UT2

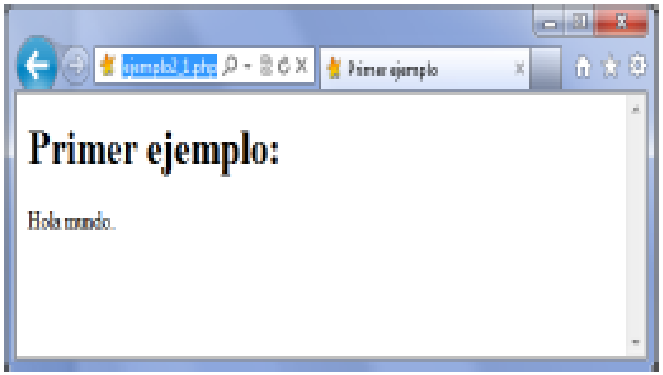
INSERCIÓN DE CÓDIGO EN PÁGINAS WEB

1. Etiquetas para la inserción de código

- Cada lenguaje que se puede utilizar para insertar código dentro de una página web utiliza una serie de **etiquetas para delimitar los fragmentos de código** que han de ser procesados por el servidor web.

- El componente del servidor encargado de procesar el código ignorará el código HTML que se encuentra fuera de dichas etiquetas.

```
<html>
  <head>
    <title>Primer ejemplo</title>
  </head>
  <body>
    <h1>Primer ejemplo: <br/></h1>
    <?php
      echo "Hola mundo.";
    ?>
  </body>
</html>
```



1. Etiquetas para la inserción de código

- Características de los lenguajes de etiquetado de código embebido
  - Todo script comienza y termina con una etiqueta de inicio y otra de fin.
  - Podemos configurar otros estilos de etiquetas en estos lenguajes.
  - Los espacios en blanco que escribamos dentro del código embebido no tienen ningún efecto.
  - El código de servidor embebido en páginas HTML está formado por un conjunto de sentencias que deben estar claramente separadas.
  - Los scripts embebidos pueden situarse en cualquier parte del recurso web ejecutado.
  - El número de scripts que podemos tener dentro de un fichero HTML es indefinido.
  - Cuando se ejecuta un código embebido, el script entero se sustituye por el resultado de dicha ejecución, incluidas las etiquetas de inicio y fin.

1. Etiquetas para la inserción de código

- **Comentarios**
  - Forma de mejorar la legibilidad del código y que se recomienda siempre como una buena práctica a la hora de programar.

- **Comentarios de una línea**
  - ***//*** comentario de una línea
  - # otro comentario de una línea

- **Comentarios de múltiples líneas**
  - /* comentario

  en más de una

  línea */

1. Etiquetas para la inserción de código

- **Inclusión de código páginas HTML**
  - La instrucción **echo** sirve tanto para cadenas de caracteres como para imprimir variables.
    - echo "ejemplo de impresión de cadena de caracteres";
  - El resultado de usar **print** sería el mismo:
    - print "ejemplo de impresión de cadena de caracteres“;
  - La diferencia entre *echo* y *print* es que la sentencia *echo* puede imprimir más de un argumento.
    - echo “imprimir primer argumento”, “ y el segundo”;
  - También se pueden imprimir números directamente y los valores de las variables:
    - echo 217;
    - echo $var; print $var;

1. Etiquetas para la inserción de código

Inclusión de código páginas HTML

- Tanto las comillas simples (‘) como dobles (“) se pueden usar para crear cadenas de caracteres.
  - echo “un texto";
  - echo ‘otro texto’;

- La impresión de caracteres reservados se hace indicándole al intérprete de PHP que el carácter a continuación de la barra invertida (\) es un carácter que debe imprimirse sin interpretarlo. **Aquí el carácter \ actúa como escape**.
  - echo "Este texto contiene una \" (una comilla doble)";
  - echo 'Este texto contiene una \' (una comilla simple)';
  - echo "Este texto contiene una barra invertida: \\";
  - echo “ y este un símbolo del dólar: \$”;

2. Variables

Definición

- "Almacenes temporales de datos que permiten gestionar los datos utilizados por la aplicación web durante el flujo de ejecución de una página determinada".
- Se identifican por el símbolo del dólar ($) seguido del nombre de una variable.
- En PHP, las variables no necesitan ser declaradas explícitamente.
- No tienen un tipo definido hasta que no se les asigna un valor
  - $var = 7; // Ahora $var es un entero
  - $var = "ana"; // Ahora es una cadena de caracteres.

2. Variables

Reglas de nombrado

- El nombre debe comenzar con una letra o con un guión bajo (“_”).
- El nombre únicamente puede contener caracteres alfanuméricos y guiones bajos (a-z, A-Z, 0-9 y _).
- El nombre de una variable no debe contener espacios en blanco.
- PHP es **case-sensitive**
  - $Variable, $variable, $Variable y $VARIABLE son variables completamente diferentes.

2. Variables

- **Tipos de datos**
  - Tipos escalares: boolean, integer, float, string.
  - Tipos compuestos: array, object.
  - Tipos especiales: NULL , resource.
  - Pseudo-tipos: mixed, number, callback.

2. Variables

Operaciones básicas

- Operaciones básicas:
  - $var = (($var - 3) * 4) / 2;

- Concatenación (se utiliza el punto .):
  - $var = "cadena" . " unida";

- Varias formas de añadir 1 a la variable $var:
  - $var = $var + 1;
  - $var += 1;
  - $var++;

- Varias formas de multiplicar o dividir:
  - $var = $var * 2; → $var *= 2;
  - $var = $var / 2; → $var /= 2;

2. Variables

Conversiones entre tipos de datos

- Funciones
  - string strval(mixedvariable) → transforma a string.
  - integer intval(mixedvariable) → transforma a integer.
  - float floatval(mixedvariable) → transforma a float.

- Genérica

  - settype(mixed variable, string type).
  - Parámetro “variable”: valores de tipo array, boolean, float, integer, object o string.
  - Parámetro “type”: cadena de caracteres que indica el tipo al que queremos transformar el parámetro “variable”.

2. Variables

- **Conversiones entre tipos de datos**

Sentencia	Resultado
(int) $var (integer) $var	Conversión a tipo integer.
(bool) $var (boolean) $var	Conversión a tipo boolean.
(float) $var (double) $var (real) $var	Conversión a tipo float.
(string) $var	Conversión a tipo string.
(array) $var	Conversión a tipo array.
(object) $var	Conversión a tipo object.

2. Variables

- **Conversiones entre tipos de datos. Ejemplos**

Valor de $var	(int) $var	(bool) $var	(string) $var	(float) $var
null	0	false	""	0
true	1	true	"1"	1
false	0	false	""	0
0	0	false	"0"	0
3.8	3	true	"3.8"	3.8
"0"	0	false	"0"	0
"10"	10	true	"10"	10
"6 metros"	6	true	"6 metros"	6
"hola"	0	true	"hola"	0

2. Variables

Conversiones entre tipos de datos

Conversión automática

- Se produce cuando combinamos en una misma expresión dos variables que inicialmente tienen tipos diferentes o cuando pasamos una variable como argumento a una función que espera un tipo de dato diferente.

- $var se convierte a tipo integer con valor 35:
  - $var = "20" + 15;

- $var se convierte a string con valor = "20 años“:
  - $var= 20 . " años";

- $var se convierte a tipo integer con valor = 20:
  - $var = 20 + " años";

- $var se convierte a tipo integer con valor = 42:
  - $var= 40 + "2 razones";

2. Variables

- **Comprobación del tipo de una variable**
  - boolean is_int(mixed variable).
  - boolean is_float(mixed variable).
  - boolean is_bool(mixed variable).
  - boolean is_string(mixed variable).
  - boolean is_array(mixed variable).
  - boolean is_object(mixed variable).

2. Variables

Estado de una variable

- Una variable puede estar en un estado indeterminado (no tener un valor asignado) e incluso puede no haber sido definida (estado indefinido).

- Funciones para comprobar el estado de una variable:
  - **boolean isset(mixed var)** → comprueba si a una variable se le ha asignado un valor no nulo.
  - **boolean empty(mixed var)** → comprueba si esa variable tiene un valor.

- Función para pasar una variable a estado "indefinido":
  - unset().

2. Variables

- **Estado de una variable**

Contenido de $var	isset($var)	empty($var)	(bool) $var
$var = null;	false	true	false
$var = 0;	true	true	false
$var = true	true	false	true
$var = false	true	true	false
$var = "0";	true	true	false
$var = "";	true	true	false
$var = "foo";	true	false	true
$var = array( );	true	true	false
unset ($var);	false	true	false

2. Variables

Ámbito de una variable

- "Contexto dentro del que la variable está definida, es decir, la zona del programa en la que puede ser accedida".

- **Ámbito local**: las variables internas a una función única *exclusivamente* pueden ser utilizadas *dentro* de dicha función.

```
function duplicar($var){
      $temp= $var  * 2;
}
$variable = 5;
duplicar($variable);
echo "El valor de la variable \$tempes: $temp";
```

→Salida: El valor de la variable $tempes: … y ningún valor para $temp.

2. Variables

Ámbito de una variable

- **Ámbito global**: una variable dentro de una función es la misma que la variable que hemos utilizado (o utilizaremos) *fuera* de esa función

```
$temp = 0;
function  duplicar($var){
      $temp= $var  * 2;
}
$variable = 5;
duplicar($variable);
echo "El valor de la variable \$tempes: $temp";
```

- →Salida: El valor de la variable $tempes: 0.

- La utilización de variables globales sin control puede resultar en un código difícil de mantener y propenso a dar errores.