

Ambientes no propietarios

Expresiones y control de flujo en PHP

Edwin Salvador

28 de abril de 2015

Sesión 4

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Organización Agile para startups

- **Día** Martes, 5 de mayo 2015
- **Hora** 18:00
- **Lugar** por definir
- **Tema 1** Pensamiento LEAN y Agile en la Organización
- **Tema 2** Lean StartUp y Desarrollo de Software Agile: Creando productos memorables y Startups exitosos

Contenido I

- 1 Invitación a charla
- 2 Preguntas**
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Preguntas

- ❶ ¿Cuál es la etiqueta que nos permite indicarle al servidor web que se debe interpretar el código como PHP?
- ❷ ¿Cuál es el caracter que le indica al interprete el final de cada sentencia?
- ❸ ¿Para que nos sirve el símbolo \$ en el lenguaje PHP?
- ❹ ¿Qué podemos almacenar en una variable PHP?
- ❺ ¿Cómo podemos convertir a una variable de un tipo a otro (string a número)?
- ❻ Responda verdadero o falso:
 - PHP es case-sensitive:
 - Los espacios son aceptados en los nombres de variables:
- ❼ ¿Cuál es la diferencia entre `++$j;` y `$j++;`?
- ❽ ¿Se puede redefinir una constante en PHP?, si la respuesta es si escriba cómo?, Si la respuesta es no, explique porqué?
- ❾ ¿Cómo se puede hacer que una variable sea accedida desde cualquier parte del código?

Preguntas II

- 10 Dijimos que las funciones en PHP son “aisladas del mundo exterior” es decir las variables declaradas fuera de ellas no son accesibles dentro y las variables declaradas dentro de ellas no son accesibles fuera de ellas.
- a Explique con un ejemplo cómo podemos acceder dentro de una función a una variable que ha sido declarada fuera de ella.
 - b Explique con un ejemplo cómo podemos acceder fuera de una función a una variable que ha sido declarada dentro de ella.

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 **Ejercicio**
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Ejercicio

¿Cuál es el resultado de las siguientes expresiones y por qué?

- `echo 'hola' . 'mundo';`
- `echo 'hola' + 'mundo';`
- `echo 4 + 'hola';`
- `echo 'hola' + 4;`
- `echo 4 + 'hola' . 4;`

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones**
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

TRUE or FALSE

- Las expresiones booleanas en PHP tienen la misma funcionalidad que en otros lenguajes de programación.

TRUE or FALSE

- Las expresiones booleanas en PHP tienen la misma funcionalidad que en otros lenguajes de programación.
- `20 > 5` es TRUE.

TRUE or FALSE

- Las expresiones booleanas en PHP tienen la misma funcionalidad que en otros lenguajes de programación.
- `20 > 5` es TRUE.
- `5 == 6` es FALSE.

TRUE or FALSE

- Las expresiones booleanas en PHP tienen la misma funcionalidad que en otros lenguajes de programación.
- `20 > 5` es TRUE.
- `5 == 6` es FALSE.
- Se pueden combinar con AND, OR, XOR para formar expresiones más complejas.

TRUE or FALSE

- Las expresiones booleanas en PHP tienen la misma funcionalidad que en otros lenguajes de programación.
- `20 > 5` es TRUE.
- `5 == 6` es FALSE.
- Se pueden combinar con AND, OR, XOR para formar expresiones más complejas.
- Se puede utilizar cualquier combinación de mayúsculas y minúsculas para los valores de TRUE o FALSE (`true`, `TRUE`, `tRuE`, `TrUE`, etc). Todas son variables predefinidas en el lenguaje.

TRUE or FALSE

- Las expresiones booleanas en PHP tienen la misma funcionalidad que en otros lenguajes de programación.
- `20 > 5` es TRUE.
- `5 == 6` es FALSE.
- Se pueden combinar con AND, OR, XOR para formar expresiones más complejas.
- Se puede utilizar cualquier combinación de mayúsculas y minúsculas para los valores de TRUE o FALSE (`true`, `TRUE`, `tRuE`, `TrUE`, etc). Todas son variables predefinidas en el lenguaje.
- Se recomienda el uso de la versión en minúsculas (`true`, `false`). La versión en mayúsculas pueden ser redefinidas las minúsculas no.

Ejemplo

¿Cuál es la salida de las siguientes expresiones?

```
<?php
echo "a: [" . (20 > 9) . "]"<br>";
echo "b: [" . (5 == 6) . "]"<br>";
echo "c: [" . (1 == 0) . "]"<br>";
echo "d: [" . (1 == 1) . "]"<br>";
?>
```

En PHP false está definido como NULL. En otros lenguajes se define como 0 o -1.

Literales y variables

```
<?php
    $myname = "Brian";
    $myage = 37;
    echo "a: " . 73 . "<br>"; // literal numerica
    echo "b: " . "Hello" . "<br>"; // literal String
    echo "c: " . FALSE . "<br>"; // literal constante
    echo "d: " . $myname . "<br>"; // variable String
    echo "e: " . $myage . "<br>"; // variable numerica
?>
```

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

- **Unarios**

- **Unarios** $a++$, $--a$, $-a$

Operadores

- **Unarios** $a++$, $--a$, $-a$
- **Binarios**

Operadores

- **Unarios** $a++$, $--a$, $-a$
- **Binarios** $+$, $-$, $*$, $/$

Operadores

- **Unarios** $a++$, $--a$, $-a$
- **Binarios** $+$, $-$, $*$, $/$
- **Ternario** . Una abreviación del `if`.

Operadores

- **Unarios** $a++$, $--a$, $-a$
- **Binarios** $+$, $-$, $*$, $/$
- **Ternario** $? x : y$. Una abreviación del `if`.

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?
 - $1 + 2 + 3 - 4 + 5$

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?
 - $1 + 2 + 3 - 4 + 5$
 - $2 - 4 + 5 + 3 + 1$

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?
 - $1 + 2 + 3 - 4 + 5$
 - $2 - 4 + 5 + 3 + 1$
 - $5 + 2 - 4 + 1 + 3$

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?
 - $1 + 2 + 3 - 4 + 5$
 - $2 - 4 + 5 + 3 + 1$
 - $5 + 2 - 4 + 1 + 3$
- ¿Y de las siguientes?

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?
 - $1 + 2 + 3 - 4 + 5$
 - $2 - 4 + 5 + 3 + 1$
 - $5 + 2 - 4 + 1 + 3$
- ¿Y de las siguientes?
 - $1 * 2 * 3 / 4 * 5$

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?

- $1 + 2 + 3 - 4 + 5$

- $2 - 4 + 5 + 3 + 1$

- $5 + 2 - 4 + 1 + 3$

- ¿Y de las siguientes?

- $1 * 2 * 3 / 4 * 5$

- $2 / 4 * 5 * 3 * 1$

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?

- $1 + 2 + 3 - 4 + 5$

- $2 - 4 + 5 + 3 + 1$

- $5 + 2 - 4 + 1 + 3$

- ¿Y de las siguientes?

- $1 * 2 * 3 / 4 * 5$

- $2 / 4 * 5 * 3 * 1$

- $5 * 2 / 4 * 1 * 3$

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?

- $1 + 2 + 3 - 4 + 5$
- $2 - 4 + 5 + 3 + 1$
- $5 + 2 - 4 + 1 + 3$

- ¿Y de las siguientes?

- $1 * 2 * 3 / 4 * 5$
- $2 / 4 * 5 * 3 * 1$
- $5 * 2 / 4 * 1 * 3$

- ¿Y estas?

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?

- $1 + 2 + 3 - 4 + 5$
- $2 - 4 + 5 + 3 + 1$
- $5 + 2 - 4 + 1 + 3$

- ¿Y de las siguientes?

- $1 * 2 * 3 / 4 * 5$
- $2 / 4 * 5 * 3 * 1$
- $5 * 2 / 4 * 1 * 3$

- ¿Y estas?

- $1 + 2 * 3 - 4 * 5$

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?

- $1 + 2 + 3 - 4 + 5$
- $2 - 4 + 5 + 3 + 1$
- $5 + 2 - 4 + 1 + 3$

- ¿Y de las siguientes?

- $1 * 2 * 3 / 4 * 5$
- $2 / 4 * 5 * 3 * 1$
- $5 * 2 / 4 * 1 * 3$

- ¿Y estas?

- $1 + 2 * 3 - 4 * 5$
- $2 - 4 * 5 * 3 + 1$

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?

- $1 + 2 + 3 - 4 + 5$
- $2 - 4 + 5 + 3 + 1$
- $5 + 2 - 4 + 1 + 3$

- ¿Y de las siguientes?

- $1 * 2 * 3 / 4 * 5$
- $2 / 4 * 5 * 3 * 1$
- $5 * 2 / 4 * 1 * 3$

- ¿Y estas?

- $1 + 2 * 3 - 4 * 5$
- $2 - 4 * 5 * 3 + 1$
- $5 + 2 - 4 + 1 * 3$

Precedencia de operadores

- ¿Cuál es el resultado de las siguientes expresiones?

- $1 + 2 + 3 - 4 + 5$
- $2 - 4 + 5 + 3 + 1$
- $5 + 2 - 4 + 1 + 3$

- ¿Y de las siguientes?

- $1 * 2 * 3 / 4 * 5$
- $2 / 4 * 5 * 3 * 1$
- $5 * 2 / 4 * 1 * 3$

- ¿Y estas?

- $1 + 2 * 3 - 4 * 5$
- $2 - 4 * 5 * 3 + 1$
- $5 + 2 - 4 + 1 * 3$

- ¿Cuál sería el resultado de las últimas si no tuviéramos la precedencia de operadores?

Ejercicio

Escribir las expresiones del último ejemplo de tal manera que los resultados coincidan con el caso de no existir la precedencia de operadores.

Ejercicio

Escribir las expresiones del último ejemplo de tal manera que los resultados coincidan con el caso de no existir la precedencia de operadores.

- $((1 + 2) * 3 - 4) * 5$

Ejercicio

Escribir las expresiones del último ejemplo de tal manera que los resultados coincidan con el caso de no existir la precedencia de operadores.

- $((1 + 2) * 3 - 4) * 5$
- $(2 - 4) * 5 * 3 + 1$

Ejercicio

Escribir las expresiones del último ejemplo de tal manera que los resultados coincidan con el caso de no existir la precedencia de operadores.

- $((1 + 2) * 3 - 4) * 5$
- $(2 - 4) * 5 * 3 + 1$
- $(5 + 2 - 4 + 1) * 3$

Precedencia de operadores

Operator(s)	Type
()	Parentheses
++ --	Increment/decrement
!	Logical
* / %	Arithmetic
+ - .	Arithmetic and string
<< >>	Bitwise
< <= > >= <>	Comparison
== != === !==	Comparison

Precedencia de operadores

Operator(s)	Type
&	Bitwise (and references)
^	Bitwise
	Bitwise
&&	Logical
	Logical
? :	Ternary
= += -= *= /= . = %= &= != ^= <<= >>=	Assignment
and	Logical
xor	Logical
or	Logical

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - **Asociatividad**
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Asociatividad de operadores (orden de evaluación)

Operator	Description	Associativity
CLONE NEW	Create a new object	None
< <= >= == != === !== <>	Comparison	None
!	Logical NOT	Right
~	Bitwise NOT	Right
++ --	Increment and decrement	Right
(int)	Cast to an integer	Right
(double) (float) (real)	Cast to a floating-point number	Right
(string)	Cast to a string	Right
(array)	Cast to an array	Right
(object)	Cast to an object	Right
@	Inhibit error reporting	Right
= += -= *= /=	Assignment	Right
.= %= &= = ^= <<= >>=	Assignment	Right
+	Addition and unary plus	Left

Asociatividad de operadores (orden de evaluación)

Operator	Description	Associativity
-	Subtraction and negation	Left
*	Multiplication	Left
/	Division	Left
%	Modulus	Left
.	String concatenation	Left
<< >> & ^	Bitwise	Left
?:	Ternary	Left
&& and or xor	Logical	Left
,	Separator	Left

Ejemplo (Asociatividad)

```
<?php  
    $level = $score = $time = 0;  
?>
```


Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Operador de igualdad (==) y de identidad (===)

Ejercicio: ¿Cuál es la salida del siguiente código?

```
<?php
    $a = "1000";
    $b = "+1000";
    if ($a == $b) echo "1";
    if ($a === $b) echo "2";
?>
```

- El operador de igualdad (==) convierte los datos a los tipos que crea más conveniente. En el ejemplo convierte los elementos a enteros y los compara.

Operador de igualdad (==) y de identidad (===)

Ejercicio: ¿Cuál es la salida del siguiente código?

```
<?php
    $a = "1000";
    $b = "+1000";
    if ($a == $b) echo "1";
    if ($a === $b) echo "2";
?>
```

- El operador de igualdad (==) convierte los datos a los tipos que crea más conveniente. En el ejemplo convierte los elementos a enteros y los compara.
- El operador de identidad (===) evita que PHP convierta los tipos de datos. \$a y \$b son comparados como strings.

Operador de igualdad (==) y de identidad (===)

Ejercicio: ¿Cuál es la salida del siguiente código?

```
<?php
    $a = "1000";
    $b = "+1000";
    if ($a == $b) echo "1";
    if ($a === $b) echo "2";
?>
```

- El operador de igualdad (==) convierte los datos a los tipos que crea más conveniente. En el ejemplo convierte los elementos a enteros y los compara.
- El operador de identidad (===) evita que PHP convierta los tipos de datos. \$a y \$b son comparados como strings.
- Se recomienda utilizar el operador identidad cuando no se está seguro de como serán convertidos los tipos de datos.

Operador de igualdad (==) y de identidad (===)

Ejercicio: ¿Cuál es la salida del siguiente código?

```
<?php
    $a = "1000";
    $b = "+1000";
    if ($a == $b) echo "1";
    if ($a === $b) echo "2";
?>
```

- El operador de igualdad (==) convierte los datos a los tipos que crea más conveniente. En el ejemplo convierte los elementos a enteros y los compara.
- El operador de identidad (===) evita que PHP convierta los tipos de datos. \$a y \$b son comparados como strings.
- Se recomienda utilizar el operador identidad cuando no se está seguro de como serán convertidos los tipos de datos.
- No igual (!=) y no idénticos (!==).

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Operadores de comparación

- $>$, \geq , $<$, \leq .

Operadores de comparación

- >, >=, <, <= .
- No hay nada especial que decir sobre estos operadores en PHP.

```
<?php
    $a = 2; $b = 3;
    if ($a > $b) echo "$a is greater than $b<br>";
    if ($a < $b) echo "$a is less than $b<br>";
    if ($a >= $b) echo "$a is greater than or equal to $b<br>";
    if ($a <= $b) echo "$a is less than or equal to $b<br>";
?>
```


Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Operadores lógicos

¿Cuál es la salida de:

```
<?php
    $a = 1; $b = 0;
    echo ($a AND $b) . "<br>";
    echo ($a or $b) . "<br>";
    echo ($a XOR $b) . "<br>";
    echo !$a . "<br>";
?>
```

- **Importante:** El operador OR no evaluará el segundo operando si el primero es verdadero.

```
<?php
    $a = 1;
    $b = 0;
    if($a || $b++) echo $b;
?>
```

Operadores lógicos

¿Cuál es la salida de:

```
<?php
    $a = 1; $b = 0;
    echo ($a AND $b) . "<br>";
    echo ($a or $b) . "<br>";
    echo ($a XOR $b) . "<br>";
    echo !$a . "<br>";
?>
```

- **Importante:** El operador OR no evaluará el segundo operando si el primero es verdadero.

```
<?php
    $a = 1;
    $b = 0;
    if($a || $b++) echo $b;
?>
```

- ¿Cómo podemos reescribir el ejemplo anterior para que la salida de \$b sea 1?

Operadores lógicos

Inputs		Operators and results		
a	b	AND	OR	XOR
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	TRUE
FALSE	TRUE	FALSE	TRUE	TRUE
FALSE	FALSE	FALSE	FALSE	FALSE

!TRUE es FALSE y !FALSE es TRUE.

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales**
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Condicionales

```
<?php
    if ($bank_balance < 100)
    {
        $money = 1000;
        $bank_balance += $money;
    }
?>
```

- **Recomendación** utilizar siempre las llaves ({}).

Condicionales

```
<?php
    if ($bank_balance < 100)
    {
        $money = 1000;
        $bank_balance += $money;
    }
?>
```

- **Recomendación** utilizar siempre las llaves ({}).
- **Indentar** el contenido de los condicionales y funciones.

Condicionales

```
<?php
    if ($bank_balance < 100)
    {
        $money = 1000;
        $bank_balance += $money;
    }
?>
```

- **Recomendación** utilizar siempre las llaves ({}).
- **Indentar** el contenido de los condicionales y funciones.
- **else** es opcional.

elseif

- Cuando se tienen más de dos opciones que elegir se puede utilizar varios elseif.

```
<?php
    if ($bank_balance < 100) {
        $money = 1000;
        $bank_balance += $money;
    }
    elseif ($bank_balance > 200) {
        $savings += 100;
        $bank_balance -= 100;
    }
    else {
        $savings += 50;
        $bank_balance -= 50;
    }
?>
```

elseif

- Cuando se tienen más de dos opciones que elegir se puede utilizar varios elseif.

```
<?php
    if ($bank_balance < 100) {
        $money = 1000;
        $bank_balance += $money;
    }
    elseif ($bank_balance > 200) {
        $savings += 100;
        $bank_balance -= 100;
    }
    else {
        $savings += 50;
        $bank_balance -= 50;
    }
?>
```

- **Recomendación** finalizar siempre con un else.

elseif

- Cuando se tienen más de dos opciones que elegir se puede utilizar varios elseif.

```
<?php
    if ($bank_balance < 100) {
        $money = 1000;
        $bank_balance += $money;
    }
    elseif ($bank_balance > 200) {
        $savings += 100;
        $bank_balance -= 100;
    }
    else {
        $savings += 50;
        $bank_balance -= 50;
    }
?>
```

- **Recomendación** finalizar siempre con un else.
- Es mejor utilizar switch en lugar de muchos elseif

switch

- Utilizar siempre el default (mejor al final del switch).

switch

- Utilizar siempre el default (mejor al final del switch).
- Tener cuidado con los break. ¿Cuándo es útil omitir el break de un case?

El operador ternario (?)

```
<?php
    echo $fuel <= 1 ? "Fill tank now" : "There's enough fuel";
?>
```

```
<?php
    $enough = $fuel <= 1 ? FALSE : TRUE;
?>
```

¿Qué hace el siguiente código?

```
<?php
    $saved = $saved >= $new ? $saved : $new;
?>
```

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

while

```
<?php
    $fuel = 10;
    while ($fuel > 1) {
        // Keep driving ...
        echo "There's enough fuel";
    }
?>
```

```
<?php
    $count = 1;
    while ($count <= 12) {
        echo "$count times 12 is " . $count * 12 . "<br>";
        ++$count;
    }
?>
```


do ... while

El bloque de código es ejecutado **al menos** una vez.

```
<?php
    $count = 1;
    do
        echo "$count times 12 is " . $count * 12 . "<br>";
    while (++$count <= 12);
?>
```

for

```
<?php
    for ($count = 1 ; $count <= 12 ; ++$count)
        echo "$count times 12 is " . $count * 12 . "<br>";
?>

<?php
    for ($i = 1, $j = 1 ; $i + $j < 10 ; $i++ , $j++) {
        // ...
    }
?>
```

- for es el bucle de repetición más utilizado y más poderoso porque simplifica la expresión.

for

```
<?php
    for ($count = 1 ; $count <= 12 ; ++$count)
        echo "$count times 12 is " . $count * 12 . "<br>";
?>

<?php
    for ($i = 1, $j = 1 ; $i + $j < 10 ; $i++ , $j++) {
        // ...
    }
?>
```

- for es el bucle de repetición más utilizado y más poderoso porque simplifica la expresión.
- Es posible realizar múltiples operaciones en los parámetros (segundo ejemplo).

Saliendo de un bucle

- se puede utilizar `break` para salir de los bucles. Útil cuando se produce un error y el bucle no debería continuar.

```
<?php
/*
    Error al escribir en el archivo,
    por lo tanto no se sigue ejecutando
    el for
*/
$fp = fopen("text.txt", 'wb');
for ($j = 0 ; $j < 100 ; ++$j) {
    $written = fwrite($fp, "data");
    if ($written == FALSE) break;
}
fclose($fp);
?>
```

Saliendo de un bucle

- se puede utilizar `break` para salir de los bucles. Útil cuando se produce un error y el bucle no debería continuar.

```
<?php
/*
    Error al escribir en el archivo,
    por lo tanto no se sigue ejecutando
    el for
*/
$fp = fopen("text.txt", 'wb');
for ($j = 0 ; $j < 100 ; ++$j) {
    $written = fwrite($fp, "data");
    if ($written == FALSE) break;
}
fclose($fp);
?>
```

- Se puede romper varios niveles de anidamiento: `break 2`;

continue

- Similar al break pero en lugar de salir del bucle, le obliga a seguir con la siguiente iteración porque no tiene sentido seguir con la actual.

```
<?php
    $j = 10;
    while ($j > -10) {
        $j--;
        if ($j == 0) continue;
        echo (10 / $j) . "<br>";
    }
?>
```

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Casting explícito

Cast type	Description
(int) (integer)	Cast to an integer by dropping the decimal portion
(bool) (boolean)	Cast to a Boolean
(float) (double) (real)	Cast to a floating-point number
(string)	Cast to a string
(array)	Cast to an array
(object)	Cast to an object

```
<?php
    $a = 56;
    $b = 12;
    $c = $a / $b; // devuelve un punto flotante
    echo $c;

    $c = (int) ($a / $b); // entero. ojo parentesis
    echo $c;

?>
```


Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Enlaces dinámicos en PHP

- Se podría escribir una página web entera utilizando un solo archivo, pero no es recomendable por la gran cantidad de código que sería necesario en un solo archivo.

Enlaces dinámicos en PHP

- Se podría escribir una página web entera utilizando un solo archivo, pero no es recomendable por la gran cantidad de código que sería necesario en un solo archivo.
- Se recomienda separar el código en diferentes partes. Registro, inicio de sesión, página principal, mensajes, administración, etc. Cada parte en un archivo independiente.

Enlaces dinámicos en PHP

- Se podría escribir una página web entera utilizando un solo archivo, pero no es recomendable por la gran cantidad de código que sería necesario en un solo archivo.
- Se recomienda separar el código en diferentes partes. Registro, inicio de sesión, página principal, mensajes, administración, etc. Cada parte en un archivo independiente.
- Se puede comprobar esto al ver que la barra de direcciones llama a diferentes archivos para cada módulo o página.

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Presentación de diseño de BDD

Contenido I

- 1 Invitación a charla
- 2 Preguntas
- 3 Ejercicio
- 4 Expresiones
- 5 Operadores
 - Precedencia
 - Asociatividad
 - Operadores relacionales
 - Operadores de comparación
 - Operadores lógicos
- 6 Condicionales
- 7 Bucles
- 8 Casting explícito
- 9 Enlaces dinámicos en PHP
- 10 Revisión del deber
- 11 Ejercicio

Ejercicio

- Escribir código PHP que inicialice un arreglo con los números del 1 al 365. Determine a que mes pertenece el día del año. Y luego calcule cuántos días faltan para que se acabe el año con cada día del año, y muestre el mensaje “Falta poco para el año nuevo” solo si faltan menos de 5 días. Hoy estamos en el día 118 del año.