitados por las etiquetas <?php (etiqueta inicial) y ?> (etiqueta final) y contienen las instrucciones de PHP. Más adelante se comentan otros posibles delimitadores de fragmentos de código PHP.

```
<?php
  echo "<p>Hola</p>";
?>
```

```
<p>Hola</p>
```

Los fragmentos de PHP no pueden anidarse, es decir, no puede haber un fragmento dentro de otro:

```
<?php
  echo "<p>Hola</p>";

<?php
  echo "<p>Adios</p>";
?>
```

```
Parse error: syntax error, unexpected '<' expecting end of file
in ejemplo.php on line 4
```

Si en una misma página hay varios fragmentos PHP, se tratan como un único programa. Los siguientes programas son equivalentes:

```
<?php
  $saludo = "Hola";          // Se define una variable
  echo "<p>$saludo</p>";    // Se escribe el valor de la variable
?>
```

```
<p>Hola</p>
```

```
<?php
  $saludo = "Hola";              // Se define una variable
?>
<?php
  echo "<p>$saludo<p>";    // Se escribe el valor de la variable
?>
```

```
<p>Hola</p>
```

Si el programa termina con un fragmento PHP, ese último fragmento PHP no necesita cerrarse. Los siguientes programas son correctos:

```
<?php
  echo "<p>Hola</p>";
```

```
<p>Hola</p>
```

```
<?php
  echo "<p>Hola</p>";
?>
<?php
  echo "<p>Adios</p>";
```

```
<p>Hola</p>
<p>Adios</p>
```

Fragmentos PHP y fragmentos HTML

En una página PHP todo lo que no son fragmentos PHP son fragmentos HTML. Los fragmentos de código HTML no están delimitados por ninguna etiqueta.

```
<?php
  echo "<p>Hola</p>";
?>
<p>¿Cómo estás?</p>
```

```
<p>Hola</p>
<p>¿Cómo estás?</p>
```

Cuando un navegador solicita una página PHP a un servidor, el servidor lee el archivo secuencialmente y va generando el resultado:

- Si la página contiene fragmentos HTML, el código HTML se incluye sin modificaciones en el resultado.

- Si la página contiene fragmentos PHP, se ejecutan las instrucciones que se encuentran en los fragmentos de código PHP. Si esas instrucciones generan texto, el texto se incluye en el resultado.

Cuando el servidor termina de leer el archivo, el servidor envía al navegador el resultado.
Es importante señalar que:

- El navegador recibe una página web (es decir, únicamente código HTML), no recibe código PHP.
- Los fragmentos PHP tienen que generar las etiquetas HTML que se necesiten para una correcta visualización de la página web en el navegador.
- El navegador no puede distinguir en la página recibida, qué parte ha sido generada en un fragmento PHP y qué parte se encontraba ya en un fragmento HTML.
- Como la página PHP se lee secuencialmente, el código HTML generado por los fragmentos PHP y el incluido en los fragmentos HTML se encuentran en el mismo orden en que se encontraban los fragmentos en la página PHP.

Los ejemplos siguientes muestran diferentes programas PHP que generan el mismo resultado:

- Programa con un único fragmento PHP

```
<?php
  echo "<p>Hola</p>";
  echo "<p>¿Cómo estás?</p>";
  echo "<p>Adios</p>";
?>
```

```
<p>Hola</p>
<p>¿Cómo estás?</p>
<p>Adios</p>
```

- Programa con un fragmento PHP y un fragmento HTML

```
<p>Hola</p>
<?php
  echo "<p>¿Cómo estás?</p>";
  echo "<p>Adios</p>";
?>
```

```
<p>Hola</p>
<p>¿Cómo estás?</p>
<p>Adios</p>
```

- Programa con dos fragmentos PHP y un fragmento HTML

```
<?php
  echo "<p>Hola</p>";
?>
<p>¿Cómo estás?</p>
```

```
<?php
  echo "<p>Adios</p>";
?>
```

```
<p>Hola</p>
<p>¿Cómo estás?</p>
<p>Adios</p>
```

En un fragmento PHP no pueden escribirse etiquetas HTML sueltas; el código HTML debe generarse siempre con instrucciones de PHP. El programa siguiente no puede ni siquiera ejecutarse y produce un error de sintaxis:

```
<?php
  <p>Hola</p>
?>
```

```
Parse error: syntax error, unexpected '<' expecting end of file
in ejemplo.php on line 2
```

De la misma manera, no se deben escribir instrucciones de PHP fuera de fragmentos PHP. No se producen errores, pero el resultado no es el esperado. Como el texto fuera de fragmentos PHP se envía tal cual al navegador, el navegador recibirá las instrucciones de PHP, que el navegador no sabe ejecutar y se mostrarán en la pantalla.

```
<?php
  echo "<p>Hola</p>";
?>
<p>¿Cómo estás?</p>
echo "<p>Adios</p>";
```

```
<p>Hola</p>
<p>¿Cómo estás?</p>
echo "<p>Adios</p>";
```

El delimitador <? ... ?>

Nota: El delimitador **<?** ... **?>** no se debe utilizar. Se comenta porque todavía se puede encontrar en código escrito hace muchos años.

El delimitador **<?** se podía utilizar antiguamente como delimitador de bloques PHP, pero desde hace años **se desaconseja completamente** su uso porque provoca conflictos, entre otros, con la instrucción de procesamiento xml (<?xml) de los documentos XML. En su lugar, se debe utilizar el delimitador **<?php .. ?>**.

Mediante la directiva short_open_tag (etiqueta de apertura abreviada) se puede permitir el uso del delimitador **<?** además del delimitador **<?php**. Si la directiva short_open_tag tiene el valor On (lo que está desaconsejado), el siguiente programa no daría errores:

```
<?
  echo "<p>Hola</p>";
?>
```

```
<p>Hola</p>
```

Pero si la directiva short_open_tag tiene el valor Off (valor recomendado), el mismo programa produciría un resultado incorrecto:

```
<?
  echo "<p>Hola</p>";
?>
```

```
<?
 echo "<p>Hola</p>";
?>
```

Este resultado se debe a que como el programa no contiene el delimitador <?php, su contenido no se reconoce como fragmento PHP. Como no se reconoce como fragmento PHP, el servidor lo envía tal cual al navegador, que no puede mostrarlo correctamente puesto que no es html correcto.

El delimitador <?= ... ?>

El delimitador **<?=** es una abreviatura de la expresión **<?php echo** que se utiliza en muchos frameworks de PHP en los documentos que sirven de plantilla de generación de interfaces, sobre todo cuando estas plantillas las van a editar usuarios que pueden no tener conocimientos de programación (diseñadores gráficos, etc.).

```
<?php
  $saludo = "Hola";
  $despedida = "Adios";
?>
<p><?= $saludo ?></p>
<p><?= $despedida ?></p>
```

```
<p>Hola</p>
<p>Adios</p>
```

Antes de PHP 5.4, publicado en marzo de 2012, para poder usar este delimitador se necesitaba activar la directiva short_open_tag. El problema era que, como se comenta en el apartado anterior, desde hace años se aconseja no activar esa directiva para impedir el uso del delimitador <?. A partir de PHP 5.4 se resolvió este problema desvinculado el delimitador <?= de la directiva short_open_tag. por tanto, desde PHP 5.4, el delimitador <?= se puede utilizar sin limitaciones.

Nota: PHP requiere que las instrucciones terminen en punto y coma al final de cada sentencia. La etiqueta de cierre de un bloque de código de PHP automáticamente implica un punto y coma, por tanto no es necesario usar un punto y coma para cerrar la última línea de un bloque de PHP.

Diferencias entre echo y print

Ambas funciones nos permiten mostrar cadenas de textos y ambas funciones no llevan paréntesis al momento de llamarlas. Esta es la forma y tipo de cada uno:

```
int print ( string $arg )
void echo ( string $arg1 [, string $... ] )
```

De la definición anterior podemos deducir que:

- print imprime una cadena, echo puede imprimir más de una separadas por coma:

```
<?php
  print "<p>Hola</p>";
  echo "<p>Hola</p>", "<p>Hola de nuevo</p>";
?>
```

```
<p>Hola</p>
<p>Hola</p>
<p>Hola de nuevo</p>
```

- print devuelve un valor int que según la documentación siempre es 1, por lo que puede ser utilizado en expresiones mientras que echo es tipo void, no hay valor devuelto y no puede ser utilizado en expresiones:

```
<?php
  // Se imprime "<p>Hola</p>" y la variable $cadena toma el valor
1
  $cadena = print "<p>Hola</p>";
  // La siguiente expresión da error
  $cadena = echo "<p>Hola</p>";
?>
```

```
Parse error: syntax error, unexpected 'echo' (T_ECHO) in ejemplo.php on line 5
```

print_r vs var_dump

Estas dos funciones **imprimen los detalles de una variable**, incluyendo su valor, **en un formato legible por el humano**. Si es un array o un objeto también imprimen los detalles de cada elemento. Se utilizan frecuentemente durante la depuración de código, situación en la que **var_dump** suele ser más útil por la mayor información que proporciona. Las diferencias:

- Si el valor de la variable es una cadena de texto, **var_dump** imprime la cadena entre dobles comillas, **print_r** no.
- **print_r** no imprime nada visible para **false** y cadenas vacías.
- **var_dump** proporciona información sobre el tamaño y tipo de datos de la variable y, en el caso de arrays y objetos, de los elementos que la componen. **print_r** no da información sobre el tamaño de la variable ni sobre el tipo de datos.

Ejemplos:

```php
<?php
  $valores = [5, 0.0, "Hola", false, ""];
  var_dump($valores);
?>
```

```
array(5) {
    [0]=> int(5)
    [1]=> float(0)
    [2]=> string(4) "Hola"
    [3]=> bool(false)
    [4]=> string(0) ""
}
```

```php
<?php
  $valores = [5, 0.0, "Hola", false, ""];
  print_r($valores);
?>
```

```
array(5) {
    [0]=> 5
    [1]=> 0
    [2]=> Hola
    [3]=>
    [4]=>
```

```
}
```

- **print_r** puede devolver el resultado en lugar de imprimirlo si se proporciona el segundo parámetro como **true**:

```
<?php
  $valores = [5, 0.0, "Hola", false, ""];
  $salida = print_r($valores, true);
?>
```

Comentarios en páginas PHP

En un fragmento PHP se pueden comentar líneas de código utilizando:

- // para comentar el resto de la línea (como en C++)
- # para comentar el resto de la línea (como en la shell de Unix o en Perl)
- /* */ para delimitar varias líneas (como en C)

Estos comentarios no se añaden al código HTML generado por la página, por lo que no pueden verse en el navegador.

```
<p><strong>
<?php
  // La instrucción print escribe texto en la página web
  echo "Hola"; // El comentario se puede escribir al final de la
línea
?>
</strong></p>
```

```
<p><strong>Hola</strong></p>
```

```
<p><strong>
<?php
  # La instrucción print escribe texto en la página web
   echo "Hola"; # El comentario se puede escribir al final de la
línea
?>
</strong></p>
```

```
<p><strong>Hola</strong></p>
```

```
<?php
  echo "<p><strong>";
   /* Dentro de un fragmento PHP no se pueden escribir etiquetas
html       sueltas,   tienen   que   estar   siempre   incluidas   en
instrucciones print
  */
?>
Hola
<?php
  echo "</strong></p>";
?>
```

```
<p><strong>Hola</strong></p>
```

Si se quieren escribir comentarios en los fragmentos HTML, hay que utilizar la etiqueta de comentarios de HTML <!-- .... -->.

Estos comentarios, como todo el código HTML situado en los fragmentos HTML, se incluyen sin modificaciones en el resultado, por lo que pueden verse en el navegador.

```
<p>
<?php
  print "Hola";
?>
</p>
<!-- El texto anterior ha sido generado por PHP -->
```

```
<p>Hola</p>
<!-- El texto anterior ha sido generado por PHP -->
```