DÍA 10 – Control de flujo, estructuras condicionales y funciones básicas en PHP

Pablo Macías Salguero | 02/04/2025

15 – Operadores de incremento & decremento:

Ejemplo	Nombre	Efecto
++$a	Pre-incremento	Incrementa $a en uno, y luego devuelve $a.
$a++	Post-incremento	Devuelve $a, y luego incrementa $a en uno.
--$a	Pre-disminución	Disminuye $a en uno, y luego devuelve $a.
$a--	Post-disminución	Devuelve $a, y luego disminuye $a en uno.

Cuando es pre-incremento o pre-decremento primero realiza la acción y luego muestra el resultados, mientras que cuando usamos post primero muestra el valor de la variable y luego hace la acción (incrementar o decrementar).

```php
// Incremento
$a = 10;
$b = 10;
// Pre-incremento
echo ++$a; // 11
// Post-incremento
echo $b++; // 11
```

```php
// Decremento
$c = 10;
$d = 10;
// Pre-decremento
echo --$c; // 9
// Post-decremento
echo $d--; // 9
```

16 – Estructura condicional simple (if):

La estructura if permite ejecutar un bloque de código si se cumple una condición determinada.

Por ejemplo, si tiene más de 18 años se muestra ‘Eres mayor de edad’.

```php
$edad = 18;

if ($edad >= 18) {
    echo "Eres mayor de edad.";
}
```

Otro ejemplo:

```php
$a = 10;
$b = 20;

if ($a > $b) {
    echo "$a es mayor que $b";
} else {
    echo "$a es igual o menor que $b";
}
```

17 – Ejemplos de estructura condicional simple:

```php
$numero = 10;

if ($numero > 5) {
    echo "El número es mayor que 5.";
}
```

```php
$edad = 20;

if ($edad >= 18) {
    echo "Eres mayor de edad.";
}
```

```php
$nota = 7;

if ($nota >= 6) {
    echo "¡Felicidades, has aprobado!";
}
```

18 – Estructura condicional doble (if-else):

Esta estructura condicional permite ejecutar un bloque de código si se cumple una condición, y ejecutar otro bloque de código si la condición no se cumple.

```php
$edad = 16;

if ($edad >= 18) {
    echo "Eres mayor de edad.";
} else {
    echo "Eres menor de edad.";
}
```

En este ejemplo se puede ver como si la persona es mayor de edad se muestra un mensaje, pero si no lo es, se muestra otro. Esto lo conseguimos con la condición ‘else’.

19 – Ejemplos de estructura condicional doble:

Aquí se muestran más ejemplos, por ejemplo, una estructura condicional para comprobar si un número es negativo o positivo:

```php
$numero = -5;

if ($numero > 0) {
    echo "El número es positivo.";
} else {
    echo "El número es negativo o cero.";
}
```

Si el número es menor de cero significa que es negativo, por lo tanto aprovechamos esa condición.

Otra estructura condicional para ver si un número es par o impar:

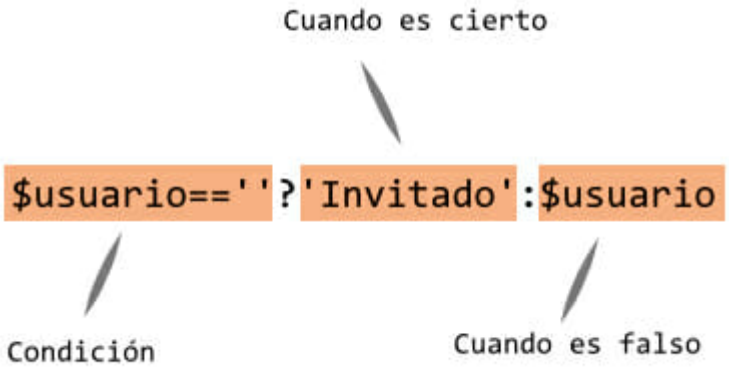
```php
$numero = 7;

if ($numero % 2 == 0) {
    echo "El número es par.";
} else {
    echo "El número es impar.";
}
```

Si el resto de la división de un número entre dos es distinto de cero significa que es impar, por lo tanto usamos esa condición para identificar el número.

20 – Operador ternario en PHP:

Un operador ternario en PHP es una forma compacta de escribir una estructura de control if-else, esta es la estructura:



En este ejemplo se puede ver que si se cumple la condición $mensaje toma el valor ‘Eres mayor de edad’, si no se cumple pasa a valer ‘Eres menor de edad’, que es la segunda opción.

```php
// Operador ternario

$edad = 20;
$mensaje = ($edad >= 18) ? "Eres mayor de edad" : "Eres menor de edad";
echo $mensaje;
```

21 – Ejemplo de operador ternario:

Este ejemplo aplica un descuento u otro dependiendo del número de camisetas que se vayan a comprar:

```php
$numCamisetas = 5;
$precioCamiseta = 10;
$precioTotal = $numCamisetas * $precioCamiseta;

$descuento = ($numCamisetas >= 3) ? $precioTotal-($precioTotal*0.20) : $precioTotal-($precioTotal*0.10);
echo "El total a pagar es $descuento euros";
echo "<br><br>";
```

22 – Estructura condicional múltiple (if-elseif):

Es una forma de evaluar varias condiciones y ejecutar diferentes bloques de código según la condición que se cumpla.

Ejemplo que muestra el día de la semana en función de un número del 1 al 7. Este ejemplo va comparando el número de la semana con los condicionales hasta que encuentra uno que se cumpla. En este caso como el día 3 es el miércoles, se muestra 'Miercoles' por pantalla.

```php
$dia = 3;

if ($dia == 1) {
    echo "Lunes";
} elseif ($dia == 2) {
    echo "Martes";
} elseif ($dia == 3) {
    echo "Miércoles";
} elseif ($dia == 4) {
    echo "Jueves";
} elseif ($dia == 5) {
    echo "Viernes";
} elseif ($dia == 6) {
    echo "Sábado";
} elseif ($dia == 7) {
    echo "Domingo";
} else {
    echo "No es un día de la semana válido";
}
```

23 – Ejemplos estructura condicional múltiple:

Este es un ejemplo que calcula el precio de una serie de ordenadores en función a la cantidad:

```php
$numOrdenadores = 3;
$precioOrdenador = 700;

if ($numOrdenadores < 5) {
    $precioTotal = ($numOrdenadores * $precioOrdenador) - ($numOrdenadores * $precioOrdenador * 0.10);
} elseif ($numOrdenadores >= 5 && $numOrdenadores < 10) {
    $precioTotal = $numOrdenadores * $precioOrdenador - ($numOrdenadores * $precioOrdenador * 0.20);
} elseif ($numOrdenadores > 10) {
    $precioTotal = $numOrdenadores * $precioOrdenador - ($numOrdenadores * $precioOrdenador * 0.40);
} else {
    echo "No se ha podido calcular el precio total";
}

echo "El precio total es $precioTotal euros";
```

24 – Condicionales anidadas:

Las estructuras condicionales son condicionales que contienen otros condicionales dentro de su estructura. Por ejemplo, en el siguiente ejemplo primero se evalúa el género de la persona y cuando ha encontrado una condición que se cumpla pasa a comparar la edad.

```php
// Estructura condicional anidada

// Ejemplo para comprobar si una persona se puede jubilar o no
$edad = 65;
$sexo = "hombre";

if ($sexo == "hombre") {
    if ($edad >= 65) {
        echo "Puede jubilarse";
    } else {
        echo "No puede jubilarse";
    }
} elseif ($sexo == "mujer") {
    if ($edad >= 60) {
        echo "Puede jubilarse";
    } else {
        echo "No puede jubilarse";
    }
} else {
    echo "Sexo no válido";
}
```

25 – Estructura de selección múltiple switch:

Esta estructura, a diferencia de if-else, permite evaluar una única expresión frente a múltiples posibles valores que puede tomar para luego ejecutar el código correspondiente al valor que coincide. Este ejemplo compara el valor de la variable 'fruta' con posibles valores que podría tener. Dependiendo del valor que coincida, ejecuta un bloque de código u otro.

```php
$fruta = "manzana";

switch ($fruta) {
    case "manzana":
        echo "La fruta es una manzana";
        break;
    case "naranja":
        echo "La fruta es una naranja";
        break;
    case "plátano":
        echo "La fruta es un plátano";
        break;
    case "pera":
        echo "La fruta es una pera";
        break;
    default:
        echo "No se ha encontrado la fruta";
}
echo "<br><br>";
```

```php
switch ($dia) {
    case 1:
        echo "Lunes";
        break;
    case 2:
        echo "Martes";
        break;
    case 3:
        echo "Miércoles";
        break;
    case 4:
        echo "Jueves";
        break;
    case 5:
        echo "Viernes";
        break;
    case 6:
        echo "Sábado";
        break;
    case 7:
        echo "Domingo";
        break;
    default:
        echo "No es un día de la semana válido";
}
```

Switch no tiene en cuenta el tipo de variable, solo tiene en cuenta el valor.

26 – Estructura de selección múltiple match:

El match permite comparar una expresión con diferentes valores posibles, y tiene un comportamiento más estricto que el switch, lo que reduce algunos errores comunes al trabajar con estructuras de selección múltiple.

```php
// Estructura de control match

$a = 7;

$x = 10;
$y = 20;
$y = 7;

$resultado = match($a) {
    $x => "Valor igual a x",
    $y => "Valor igual a y",
    $z => "Valor igual a z",
    default => "No coincide con ninguna variable"
};

echo $resultado . "<br>";
```

```php
$edad = 18;

$resultado = match(true) {
    $edad < 18 => "Menor de edad",
    $edad >= 18 && $edad < 65 => "Adulto",
    default => "Adulto mayor"
};

echo $resultado . "<br>";
```

27 – Ciclo while:

Es un bucle, una estructura que se repite mientras se cumpla una condición. En el siguiente ejemplo se puede ver como el bucle se repite mientras la variable contador tenga un valor menor o igual a 1:

```php
// Bucle while

// Contador de números desde 1 a 20
$contador = 1;
while($contador <= 20) {
    echo $contador . "<br>";
    $contador++;
}
```

La condición se evalúa antes de cada iteración, por lo que si la condición es falsa desde el principio, el bloque de código no se ejecutará ni una sola vez.

28 – Ciclo do while:

El bucle do-while es similar al while con la diferencia de que la condición se evalúa después de ejecutar el bloque de código, lo que significa que se ejecutará siempre al menos una vez aunque la condición sea falsa desde el principio.

Aquí muestro algunos ejemplos:

```php
// Contador de números desde 1 a 20
$contador = 1;
do {
    echo $contador . "<br>";
    $contador++;
} while($contador <= 20);
```

```php
// Tabla de multiplicar
$num = 5;
$i = 1;
do {
    echo "$num x $i = " . ($num * $i) . "<br>";
    $i++;
} while($i <= 10);
```

5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50

```php
$num = 7;
$i = 12;
do {
    echo "$num x $i = " . ($num * $i) . "<br>";
    $i--;
} while($i >= 1);
```

7 x 12 = 84
7 x 11 = 77
7 x 10 = 70
7 x 9 = 63
7 x 8 = 56
7 x 7 = 49
7 x 6 = 42
7 x 5 = 35
7 x 4 = 28
7 x 3 = 21
7 x 2 = 14
7 x 1 = 7

29 – Ciclo for:

El bucle for se utiliza para repetir un bloque de código un número definido de veces. A diferencia de los bucles while y do-while, el bucle for se utiliza cuando sabes de antemano cuántas veces deseas que se ejecute el bloque de código.

Ejemplos:

```php
for ($i=1; $i<=20; $i++) {
    echo $i."<br>";
}

for ($i=20; $i>=1; $i--) {
    echo $i."<br>";
}
```

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

```php
// Tabla multiplicar del 7
$num = 7;
for ($i=1; $i<=12; $i++) {
    echo $num." x ".$i." = ".$num*$i."<br>";
}
```

7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
7 x 11 = 77
7 x 12 = 84

30 – Ciclo foreach:

Esta estructura se utiliza con arrays u objetos. A diferencia de otros bucles como for o while, foreach simplifica el proceso de iteración, ya que no necesitas preocuparte por los índices o claves de los elementos, lo que lo hace más legible y fácil de usar cuando trabajas con colecciones de datos.

Ejemplos:

```php
$colores = ["rojo", "azul", "verde", "amarillo"];
foreach ($colores as $color) {
    echo $color . "<br>";
}
```

rojo
azul
verde
amarillo

```php
$frutas = [
    "manzana" => "roja",
    "plátano" => "amarillo",
    "uva" => "morado",
    "naranja" => "naranja"
];
foreach ($frutas as $fruta => $color) {
    echo "La $fruta es de color $color.<br>";
}
```

La manzana es de color roja.
La plátano es de color amarillo.
La uva es de color morado.
La naranja es de color naranja.

```php
$perifericos = [
    ["codigo" => "A0001", "descripcion" => "raton"],
    ["codigo" => "A0002", "descripcion" => "teclado"],
    ["codigo" => "A0003", "descripcion" => "monitor"],
    ["codigo" => "A0004", "descripcion" => "impresora"]
];
foreach ($perifericos as $periferico) {
    echo "Codigo: {$periferico['codigo']} Nombre: {$periferico['descripcion']}<br>";
}
```

Codigo: A0001 Nombre: raton
Codigo: A0002 Nombre: teclado
Codigo: A0003 Nombre: monitor
Codigo: A0004 Nombre: impresora

31 – Break & continue:

- Break: Se utiliza para salir de un bucle (como for, while, foreach) o de una estructura switch, deteniendo su ejecución antes de que se cumpla la condición de salida natural.

En este ejemplo vemos cómo antes de mostrar ‘GPU’ y ‘CPU’ la ejecución termina y por lo tanto no los muestra.

```php
// Break: Termina el bucle
$pc = ["SO", "RAM", "SSD", "GPU", "CPU"];
foreach ($pc as $component) {
    if ($component === "GPU") {
        break;
    }
    echo $component . "<br>";
}
```

SO
RAM
SSD

- Continue: Se utiliza para omitir el resto del código dentro de un bucle y pasar directamente a la siguiente iteración del bucle.

En el siguiente ejemplo hemos puesto un if para que cuando el programa llegue a ‘SSD’ salte a la siguiente iteración. Por lo tanto no aparece en el resultado:

```php
// Continue: Salta a la siguiente iteración
$pc = ["SO", "RAM", "SSD", "GPU", "CPU"];
foreach ($pc as $component) {
    if ($component === "SSD") {
        continue;
    }
    echo $component . "<br>";
}
```

SO
RAM
GPU
CPU

Otros ejemplos:

```php
for ($i=1; $i<=10; $i++) {
    if ($i==5) {
        continue; // Salta el 5
    }
    echo $i . "<br>";
}
```

1
2
3
4
6
7
8
9
10

```php
while ($i<=10) {
    if ($i==3) {
        continue; // Salta el 3
    }
    echo $i . "<br>";
    $i++;
}
```

1
2
3
4
6
7
8
9
10

32 – Incluir archivo PHP con require e include:

- Require: Incluye el archivo especificado y detiene la ejecución del script si el archivo no se puede incluir (por ejemplo, si el archivo no existe).
- Include: Similar a require, pero no detiene la ejecución del script si el archivo no se puede incluir. El código continuará ejecutándose aunque no se encuentre el archivo.

Ejemplos de ambos:

```php
// Rquire e Include
require '../bucles/doWhile/index.php';
include '../bucles/doWhile/index.php';

// Require_once e Include_once - se incluye una sola vez
require_once '../bucles/doWhile/index.php';
include_once '../bucles/doWhile/index.php';
```

- Require_once e Include_once: Incluye el archivo solo una vez, incluso si se llama varias veces en el código. Esto evita incluir el archivo más de una vez y previene errores de redefinición de funciones o clases.

Ejemplos:

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <?php include_once 'nav.php' ?>

    <h1>Ejemplo de include y require</h1>
    <p>Este es un ejemplo de cómo incluir archivos en PHP.</p>

    <?php include_once 'footer.php' ?>
</body>
</html>
```

33 – Crear y usar funciones propias en PHP:

Para evitar la redundancia de código usamos las funciones. Son bloques de código que podemos reutilizar para no volver a escribirlo.

Para crear una función lo hacemos con ‘function’ y el nombre que le queramos dar. Luego entre paréntesis podemos poner el nombre de unas variables por si le queremos pasar parámetros. Aquí hay algunos ejemplos.

Esta función simplemente muestra texto por pantalla, no recibe parámetros, así que cuando la llamamos muestra ‘ Hola, mi nombre es Juan’.

```php
// Crear y usar funciones
function saludo() {
    echo 'Hola, mi nombre es Juan<br>';
}
saludo();
```

Esta función si tiene parámetros, así que cuando la llamamos tenemos que darle valor. En este caso lo que hace la función es sumar los valores. Luego con return devuelve el resultado, por eso igualamos la función a una variable, porque el resultado de la ejecución devuelve un valor que luego mostraremos por pantalla:

```php
// Return
function suma($a, $b) {
    return $a + $b;
}
$resultado = suma(5, 10);
echo 'La suma es: ' . $resultado . '<br>';
```

```php
// Parámetros
function sumaConParametros($a, $b) {
    return $a + $b;
}
$resultado = sumaConParametros(5, 2);
echo 'La suma con parámetros es: ' . $resultado . '<br>';
```

```php
// Media notas
function mediaNotas($a, $b, $c) {
    return ($a + $b + $c) / 3;
}
$media = mediaNotas(5, 8, 10);
echo 'La media de las notas es: ' . $media . '<br>';
```

34 – Incluir y llamar funciones desde otro archivo:

Para incluir una función externa tenemos que incluir el archivo en el que se encuentra. Luego podemos llamar a esa función usando su nombre:

```php
// Incluir funciones en PHP
include 'funciones.php';

echo "La media es: ".mediaNotas(5, 8, 10) . "<br>";
```

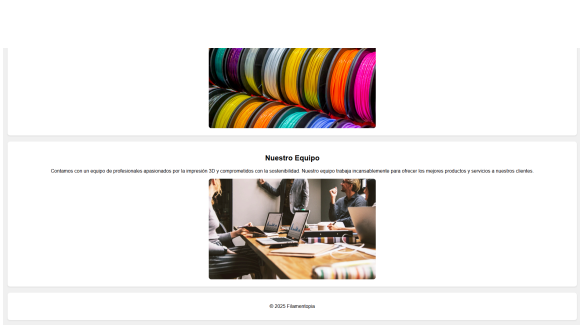
Hola, mi nombre es Juan
La suma es: 15
La suma con parámetros es: 7
La media de las notas es: 7.6666666666667
La media es: 7.6666666666667

Proyecto:

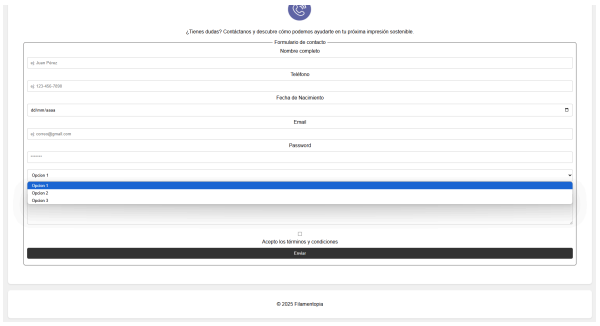
Por ahora el proyecto ha sido desarrollado con HTML y CSS, pero ahora que estamos repasando PHP iremos incluyendo funcionalidades de PHP que puedan mejorar nuestro proyecto.

Inicio:

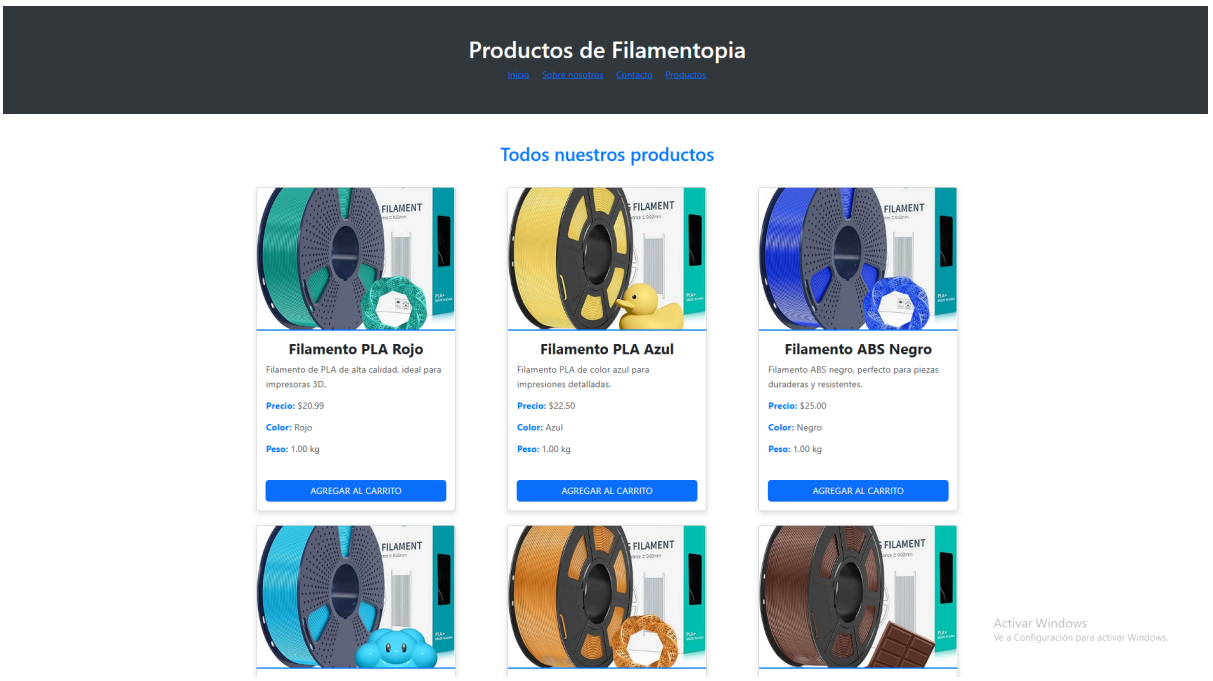
Sobre nosotros:



Contacto:



Productos:



He añadido una nueva sección. Este apartado es una tienda para comprar productos, filamento para impresión 3d en este caso. Para ello he creado una base de datos con productos y a través de PHP accedo a los datos y los mostramos en la web.

Más adelante iré añadiendo funcionalidades como un carrito, registro, etc.

Por ahora tenemos la base de datos y la página para mostrar los productos con sus datos, imágenes y botones.

Enlace al protecto en GitHub:
https://github.com/pmacsal2707/RepasoConceptos/tree/master/proyecto