

DESARROLLO WEB EN ENTORNO SERVIDOR
TÉCNICO EN DESARROLLO DE APLICACIONES WEB

Inserción de código en páginas web

ÍNDICE

/ 1. Introducción y contextualización práctica	3
/ 2. Lenguajes y tecnologías	4
2.1. Contenedores de servlets	5
/ 3. Obtención del lenguaje de marcas	5
/ 4. Etiquetas para la inserción de código	6
/ 5. Lenguaje de programación PHP	7
5.1. Configuración de PHP. Directivas	8
/ 6. Variables en PHP	9
6.1. Ámbito de una variable y constantes	10
6.2. Tipos de datos	11
6.3. Conversión tipos de datos	12
6.4. Cadenas de texto	13
/ 7. Caso práctico 1: “Crear e imprimir variables y tipos de datos”	14
/ 8. Caso práctico 2: “Crear e imprimir variables y tipos de datos con especificaciones”	15
/ 9. Operadores	16
/ 10. Resumen y resolución del caso práctico de la unidad	18
/ 11. Bibliografía	19
/ 12. Webgrafía	19

OBJETIVOS

Reconocer los mecanismos de generación de páginas web a partir de lenguaje de marcas con código embebido.

Identificar las principales tecnologías asociadas.

Utilizar etiquetas para la inclusión de lenguajes de marcas.

Reconocer la sintaxis del lenguaje de programación a utilizar.

Escribir sentencias simples y comprobar su funcionamiento.

Utilizar variables y operadores.

Identificar ámbitos de utilización de variables.

/ 1. Introducción y contextualización práctica

En la actualidad, existen diferentes lenguajes y tecnologías aplicadas al desarrollo web en entorno servidor, útiles para modelar páginas y aplicaciones web dinámicas. En este tema, veremos el funcionamiento general de JSP o ASP, centrándonos en el lenguaje de programación PHP, uno de los más extendidos para el desarrollo de páginas web con contenido dinámico.

En esta unidad se sentarán, por tanto, las bases del lenguaje de programación PHP. Para trabajar con este lenguaje, necesitamos situarlo en un servidor. En el tema 1 trabajamos en la instalación y puesta en funcionamiento de este (XAMPP), que nos permitirá el desarrollo con PHP.

PHP, al igual que el resto de lenguajes, tiene su propia sintaxis y reglas de programación. Se prestará especial interés, en este primer capítulo sobre PHP, a la creación de variables, tipos de datos, etiquetas de inserción u operadores, que serán importantes para continuar con el aprendizaje de este lenguaje en temas posteriores.

Escucha el siguiente audio donde planteamos la contextualización práctica de este tema. Encontrarás su resolución en el apartado «Resumen y resolución del caso práctico de la unidad».



Fig. 1. Primer programa con PHP.



Audio intro. "Elementos clave en PHP"

<https://bit.ly/33HFw7z>





/ 2. Lenguajes y tecnologías

El desarrollo en entorno servidor utiliza distintas tecnologías y lenguajes que pueden ser interpretados por un navegador HTML, ya que el objetivo principal de los servidores web es ofrecer respuesta a las peticiones derivadas del lado del cliente.

Los lenguajes y tecnologías clave para el desarrollo en entorno servidor son múltiples: PHP, JSP y Servlet o ASP, entre otras.

- **PHP:** PHP (Hypertext Processor). Lo soportan casi todos los servidores web y es de código abierto. En la actualidad, es uno de los lenguajes de programación en entorno servidor más utilizados, gracias a su potencia y simplicidad. PHP permite embeber fragmentos de código desarrollado en este lenguaje, en una página web HTML. Se trata de una tecnología del lado del servidor, también conocido como backend. Una de las características más importantes es que permite utilizar tanto programación estructurada como orientada a objetos.
- **ASP:** ASP (Active Server Pages). Se trata de la tecnología para el desarrollo de entorno servidor de Microsoft para modelar los sistemas de páginas dinámicas. Se basa en los lenguajes de programación Jscript (JavaScript de Microsoft) o en Visual Basic Script.
- **JSP:** JSP (Java Server Pages). En el caso de servidores de aplicaciones empresariales, se utiliza Java como lenguaje de programación. Gracias a JSP, podremos desarrollar aplicaciones web que se ejecuten en varios servidores. Para el desarrollo de estas páginas, se utiliza HTML/XML, junto con etiquetas que permiten incluir sintaxis en Java.

El funcionamiento de este tipo de tecnología se basa en la petición de una página JSP desde el navegador. Esta petición será tratada por el servlet correspondiente, que se compilará y ejecutará, y, en consecuencia, construirá la página solicitada.

Como se puede ver en el ejemplo de código siguiente, se construye una página HTML, que presentará partes estáticas y dinámicas. El código Java embebido dentro de las etiquetas `<% %>` modela la parte dinámica.

```
<html>
  <head>
    <title>Probando JSP</title>
  </head>
  <body>
    <h3> **** Fecha actual ***
      <%= new java.util.Date() %>
    </h3>
  </body>
</html>
```

Código 1. Programa básico con JSP.



2.1. Contenedores de servlets

En Java encontramos un tipo de objeto utilizado para el desarrollo en entorno servidor, los Servlet, que se encargan de modelar y dar respuestas a las peticiones entre cliente y servidor.

Para el tratamiento de estos objetos de forma dinámica, necesitamos **Contenedores de Servlets**, los cuales reciben las peticiones de páginas web y las redireccionan hacia el objeto Servlet que se encarga de su tratamiento. Los contenedores pueden soportar protocolos HTTPS y HTTP desde su versión HTTP/1.0.

Podríamos decir que la función de estos elementos es distribuir las peticiones hacia el objeto Servlet adecuado para su tratamiento.

Uno de los contenedores de servlets más utilizados en la actualidad es Apache Tomcat.

El proceso íntegro para el envío y recepción de la respuesta de una petición se basa, de forma general, en el siguiente diagrama de flujo:

- En primer lugar, el usuario, desde un navegador web, realiza una petición de página o servicio a un servidor web a través de HTTP. En este servidor se encuentran los contenedores Servlets. Este envío se realiza a través de una petición `HttpServletRequest`.
- Tras la recepción de la petición, el contenedor envía la petición al Servlet.
- A continuación, desde la clase en Java que modela el Servlet, se construye la página que da respuesta a la petición inicial y esta se envía de nuevo al contenedor.
- Finalmente, se envía al cliente, a través del navegador, la página web solicitada. Para modelar la respuesta, se utiliza un elemento `HttpServletResponse`.

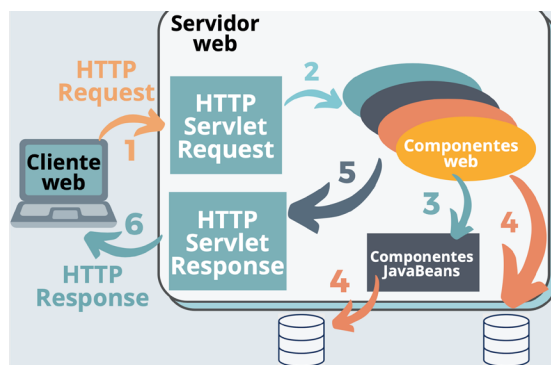


Fig. 2. Diagrama de flujo para Servlets.

/ 3. Obtención del lenguaje de marcas

Desde el entorno cliente, normalmente desde un navegador web, se realiza una petición de una página o aplicación web, cuya base estructural está programada con lenguaje de marcas HTML. Esta petición de página se solicita a un servidor web.

En el caso de las páginas estáticas, todo podría estar íntegramente desarrollado sobre HTML. Ahora bien, la inclusión de más páginas y aplicaciones web dinámicas requiere del uso de ciertas tecnologías que permiten actualizar y adaptar el contenido de un sitio en función del usuario o de ciertos parámetros de entrada.

Por lo tanto, cierto fragmento de código o fichero externo, habitualmente embebido en una página HTML, se encargará de modelar una petición concreta y de extraer los datos de la respuesta recuperada. Uno de los lenguajes más utilizados para la creación de páginas web dinámicas es PHP, un lenguaje registrado bajo una licencia Open Source que permite su descarga y uso de forma completamente gratuita.

El desarrollo e interpretación de una página cualquiera, implementada en PHP, requiere de un servidor web. Este elemento es clave, de lo contrario, no se podrá programar con PHP.

Tras realizar el procesamiento oportuno en PHP, se modela y muestra el resultado final en lenguaje HTML, interpretable por el navegador que recibe la respuesta del servidor en el lado del cliente.

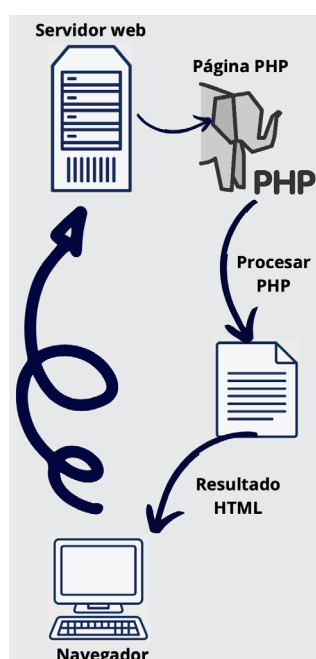


Fig. 3. Diagrama de flujo Cliente-Servidor PHP.

/ 4. Etiquetas para la inserción de código

Los lenguajes de marcas permiten desarrollar documentos a través de múltiples etiquetas (también conocidas como marcas) que determinan la estructura y apariencia de un fichero HTML y del contenido que recogen. Es posible diferenciar entre 3 tipos de marcado:

- **Marcado de presentación:** En este tipo, las etiquetas aparecen ocultas al usuario, es decir, en las herramientas que utilizan este tipo de marcado el usuario final solo verá el resultado. Por ejemplo, Microsoft Word o Pages, entre otros, utilizan un marcado de presentación. En este caso, las etiquetas no son visibles para el usuario final del documento, pero tampoco lo son para el desarrollador de dicho fichero.
- **Marcado de procedimiento:** En este caso, a través de un conjunto de etiquetas, se da formato al documento. Entonces, las etiquetas sí serán visibles para el desarrollador, pero no para el usuario final. HTML es un claro ejemplo de este tipo de marcado.
- **Marcado descriptivo:** Esta última marcación realiza la descripción de fragmentos de código gracias al uso de diferentes marcas. Hablamos de lenguajes como XML.



Para la inserción de código PHP en una página web, deben entenderse dos puntos clave:

- Qué etiquetas son necesarias para embeber el código en una página HTML para que su sintaxis sea reconocida.
- Cómo almacenar un fichero que implementa este lenguaje.

Para poder embeber un fragmento de código o fichero PHP en un documento HTML, se utilizan las etiquetas de cierre y apertura: '`<?php`' y '`?>`'

```
<?php
//Funciones y código en PHP
?>
```

Código 2. Inserción etiquetas base PHP.

Para la inserción de código JSP, serán necesarias nuevas etiquetas de apertura y de cierre, en este caso: `<? y ?>`.

En el caso de ASP, se reemplaza el símbolo de interrogación por el de porcentaje, `<% %>`. Además, es posible realizar esta inserción utilizando las etiquetas `<script>` de la siguiente forma:

```
<script language="VBScript" runat="Server">
...
</script>
```

Código 3. Inserción base ASP.

El atributo `runat` permite especificar el entorno exacto de ejecución del fragmento de código. Concretamente, se indica `Server`, puesto que se está haciendo referencia al entorno servidor.

/ 5. Lenguaje de programación PHP

Todo archivo que implemente lenguaje en PHP tiene que guardarse utilizando la extensión `.php` seguido del nombre del fichero. Este tipo de archivos pueden contener, además de las instrucciones codificadas con PHP, otro tipo de código implementado con los lenguajes propios del desarrollo web, como HTML, JavaScript o CSS.

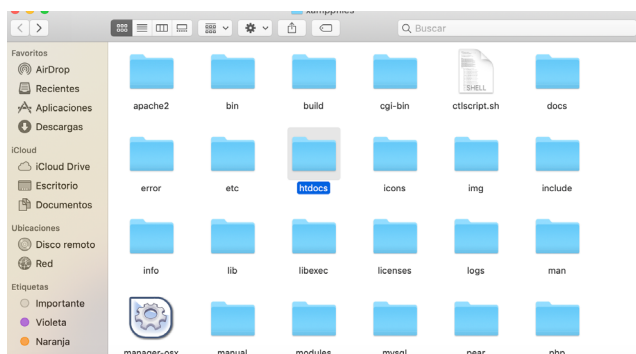


Fig. 4. Ficheros y estructura de carpetas.



A lo largo de este tema y en los sucesivos, veremos funciones comunes utilizadas en el desarrollo con PHP que se deben conocer. Dos de las más habituales son echo y print, puesto que nos permiten imprimir y mostrar por pantalla. Por lo tanto, son de gran utilidad para probar el funcionamiento de nuestra aplicación.

Como ya hemos adelantado, para poder embeber un fragmento de código o fichero PHP en un documento HTML, se utilizan las etiquetas de cierre y apertura `<?php` y `?>`. Todo aquel código que se escriba fuera de estas etiquetas, será ignorado por PHP, pero HTML sí tratará de interpretarlo.

```
<html>
<head>
  <title>Prueba de PHP</title>
</head>
<body>
  <?php
    echo "Hola Mundo!!!";
  ?>
</body>
</html>
```

Código 4. HolaMundo PHP + HTML.

En PHP, a diferencia de HTML, es necesario separar cada instrucción incorporando al final de la línea el símbolo ; (punto y coma). Puede omitirse en la última línea de un bloque de código PHP. En el código anterior, se podría omitir, aunque es aconsejable utilizarlo, para evitar errores si luego se incorporan nuevas líneas.

5.1. Configuración de PHP. Directivas

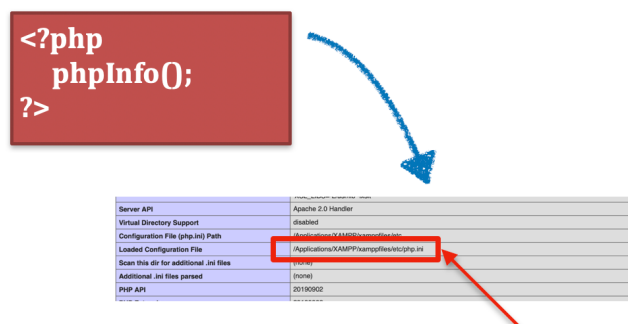
La configuración de PHP en el servidor se realiza a través del fichero `php.ini`, en el cual se recogen múltiples directivas, entre las que destacan algunas de las más utilizadas.

Nombre	Descripción
phpinfo	Devuelve información general sobre PHP. Por ejemplo, la ubicación exacta del fichero de configuración <code>php.ini</code>
file_uploads	Esta función determina si se pueden subir ficheros al servidor por HTTP
upload_max_filesize	Si se pueden subir ficheros, esta función indica el tamaño máximo de subida por fichero
short_open_tags	Aunque no es aconsejable utilizar este tipo de indicadores, esta función indica si se puede o no utilizar la forma abreviada para las etiquetas <code><?php</code> y <code>?></code> , que sería <code><? ?></code>
error_reporting	Muestra qué tipo de errores se indican en el caso de que ocurran
max_execution_time	Ajusta el número de segundos máximos para la ejecución de un script en PHP.

Tabla 1. Directivas PHP.



Por ejemplo, para obtener toda la información relativa a PHP sobre la versión que estamos utilizando o la ubicación exacta de los archivos, se implementa el siguiente fragmento de código, que tiene como resultado la tabla que se muestra más abajo.



```
<?php
phpInfo();
?>
```

→

Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/Applications/XAMPP/xamppfiles/etc
Loaded Configuration File	/Applications/XAMPP/xamppfiles/etc/php.ini
Scan this dir for additional ini files	(none)
Additional ini files parsed	
PHP API	20150902

Fig. 5. "Contenido ejemplo directiva PHPInfo"



Audio 1. "Configuración de PHP directivas"

<https://bit.ly/2PH6gwl>



/ 6. Variables en PHP

Las variables en PHP, al igual que en el resto de lenguajes de programación, son un elemento muy importante para el desarrollo de cualquier programa. Una variable está formada por un espacio de almacenamiento (en la memoria principal de un equipo), que estará identificada por una dirección concreta. Ahora bien, se utiliza un nombre simbólico, es decir, un identificador único para dicha variable.

Para la creación de variables en PHP, se coloca antes del nombre el símbolo \$. Además, hay que tener en cuenta que, en este caso, es sensible a minúsculas y mayúsculas. A continuación, se recogen las directrices para la creación del nombre de una variable:

- El nombre de la variable va precedido por el **símbolo \$**:

```
$var='hola;' // Correcto
var=hola; // Incorrecto
```

Código 5. Símbolo \$ variables PHP.

- Es sensible a minúsculas y mayúsculas. En el siguiente caso, se crearían dos variables diferentes:

```
$Var='hola;' // Correcto
$var='hola2;' // Correcto
```

Código 6. Minúsculas y Mayúsculas en PHP.



- El primer valor debe ser el carácter de subrayado o una letra.

```
$Var='hola'; // Correcto
$_var='hola2'; // Correcto
$2var='hola3' // Incorrecto
```

Código 7. Valores válidos variables PHP.

- Es posible utilizar cualquier letra a-z y A-Z y cualquier dígito del 0 al 9.

```
$999='hola'; // Incorrecto
$v999V='hola2'; // Correcto
```

Código 8. Valores válidos variables PHP.

La asignación de un valor a una variable puede realizarse por valor o por referencia. La asignación por valor se utiliza en los ejemplos anteriores, en los que se define una variable precedida de \$ y, a continuación, se le asigna un valor.

En el caso de hacerlo por referencia, se le indica que tome el contenido de otra variable ya existente. En este caso, se precede por el símbolo &. Para asignar una variable por referencia, es necesario que esta se haya creado previamente. Por ejemplo, en el caso de tener \$variable1=&\$variable2, la asignación sería correcta solo si variable2 se ha definido previamente.

6.1. Ámbito de una variable y constantes

El ámbito de una variable se refiere al contexto sobre el que está definida, es decir, permite establecer las partes del programa sobre las que puede ser usada. De esta forma, una variable global será accesible desde todo el programa, mientras que una de tipo local restringe su uso a la función donde ha sido definida.

Tipo	Definición	Ejemplo
global	Las variables globales son accesibles desde todo el programa	global \$var=0;
local	Las variables locales son accesibles desde la función donde han sido definidas. Pierden su valor cuando el programa abandona la ejecución de la función	\$var=0.1
static	Las variables estáticas solo existen dentro de la función donde han sido definidas, pero mantienen su valor cuando se deja de ejecutar la función donde toman el valor	static \$var=0;

Tabla 2. Ámbitos de una variable.

Las variables con nombre dinámico permiten asignar como nombre de variable el contenido de otras variables. De esta forma, modificando el contenido de la segunda de estas variables, se podría cambiar de forma dinámica el nombre de las variables definidas en un programa. Para implementar este tipo de asignación, se utiliza el símbolo de dólar doble, es decir, \$\$.



Creación de constantes

Las constantes son un tipo de variables particular, puesto que, tras ser definidas, no pueden modificar el valor que contienen durante la ejecución del programa, es decir, permanecen constantes. Si se desea modificar el valor contenido, se puede hacer accediendo al código y modificando el valor asignado previamente, a través del proceso que se describe a continuación.

Para definir el valor de las constantes, se utiliza la función `define()`, la cual recibe como argumentos obligatorios el nombre de la constante y el valor que se le asigna.

```
define(nombre, valor);
```

Código 9. Creación de constantes

Por otro lado, a diferencia de las variables, para acceder al valor de un elemento definido como constante, no debe escribirse el símbolo `$` antes del nombre. Por ejemplo, para asignar el valor 15 a la constante `IRPF`, la sintaxis correcta sería:

```
define("IRPF", 15);  
echo IRPF; #Correcto  
echo $IRPF; #Incorrecto
```

Código 10. Uso de constantes.

6.2. Tipos de datos

En el lenguaje de programación PHP, existen varios tipos de variables que se adecúan a los diferentes tipos de datos necesitamos manejar:

Tipo	Definición	Ejemplo
Integer	Se utiliza para crear variables con números enteros. En su creación, basta con indicar el número entero sin ningún otro carácter	<code>\$var=0;</code>
Float	Se utiliza para crear variables con números decimales. Para su creación, se utiliza el punto como símbolo separador de la parte entera y de la decimal	<code>\$var=0.1;</code>
Boolean	Se utiliza para almacenar los valores booleanos <code>true</code> o <code>false</code> . Se crea utilizando las cadenas booleanas sin comillas	<code>\$var=true;</code> <code>\$var2=false;</code>
String	Se utiliza para almacenar cadenas de texto. Se expresa mediante comillas	<code>\$var="hola";</code>
null	Este tipo se utiliza para indicar que la variable se encuentra vacía	<code>\$var=null;</code>
object	Elemento utilizado para almacenar los objetos en PHP	

Tabla 3. Tipos de datos en PHP.



En PHP, para indicar que una variable está vacía y que no contiene nada, se utiliza la palabra reservada null.

Además de los tipos simples descritos, existen otros elementos comunes en el desarrollo con este lenguaje. Estamos hablando de los arrays y los iterables.

- **Iterables:** Este elemento implementa un pseudotipo, que es un elemento que permite ser recorrido. Para realizar esta acción, se utiliza la función foreach.
- **Arrays:** Este tipo de elementos, muy común en todos los lenguajes de programación, permite organizar un conjunto de elementos en base a una clave de posición determinada. Para la creación de un array, se pueden utilizar varias sintaxis:

```
$array1=array ("1", "2"); //modo 1  
$array1=array (clave1 => "1",  
               clave2 => "2",); //modo 2
```

Código 11. Sintaxis creación de arrays en PHP.

La definición de las claves de la segunda sintaxis debe realizarse utilizando String o Integer, y son equivalentes expresiones tales como "1" y 1.

6.3. Conversión tipos de datos

La conversión entre los tipos de datos es común en todos lenguajes de programación, pero, sobre todo, en el desarrollo web en general. Existen diferentes métodos de conversión entre tipos.

La creación de variables en PHP no indica, durante su creación, el tipo, por lo que se define cuando esta recibe el dato concreto. Ahora bien, una vez se establece el tipo de contenido, es posible convertirlo. En PHP, encontramos algunas funciones útiles que realizan una versión directa a los tipos más comunes: enteros, decimales y cadenas de caracteres.

Tipo	Definición
string strval (mixed variable)	Convierte a cadena de texto cualquier variable que se le pasa como parámetro
integer intval (mixed variable)	Convierte a entero cualquier variable que se le pasa como parámetro
float floatval (mixed variable)	Convierte a decimal cualquier variable que se le pasa como parámetro

Tabla 4. Funciones conversión de datos.

Además de los métodos expuestos para la conversión de datos, PHP incorpora funciones de tipo booleano, que permiten averiguar el tipo de dato. Estas devuelven true, si la variable pasada por argumento es del tipo especificado en la función; en caso contrario, devolverá el valor false.



- **is_bool(nombreVariable).** Devuelve true si nombreVariable es un tipo de dato booleano.
- **is_string(nombreVariable).** Devuelve true si nombreVariable es un tipo de dato String, una cadena de texto.
- **is_intl(nombreVariable).** Devuelve true si nombreVariable es un tipo de dato entero.
- **is_numeric(nombreVariable).** Devuelve true si nombreVariable es un tipo de dato numérico, pero, además, considera como numérico una cadena de texto que contiene una secuencia de números. Esta función es muy útil, puesto que, en ocasiones, se pueden definir cajas de texto como tipo String, dentro de las cuales se van a introducir elementos numéricos.
- **is_null() o empty().** Resultan dos funciones últimas para comprobar si la variable se encuentra vacía.
- **is_array().** Devuelve true si se pasa por parámetro una variable de array.



Vídeo 1. "Creación y otras acciones con variables en PHP"
<https://bit.ly/3krP8cE>



6.4. Cadenas de texto

Una cadena de texto (String) es un conjunto de caracteres alfanuméricos que pueden estar formados por una sola palabra o por textos completos. Las cadenas de texto se definen mediante el uso de las **comillas dobles ("...")**, situando entre estos elementos cualquier cadena de texto que se vaya a utilizar.

Para extraer el contenido de las variables de tipo cadena de texto, se aconseja utilizar un par de llaves ({ }), entre las que se sitúa el nombre de la variable precedido del símbolo \$.

```
$contenidoVariable="hola";  
echo {$contenidoVariable};
```

Código 12. Creación e impresión de cadenas de texto.

Otra de las acciones más importantes, en lo que se refiere al tratamiento de cadenas de texto, es la unión entre dos o más String. Para concatenar dos elementos de texto independientes, se utiliza el punto (.), que se coloca entre las dos cadenas que se van a unir.

```
$var1 . $var2
```

Código 13. Concatenación de cadenas de texto.

En la concatenación de ejemplo anterior, se unen dos String, que dan como resultado «Hola mundo». No obstante, esta nueva cadena no ha sido almacenada en ningún sitio, es decir, se imprime, pero su valor se pierde. Para guardarlo, bastará con realizar la siguiente asignación:

```
$resultado= $var1 . $var2
```

Código 14. Concatenar y almacenar cadenas de texto.



Al escribir una línea de texto, o más de una, en ocasiones, será deseable que estas presenten ciertos elementos de formato, tales como un retorno de carro, un tabulador o, incluso, para poder imprimir caracteres reservados para el desarrollo en PHP, como las comillas o el símbolo del \$.

Tipo	Definición
\n	Salto de carro
\r	Retorno de carro
\\	Para imprimir una barra invertida
\\$	Para imprimir un símbolo de dólar
\"	Para imprimir las comillas dobles

Tabla 5. Caracteres reservados en PHP.

Algunas funciones útiles para el tratamiento de String en PHP son:

- **str_replace** ("cadenaOriginal", "cadenaNueva", \$cadenaTotal). Sustituye en cadenaTotal las ocurrencias de la cadenaOriginal por la cadenaNueva.
- **str_lower**(\$cadenaTotal) o **str_upper** (\$cadenaTotal). Modifica el contenido a minúsculas o mayúsculas.
- **strlen**(\$cadenaTotal). Indica la longitud de la cadena.
- **ltrim**(\$cadenaTotal). Elimina los espacios en blanco de la cadena.



Vídeo 2. "Constantes y funciones con cadenas"
<https://bit.ly/33K5Kqm>



/ 7. Caso práctico 1: "Crear e imprimir variables y tipos de datos"

Planteamiento: Para crear un primer programa con PHP que utilice todos los tipos de datos vistos en este tema, se pide diseñar un programa que crea y da valor a al menos 5 tipos de datos. Tras este proceso, se debe imprimir el valor de cada una de ellos, indicando el tipo exacto al que pertenecen.

Nudo: En primer lugar, se crean todas las variables y se les asigna un valor. A continuación, sería necesario comprobar de qué tipo son. Cuando cumpla la condición del tipo evaluado, se imprimirá su valor, junto el tipo al que pertenecen. Sin embargo, dado que aún no hemos visto cómo se implementan las estructuras de control con PHP, lo haremos de forma manual.



```
<?php
    $varTexto="hola"; // Cadena de texto
    $varEntero=1234; // Entero
    $varBoolean=true; // Boolean
    $varFloat=1.2; // Float
    $varArray= array ("1", "2"); //Array
    /* Para introducir un salto de carro y ser mostrado por pantalla utilizamos
    <b/r> */
    echo "Variable de tipo cadena de      texto: ". $varTexto . "<br/>";
    echo "Variable de tipo entero: ". $varEntero . "<br/>";
    echo "Variable de tipo boolean: ". $varBoolean . "<br/>";
    echo "Variable de tipo float: ". $varFloat . "<br/>";
    echo "Variable de tipo compuesto: " ;
        print_r($varArray);      #La función print_r es la utilizada para
    imprimir el contenido de un array
    ?>
```

Código 15. Programa Caso Práctico 1.

Desenlace: La salida resultante se muestra a continuación:

```
Variable de tipo cadena de texto: hola
Variable de tipo entero: 1234
Variable de tipo boolean: 1
Variable de tipo float: 1.2
Variable de tipo compuesto Array: Array ( [0] => 1 [1] => 2 )
```

Fig. 6. Imagen salida programa Caso Práctico 1.

/ 8. Caso práctico 2: “Crear e imprimir variables y tipos de datos con especificaciones”

Planteamiento: En este segundo caso, se desea volver a imprimir las variables creadas en el Caso Práctico 1, pero en base a las siguientes especificaciones:

- Los números decimales se mostrarán como enteros.
- El array contendrá en su interior los valores posibles de una variable de tipo boolean.
- La variable boolean no imprimirá ningún valor.
- La variable cadena de texto y la de tipo entero se concatenan en una sola línea.

Nudo: En primer lugar, se crean todas las variables y se les asigna el nuevo valor. Para cumplir con las especificaciones expuestas, se tendrán en cuenta los siguientes casos:

- Para concatenar cadenas se utiliza "."
- La concesión de decimal a entero se realiza utilizando la función intval().

```
<?php
    $varTexto="hola"; // Cadena de texto
    $varEntero=1234; // Entero
    $varBoolean=false; // Boolean
    $varFloat=1.2; // Float
    $varArray= array ("true", "false");
    /* Para introducir un salto de carro y ser mostrado por pantalla utilizamos
    <b/r> */
    echo "Variable concatenada: ". $varTexto . $varEntero "<br/>";
    echo "Variable de tipo boolean: ". $varBoolean . "<br/>";
    echo "Variable de tipo float: ". intval($varFloat) . "<br/>";
    echo "Variable de tipo compuesto: " ;
    print_r($varArray);    #La función print_r es la utilizada para imprimir el
    contenido de un array
    ?>
```

Código 16. Programa Caso Práctico 2.

Desenlace: La salida resultante se muestra a continuación:

```
Variable de tipo cadena de texto: hola1234
Variable de tipo boolean: 1
Variable de tipo float: 1
Variable de tipo compuesto: Array ( [0] => true [1] => false )
```

Fig. 7. Imagen salida programa Caso Práctico 2.

/ 9. Operadores

El desarrollo de cualquier programa requiere del uso de múltiples operadores, tanto para el cálculo y actualización del valor de ciertas variables, como para realizar procesos de comparación de elementos que determinen el resto de la ejecución.

Podemos distinguir entre varios bloques de operadores, destacando: **aritméticos, lógicos, de comparación y de asignación.**

Los operadores de tipo aritmético se utilizan para realizar todo tipo de operaciones: suma, resta, incremento, decremento...



Tipo	Expresión	Ejemplo
Incremento	++	\$var1++
Decremento	--	\$var1--
Suma	+	\$var1/\$var2
Resta	-	\$var1-\$var2
Multiplicación	*	\$var1*\$var2
División	/	\$var1/\$var2

Tabla 6. Operadores aritméticos.

Uno de los más utilizados para la evaluación de múltiples condicionantes son los de tipo lógico. Por ejemplo, se puede buscar, entre un bucle while, todos los valores que sean menores que 0 o mayores de 10.

Tipo	Expresión	Ejemplo
CONJUNCIÓN	&&	\$var1&&\$var2
DISYUNCIÓN		\$var1 \$var2
NEGACIÓN	!	!\$var1

Tabla 7. Operadores lógicos.

Los operadores de asignación permiten actualizar el valor contenido en una variable, ya sea eliminando el valor anterior para ingresar uno nuevo (=) o modificando el contenido operando sobre él.

Tipo	Expresión	Ejemplo
Asignación	=	\$var1=1
Suma y asigna	+=	Si \$var=1, el resultado de \$var+=2 sería 3, es decir, se actualiza el contenido inicial incrementado el valor indicado tras el operador de suma y asignación.
Resta y asigna	-=	
Multiplicación y asignación	*=	
División y asignación	/=	

Tabla 8. Operadores de asignación.



/ 10. Resumen y resolución del caso práctico de la unidad

A lo largo del este tema, hemos realizado una primera toma de contacto con algunos lenguajes de programación para el desarrollo web en entorno servidor, centrándonos, sobre todo, en PHP, puesto que será el que utilizaremos en este módulo.

Hemos visto, también, la creación y definición de variables. Los ámbitos de estas, así como analizar los principales tipos de datos, resulta clave para comenzar la implementación en cualquier lenguaje de programación. Algunos de los tipos de datos más importantes que se han visto son: enteros, cadenas de texto o String, booleanos o float. La conversión entre los tipos de datos mencionados será de gran utilidad para el correcto tratamiento de la información.

Además de estos datos de tipo sencillo, existen otros compuestos, como los iterables o los arrays. En el tema 3 nos adentraremos un poco más en el funcionamiento de estos últimos, puesto que resultan clave en el tratamiento de conexiones y sesiones desarrolladas en PHP, entre cliente y usuario.

Finalmente, se recogen en el último apartado los diferentes tipos de operadores. Estas tablas serán muy útiles y se deberán tener presentes para todo el desarrollo posterior con PHP.

Resolución del caso práctico de la unidad.

Como se ha visto en el apartado 9, existen diferentes tipos de operadores que recogen un amplio espectro de funcionalidades útiles para el desarrollo web: aritméticas, lógicas y de asignación. Pero, además, existe el bloque de los operadores de comparación, que son un elemento clave para la implementación de estructuras de control.

Los operadores de comparación resultan especialmente útiles, junto con las estructuras de control, que permiten determinar el camino del algoritmo que se seguirá en función del caso.

Tipo	Expresión
Igual que	'=='
Mismo tipo de dato y mismo valor	'===
Mayor / Mayor o igual	> >=
Menor / Menor o igual	< <=
Distinto a	!=

Tabla 9. Operadores de comparación.

Si, por ejemplo, se va a evaluar si un valor es superior a 0, se utilizará una bucle de tipo while, que, mientras que el valor sea superior a 0, (while \$valor>0), continuará pidiendo un nuevo valor.



/ 11. Bibliografía

Ganzabal, X. (2019). Desarrollo web en entorno servidor. (1.a. ed.). Madrid: Síntesis.

Vara, J. M. (2012). Desarrollo en entorno Servidor. (1.a ed.). Madrid: Rama.

/ 12. Webgrafía

PHP. Recuperado de: <https://www.php.net/manual/es/intro-what-is.php>

Programación Lado Servidor. Recuperado de: <https://developer.mozilla.org/es/docs/Learn/Server-side>

MEDAC