

# Programación en PHP

---

## 1. Generalidades.

PHP significa Personal Home Page, el cual fue concebido por Rasmus Lerdorf en 1994. PHP es un lenguaje que reúne muchas características de C++, JAVA y PERL.

PHP está basado en scripts escritos en lenguaje PERL aunque después fueron reescritos en C++.

PHP es un lenguaje interpretado y que se ejecuta en Servidor; es decir, los programas son ejecutados a través de un intérprete antes de transferir al cliente el resultado en forma de HTML puro.

## 2. Delimitadores:

El código escrito en PHP debe estar conformado por un bloque de código encerrado entre las etiquetas:

```
<?php
    Bloque de Código
?>
```

## 3. Variables:

- Las variables se declaran automáticamente, esto significa que no es necesario declarar formalmente una variable, de tal forma que se declara cuando se le asigna un valor.
- El nombre de una variable debe comenzar con el símbolo \$.
- PHP hace distinción entre los nombres de identificadores escritos con mayúsculas y minúsculas.

### 3.1. Tipos de Datos:

PHP soporta los siguientes tipos de datos:

#### 3.1.1. Enteros.

```
$Numero = 1234; // decimal
$Numero = 0123; // octal
$Numero = 0x12; // hexadecimal
```

#### 3.1.2. Reales.

```
$Numero = 1.23;
$Numero = 1.2e3;
```

### 3.2. Constantes:

Las constantes son declaradas empleando la función `define()`

```
define("CONSTANTE", "HOLA");
```

### 3.3. Operadores:

OPERADOR DE ASIGNACIÓN	
=	Asignación.
OPERADORES ARITMÉTICOS	
\$a + \$b;	Suma
\$a - \$b;	Resta
\$a++;	Post-Incremento, esta sentencia devuelve el valor de \$a y lo incrementa en 1.
++\$a;	Pre-Incremento, incrementa en 1 el valor de \$a y devuelve el valor incrementado.
\$a--;	Post-Decremento
--\$a;	Pre-Decremento
\$a * \$b;	Multiplicación
\$a / \$b;	División
\$a % \$b;	Módulo
OPERADORES DE COMPARACIÓN	
\$a == \$b;	Igual que
\$a === \$b;	Igualdad en valor y tipo de dato
\$a > \$b;	Mayor que
\$a < \$b;	Menor que
\$a >= \$b;	Mayor o igual
\$a <= \$b;	Menor o igual
\$a != \$b;	Distinto a
CONECTORES LÓGICOS	
\$a && \$b;	AND
\$a    \$b;	OR
!\$a;	NOT
OTROS	
@	Eliminación de error.

### 3.4. Estructuras de Control:

SELECCIÓN IF	
<code>if (expresión) sentencia;</code>	<code>if (expresión) {     sentencias; }</code>
<code>if (expresión){     sentencias; } else{     sentencias; }</code>	<code>if (expresión){     sentencias; } elseif (expresión){     sentencias; } else{     sentencias; }</code>

SELECCIÓN SWITCH
<code>switch (\$variable) {     case valor1:         sentencias;         [break;]     case valor2:         sentencias;         [break;]     case valor3:         sentencias;         [break;]      default:         sentencias; }</code>

REPETICIÓN FOR	REPETICIÓN WHILE
<code>for (expr1,expr2,expr3) {     sentencias; }</code>	<code>while (expresión) {     sentencias; }</code>
<code>foreach (\$vector as \$variable) {     sentencias; }</code>	<code>do {     sentencias; } while(expresión)</code>
<code>foreach (\$vector as \$clave =&gt; \$valor) {     sentencias; }</code>	

### 3.5. Cadena de Caracteres.

Una cadena de caracteres puede estar delimitada por comillas o apóstrofes. Cuando se utilizan comillas el intérprete de PHP previamente reemplaza los nombres de variables que contenga por el valor almacenado en la variable, pero cuando se utilizas apóstrofes el intérprete no reemplaza el valor de las variables.

```
$Var1 = 5;  
$Var2 = "Yo tengo $Var1 dedos";           // En $Var2 queda: "Yo tengo 5 dedos"  
$Var3 = 'Yo tengo $Var1 dedos';           // En $Var3 queda: "Yo tengo $Var1 dedos"
```

Cuando una cadena de caracteres es muy extensa, se puede usar la siguiente forma de asignación:

```
$Cadena=<<<EOD  
Este es un texto extenso que  
Ha sido asignado a la variable EOD;
```

FUNCIONES PARA EL MANEJO DE CADENAS DE CARACTERES.	
<b>strlen</b> (cadena)	Retorna el número de caracteres de la cadena.
<b>split</b> (separador,cadena)	Divide una cadena en varias usando un carácter separador.
<b>sprintf</b> (formato, var1, var2...)	Formatea una cadena al igual que printf, pero es devuelto como una cadena.
<b>substr</b> (cadena, inicio, longitud)	Devuelve una subcadena, empezando por inicio y de una longitud dada.
<b>chop</b> (cadena)	Elimina los saltos de línea y los espacios finales.
<b>strpos</b> (cadena1, cadena2)	Busca cadena2 dentro de cadena1 indicándo la posición en la que se encuentra.
<b>strrpos</b> (cadena,subcadena)	Última ocurrencia de una cadena en otra
<b>str_replace</b> (cadena1, cadena2, texto)	Reemplaza la cadena1 por la cadena2 en el texto.
<b>ltrim</b> (cadena)	Elimina los blancos que aparecen a la derecha de una cadena.
<b>rtrim</b> (cadena)	Elimina los blancos que aparecen por la derecha en una cadena.
<b>trim</b> (cadena)	Elimina los blancos que aparecen a izquierda y derecha de la cadena de caracteres
<b>str_pad</b> (cadena,longitud,relleno,lugar)	Comprueba si la longitud es menor que el valor indicado, si es así añade los caracteres necesarios. El lugar a añadir puede ser: str_pad_left añade por la derecha (opción por defecto) str_pad_right añade por la izquierda str_pad_both añade por ambos extremos.
<b>str_repeat</b> (caracter,numero_veces)	Repite un carácter el número de veces indicado
<b>strtolower</b> (cadena)	Pasa toda la cadena a letras minúsculas
<b>strtoupper</b> (cadena)	Pasa toda la cadena a letras mayúsculas
<b>ucfirst</b> (cadena)	Pasa a mayúscula el primer carácter de una cadena
<b>ucwords</b> (cadena)	Pone en mayúsculas el primer carácter de cada palabra
<b>str_replace</b> (subcadena1,subcad2,cadena)	Sustituye una palabra por otra dentro de una cadena
<b>strtr</b> (cadena,originales,traducidos)	Traduce ciertos caracteres. Ejemplo: \$persona=strtr(\$persona,"áéíóú","a,e,i,o,u"); de esta forma cambiaría todas las vocales con acento por vocales sin acento.
<b>substr_replace</b> (cadena,nueva,inicio,long)	Sustituye una porción del contenido de una cadena
<b>substr_count</b> (cadena,subcadena)	Frecuencia de aparición de una cadena
<b>strchr</b> (cadena,caracter)	Devuelve la subcadena que comienza en la primera aparición del carácter
<b>strstr</b> (cadena,subcadena)	Localiza subcadena dentro de la cadena original
<b>stristr</b> (cadena,subcadena)	Igual que la función anterior pero sin distinción entre mayúsculas y minúsculas
<b>ord</b> (cadena)	Devuelve el valor ASCII de un carácter
<b>strcmp</b> (cadena1,cadena2)	Compara dos cadenas siendo sensible a mayúsculas y minúsculas
<b>strcasecmp</b> (cadena1,cadena2)	Compara dos cadenas sin ser sensible a mayúsculas y minúsculas
<b>strncmp</b> (cadena1,cadena2,tamaño)	Compara los N primeros caracteres de una cadena
<b>strnatcmp</b> (cadena1,cadena2)	Sensible a mayúsculas y minúsculas. Compara dos cadenas.
<b>strnatcasecmp</b> (cadena1,cadena2)	No sensible a mayúsculas y minúsculas. Compara dos cadenas.
<b>chunk_split</b> (cadena,longitud,separador)	Coge una cadena e introduce separadores a una distancia determinada, sin modificar el original.

## 3.6. Arreglos.

Los elementos de un arreglo pueden ser referenciados de manera indexada por un número o asociativamente por una clave. Los arreglos se pueden declarar usando la función `array()` o inicializando cada elemento del arreglo en forma.

```
$Vector1          = array('Pedro', 'Juan', 'Diego');
$Vector2[0]        = 'Pedro';
$Vector2[1]        = 'Juan';
$Vector2[2]        = 'Diego';
$Vector3['Alumno1'] = 'Pedro';
$Vector3['Alumno2'] = 'Juan';
$Vector3['Alumno3'] = 'Diego';
$Vector2[]         = 'Carlos'; // Añade un elemento a Vector2.
$Vector4          = array('CODIGO' => 123, 'NOMBRE' => 'Lidia', 'EDAD' => 15);
```

En el caso de las matrices se puede considerar a estos como un arreglo de arreglos.

```
$Matriz1[0][0]      = 123; $Matriz1[0][1]      = 'OSCAR';
$Matriz1[1][0]      = 223; $Matriz1[1][1]      = 'LIDIA';
$Matriz1[2][0]      = 321; $Matriz1[2][1]      = 'MARIO';

$Matriz2['ALUMNO1']['CODIGO'] = 123; $Matriz2['ALUMNO1']['NOMBRE'] = 'OSCAR';
$Matriz2['ALUMNO2']['CODIGO'] = 223; $Matriz2['ALUMNO2']['NOMBRE'] = 'LIDIA';
$Matriz2['ALUMNO3']['CODIGO'] = 321; $Matriz2['ALUMNO3']['NOMBRE'] = 'MARIO';

$Matriz3 = array('ALUMNO1' => array('CODIGO' => 123, 'NOMBRE' => 'OSCAR'),
                 'ALUMNO2' => array('CODIGO' => 223, 'NOMBRE' => 'LIDIA'),
                 'ALUMNO3' => array('CODIGO' => 321, 'NOMBRE' => 'MARIO'));
```

### 3.6.1. Recorrido de un Arreglo.

Para recorrer una tabla (arreglo) se puede realizar emplear los índices de posicionamiento o las siguientes funciones:

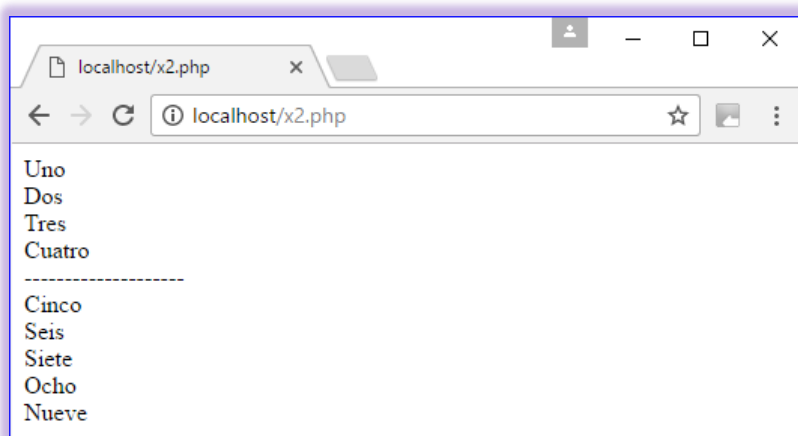
Funciones para el uso con arreglos asociativos.	
<code>current()</code>	Devuelve el valor del elemento señalado por el puntero interno del arreglo.
<code>pos()</code>	Realiza la misma función que <code>current()</code>
<code>reset()</code>	Mueve el puntero interno del arreglo al primer elemento.
<code>end()</code>	Mueve el puntero interno del arreglo al último elemento.
<code>next()</code>	Mueve el puntero interno del arreglo al siguiente elemento del actualmente activo.
<code>prev()</code>	Mueve el puntero interno del arreglo al anterior elemento del actualmente activo.
<code>count()</code>	Devuelve el número de elementos del arreglo.

### Ejemplo: Uso de arreglo asociativo.

```
<?php
$semana = array("lun","mar","mié","jue","vie","sáb","dom");
echo count($semana); // Despliega el valor 7.
reset($semana); // El puntero se ubica en el primer elemento.
echo current($semana); // Despliega el valor lun.
next($semana); // Avanza al elemento mar.
end($semana); // El puntero se ubica en el último elemento.
prev($semana); // Se ubica en sáb.
?>
```

```
01 <?php
02 $array[0][0] = "Uno";
03 $array[0][1] = "Dos";
04 $array[1][0] = "Tres";
05 $array[1][1] = "Cuatro";
06 for($i = 0; $i < count($array); $i++) {
07     for($j = 0; $j < count($array[$i]); $j++) {
08         echo $array[$i][$j].'\n';
09     }
10 }
11 echo "-----";
12 $array2[0][0][0] = "Cinco";
13 $array2[0][0][1] = "Seis";
14 $array2[0][0][2] = "Siete";
15 $array2[0][1][0] = "Ocho";
16 $array2[0][1][1] = "Nueve";
17 for($i = 0; $i < count($array2); $i++) {
18     for($j = 0; $j < count($array2[$i]); $j++) {
19         for($k = 0; $k < count($array2[$i][$j]); $k++) {
20             echo $array2[$i][$j][$k].'\n';
21         }
22     }
23 }
24 ?>
```

El resultado sería el siguiente:



## 3.6.2. Ejemplos

Las siguientes declaraciones de arreglos son equivalentes:

```
$arreglo = array(
    array('nombre' => 'Antonio', 'apellidos' => 'Gómez Gómez' , 'telefono' => '675832145'),
    array('nombre' => 'Pedro' , 'apellidos' => 'Guillén Gastón', 'telefono' => '674562178'),
    array('nombre' => 'Dolores', 'apellidos' => 'Candela Quema' , 'telefono' => '689765432'),
);

$arreglo[0]['nombre']      = 'Antonio';
$arreglo[0]['apellidos']   = 'Gómez Gómez';
$arreglo[0]['telefono']    = '675832145';
$arreglo[1]['nombre']     = 'Pedro';
$arreglo[1]['apellidos']   = 'Guillén Gastón';
$arreglo[1]['telefono']    = '674562178';
$arreglo[2]['nombre']     = 'Dolores';
```

## 4. Funciones Personalizadas:

### Sintaxis

```
function identificador([argumentos]) {
    bloque de código
    return [valor];
}
```

### Parámetros Valor y Variable

```
<?php

function suma1($x, $y){
    $x = $x + 1;
    $y = $y + 1;
    return $x + $y;
}

function suma2(&$x, $y){
    $x = $x + 1;
    $y = $y + 1;
    return $x + $y;
}

$a = 1;
$b = 2;

// parámetros por valor
echo suma1($a, $b); // Desplegará el valor 5
echo $a;            // Desplegará el valor 1
```

```

    echo $b;                // Desplegará el valor 2

//parámetros por referencia
    echo suma2($a, $b);    // Desplegará el valor 5
    echo $a;                // Desplegará el valor 2
    echo $b;                // Desplegará el valor 2

?>

```

Las variables pueden ser declaradas como estáticas o globales (static o global), lo que define el ámbito de acción. Las variables estáticas se definen dentro de una función, la primera vez que es llamada dicha función la variable se inicializa, guardando su valor para posteriores llamadas.

### Uso de variables estáticas

```

<?php
function contador() {
    static $count = 0;
    $count = $count + 1;
    return $count;
}
echo contador()."<BR>";    // Desplegará 1.
echo contador()."<BR>";    // Desplegará 2.
echo contador()."<BR>";    // Desplegará 3.

?>

```

Las variables globales, no se pueden declarar dentro de una función, lo que se hace es llamar a una variable que ya ha sido declarada fuera de ella, tomando el valor que tenga en ese momento.

### Uso de variables globales

```

<?php

$a = 1;

function ver_a() {
    global $a;
    echo $a."<BR>";    // Despliega el valor de $a.
    $a += 1;          // sumamos 1 a $a.
}

echo ver_a();          // Despliega 1.
echo ver_a();          // Despliega 2.
$a = 7;
echo ver_a();          // Despliega 7.
echo ver_a();          // Despliega 8.

?>

```



Por lo general, todos los script tienen partes de código iguales, las funciones **include()** y **require()** hacen una llamada a un determinado archivo de dos maneras diferentes, con **include()**, se inserta lo que contenga el fichero; en cambio **require()**, le decimos que el script necesitará parte de código que se encuentra en el fichero que llama **require()**.

#### Uso de include()

```
<?php
    include ("header.inc");
    echo "Programación en PHP";
    include ("footer.inc");
?>
```

HEADER.INC:

```
<html>
<body>
```

FOOTER.INC:

```
<html>
<body>
```

#### Uso de require()

```
<?php
    require ("config.inc");
    include ("header.inc");
    echo $cadena;
    include ("footer.inc");
?>
```

CONFIG.INC:

```
<?php
    $cadena = "Programación en PHP";
?>
```

HEADER.INC:

```
<html>
<body>
```

FOOTER.INC:

```
<html>
<body>
```

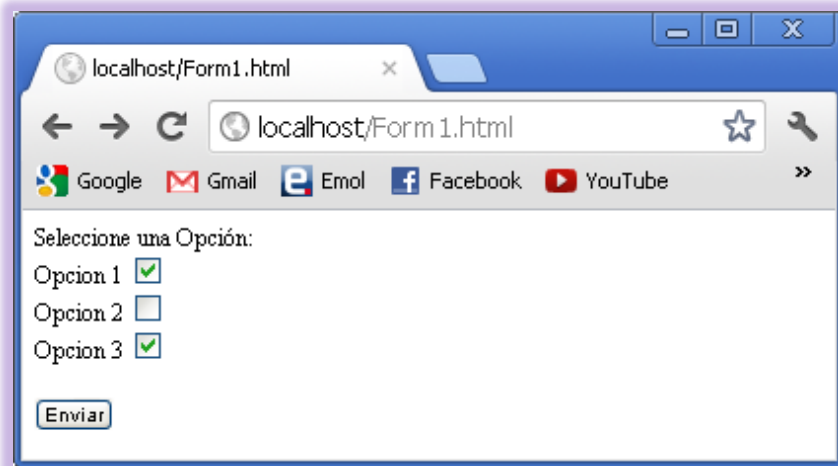
## 5. Formularios.

Un formulario HTML es una sección de un documento que contiene contenido normal, código, elementos especiales llamados controles (casillas de verificación (checkboxes), radiobotones (radio buttons), menús, etc.), y rótulos (labels) en esos controles. Los usuarios normalmente "completan" un formulario modificando sus controles (introduciendo texto, seleccionando objetos de un menú, etc.), antes de enviar el formulario a un agente para que lo procese (p.ej., a un servidor web, a un servidor de correo, etc.)

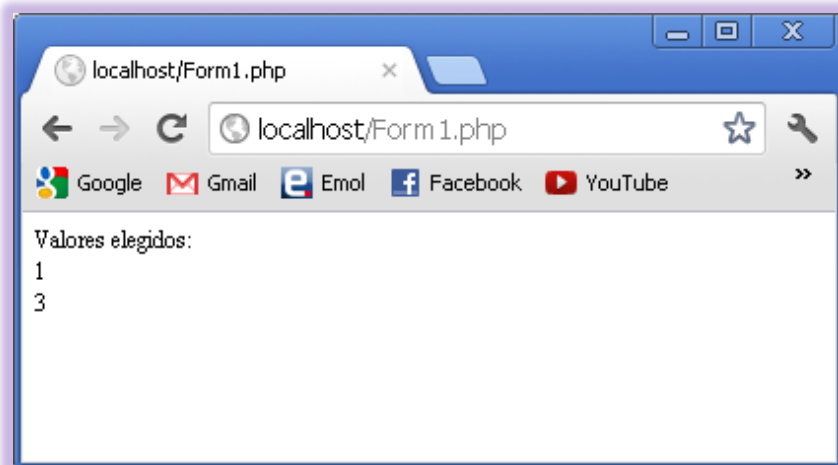
Un formulario es un bloque de código delimitado por los tags <FORM> y </FORM>. Dentro de los atributos de este tag más relevantes para la ejecución de un programa en PHP son los siguientes: METHOD y ACTION. El primero establece la forma de enviar valores al programa PHP y el segundo el programa PHP que será invocado cuando se presione el botón SUBMIT.

### 5.1. EJEMPLO 1.

Realizar una página en donde se puedan desplegar 3 casillas de verificación (CheckBox) como se aprecia en la siguiente figura:



Al presionar el botón **Enviar**, se deberá desplegar las opciones que fueron elegidas. Si consideramos las casillas de verificación seleccionadas en la imagen anterior, el resultado a desplegar sería el siguiente:



El código que permite esto está contenido en 2 archivos: Form1.HTML y Form1.PHP

### Form1.HTML

```
01 <HTML>
02   <HEAD>
03   </HEAD>
04   <BODY>
05     <FORM METHOD=POST ACTION="Form1.php">
06       Seleccione una Opción: <br>
07       Opcion 1 <INPUT NAME = "check1" TYPE = "Checkbox" VALUE = 1> <BR>
08       Opcion 2 <INPUT NAME = "check2" TYPE = "Checkbox" VALUE = 2> <BR>
09       Opcion 3 <INPUT NAME = "check3" TYPE = "Checkbox" VALUE = 3> <BR>
10       <BR>
11       <INPUT TYPE=SUBMIT>
12     </FORM>
13   </BODY>
14 </HTML>
```

### Form1.PHP

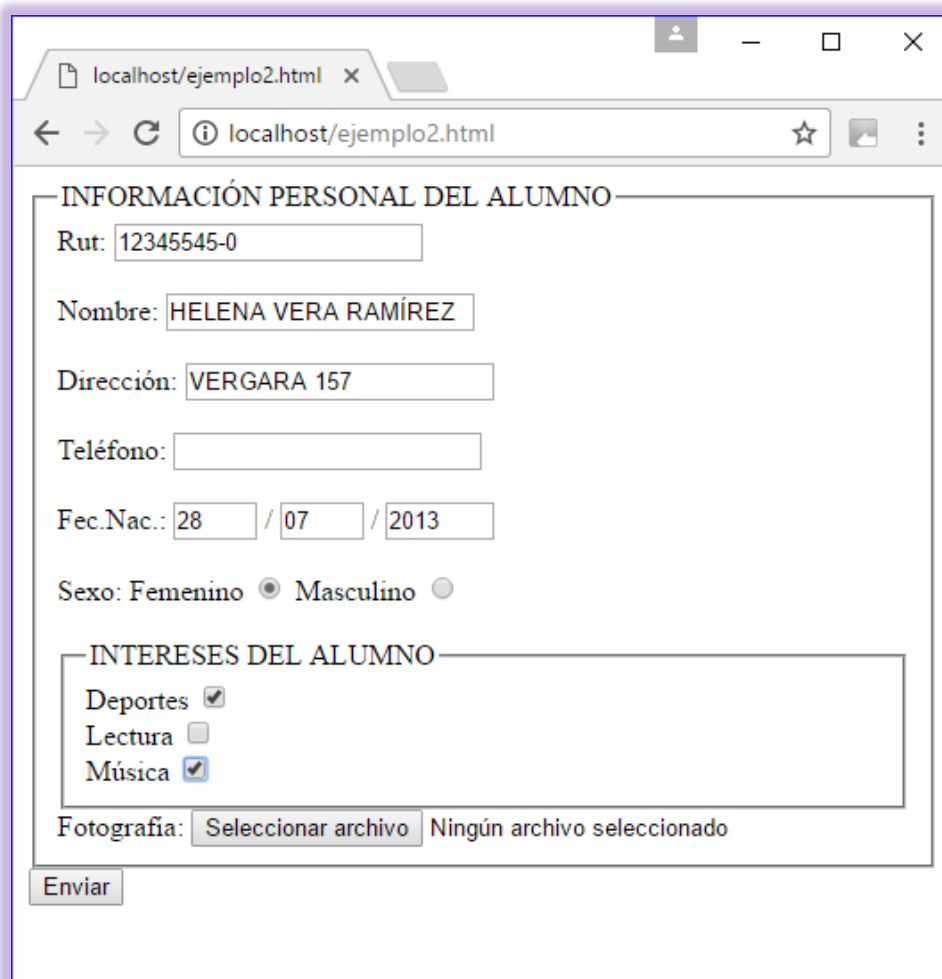
```
01 <?php
02   echo "Valores elegidos: <br>";
03   if(isset($_POST['check1'])) echo $_POST['check1']."<br>";
04   if(isset($_POST['check2'])) echo $_POST['check2']."<br>";
05   if(isset($_POST['check3'])) echo $_POST['check3']."<br>";
06   ?>
```

Como se puede observar, en la 04 del archivo Form1.HTML se establece el método de paso de valores (METHOD=POST) y el archivo (Form1.PHP) que recibe los valores cuando se presiona el botón Enviar.

En el programa Form1.PHP, la función `isset()` permite determinar si las variables si una o más variables están definidas y no es NULL, devolviendo true o false.

## 5.2. EJEMPLO 2.

El siguiente ejemplo presenta un formulario de aspecto básico pero que solicita el ingreso de datos, empleando para ello diversos controles. El programa EJEMPLO2.PHP procesará los datos recibidos desde el formulario EJEMPLO2.HTML



The screenshot shows a web browser window with the address bar displaying 'localhost/ejemplo2.html'. The form is titled 'INFORMACIÓN PERSONAL DEL ALUMNO' and contains the following fields:

- Rut: 12345545-0
- Nombre: HELENA VERA RAMÍREZ
- Dirección: VERGARA 157
- Teléfono: (empty)
- Fec.Nac.: 28 / 07 / 2013
- Sexo: Femenino (selected) / Masculino

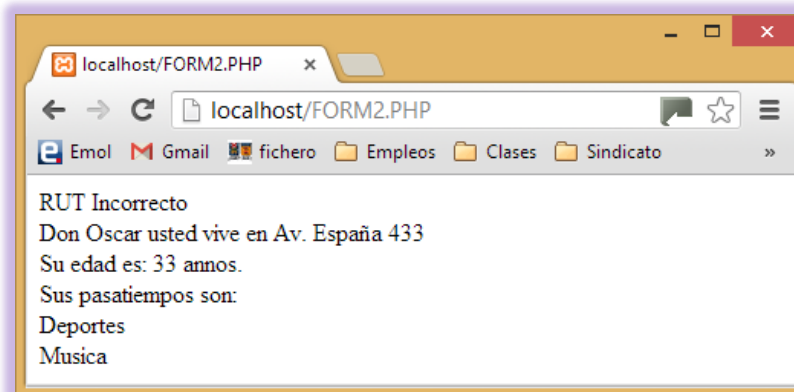
Below this section is another section titled 'INTERESES DEL ALUMNO' with the following options:

- Deportes ☒
- Lectura ☐
- Música ☒

At the bottom of the form is a 'Fotografía:' field with a 'Seleccionar archivo' button and the text 'Ningún archivo seleccionado'. An 'Enviar' button is located at the bottom left of the form.

EJEMPLO2.HTML

Y el resultado a obtener sería el siguiente:



The screenshot shows a web browser window with the address bar displaying 'localhost/FORM2.PHP'. The output of the script is displayed in a text area:

RUT Incorrecto  
Don Oscar usted vive en Av. España 433  
Su edad es: 33 años.  
Sus pasatiempos son:  
Deportes  
Musica

EJEMPLO2.PHP

El código HTML del archivo Ejemplo2.HTML es el siguiente:

## EJEMPLO2.HTML

```
01 <HTML>
02 <HEAD>
03   <META CHARSET = "UTF-8">
04 </HEAD>
05 <BODY>
06   <FORM METHOD = "POST" ACTION = "EJEMPLO2.PHP">
07     <FIELDSET>
08       <LEGEND>INFORMACIÓN PERSONAL DEL ALUMNO</LEGEND>
09       Rut:      <INPUT TYPE = "TEXT" NAME = "TXT_RUT"      > <BR><BR>
10       Nombre:  <INPUT TYPE = "TEXT" NAME = "TXT_NOMBRE"  > <BR><BR>
11       Dirección:<INPUT TYPE = "TEXT" NAME = "TXT_DIRECCION"> <BR><BR>
12       Teléfono:<INPUT TYPE = "TEXT" NAME = "TXT_TELEFONO" > <BR><BR>
13       Fec.Nac.: <INPUT TYPE = "TEXT" NAME = "TXT_DIA"  SIZE = 2 MAXLENGTH = 2> /
14                <INPUT TYPE = "TEXT" NAME = "TXT_MES"  SIZE = 2 MAXLENGTH = 2> /
15                <INPUT TYPE = "TEXT" NAME = "TXT_ANO"  SIZE = 4 MAXLENGTH = 4>
16       <BR> <BR>
17       Sexo: Femenino <INPUT TYPE = "RADIO" NAME = "OPT_SEX0" VALUE = "F">
18            Masculino <INPUT TYPE = "RADIO" NAME = "OPT_SEX0" VALUE = "M">
19       <BR><BR>
20       <FIELDSET>
21         <LEGEND>INTERESES DEL ALUMNO</LEGEND>
22         Deportes <INPUT TYPE = "CHECKBOX" NAME = "CHK_DEPORTES"> <BR>
23         Lectura  <INPUT TYPE = "CHECKBOX" NAME = "CHK_LECTURA" > <BR>
24         Música   <INPUT TYPE = "CHECKBOX" NAME = "CHK_MUSICA"  > <BR>
25       </FIELDSET>
26       Fotografía: <INPUT TYPE = "FILE" NAME "IMG_FOTO" SIZE = "10"
27                  ACCEPT = "IMAGE/PNG"> <BR>
28     </FIELDSET>
29     <INPUT TYPE = "SUBMIT" />
30   </FORM>
31 </BODY>
32 </HTML>
```

El código PHP de Form2.PHP es el siguiente:

C:\xampp\htdocs\form2.php - Notepad++

Archivo Editar Buscar Vista Codificación Lenguaje Configuración Macro Ejecutar Plugins Ventana ?

ejemplo1.html x form1.php x ejemplo2.html x form2.php x

```
1 <?php
2 $RutNumero = substr($_POST['Rut'],0,strlen($_POST['Rut'])-2);
3 $RutDigito = substr($_POST['Rut'],strlen($_POST['Rut'])-1,1);
4
5 if(DV($RutNumero) != $RutDigito) echo "RUT Incorrecto <BR>";
6
7 if ($_POST['Sexo'] == "F")
8     {echo "Dona ";}
9 else
10     {echo "Don ";}
11
12 $Edad = (2013 - $_POST['Anno']);
13
14 echo $_POST['Nombre']. " usted vive en ".$_POST['Direccion']. "<BR>";
15 echo "Su edad es: ".$Edad. " annos.<BR>";
16 echo "Sus pasatiempos son: <BR>";
17 if(isset($_POST['Deportes']) && $_POST['Deportes'] == "on") {echo "Deportes<BR>";}
18 if(isset($_POST['Lectura']) && $_POST['Lectura'] == "on") {echo "Lectura<BR>";}
19 if(isset($_POST['Musica']) && $_POST['Musica'] == "on") {echo "Musica<BR>";}
20 if(isset($_POST['Foto'])) echo "Su fotografia se encuentra en el archivo: ".$_POST['Foto'];
21
22 /*****/
23 function DV($RUT){
24     $SUMA = 0;
25     $FACTOR = 2;
26     $POSICION = strlen($RUT);
27     while ($POSICION != 0) {
28         $DIGITO = substr($RUT, $POSICION, 1);
29         $SUMA = $SUMA + $FACTOR * $DIGITO;
30         $FACTOR = $FACTOR + 1;
31         $POSICION = $POSICION - 1;
32
33         if ($FACTOR == 9) $FACTOR = 2;
34     }
35     $DIFERENCIA = 11 - $SUMA % 11;
36     switch ($DIFERENCIA) {
37         case 11:
38             return "0";
39             break;
40         case 10:
41             return "K";
42             break;
43         default:
44             return $DIFERENCIA;
45     }
46 }
47 ?>
```

Aquí va un ==

PHP Hypertext Prepr length : 1463 lines : 47 Ln : 47 Col : 1 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS

## 6. Manejo de Bases de Datos.

### 6.1. Establecer conexión a un Servidor de Datos.

Para establecer conexión a un servidor de Datos, se emplea la función *mysqli\_connect()* la que presenta el siguiente formato:

Conexión a un Servidor de Datos
---------------------------------

<pre>\$Conexion = mysqli_connect(hostname, usuario, password, database);</pre>
--

Si la conexión es correcta, la función retorna un valor que servirá para hacer referencia a dicha conexión, la cual será empleada en otras operaciones. Si la conexión no resulta exitosa, la función devuelve un valor falso.

### 6.2. Cerrar una conexión abierta.

Cierre de una Conexión
------------------------

<pre>mysqli_close(\$Conexion);</pre>
--------------------------------------

### 6.3. Ejecución de sentencias SQL.

Mediante el uso de la función *mysqli\_query()* es posible ejecutar sentencias SQL sobre la base de datos señalada.

Query
-------

<pre>\$Resultado = mysqli_query(\$Conexión, \$SQL_Query);</pre>
---

#### 6.3.1. Cantidad de Filas Consultadas o Modificadas.

Obtención de cantidad de Filas procesadas o afectadas.
--

<pre>\$Cantidad = mysqli_affected_rows(\$Conexión);</pre>
---

## 6.4. Ejemplo.

Los siguientes ejemplos, realizan las operaciones SELECT, INSERT, UPDATE y DELETE sobre la Base de Datos COLEGIO.

La base de datos COLEGIO, contiene una tabla llamada ALUMNOS, cuya estructura de registro es la siguiente:

ALUMNOS		
CAMPO	TIPO	DESCRIPCIÓN
CODIGO	VARCHAR(2)	Código de identificación del alumno (PK)
NOMBRE	VARCHAR(20)	Nombre del alumno.

### SELECT SQL

```
01 <?php
02 // CONSULTA A LA TABLA ALUMNOS.
03 $Conexion = mysqli_connect("localhost","root","","COLEGIO");
04
05 if(!$Conexion){
06     echo "Error MySQL: " . mysqli_error($Conexion);
07     exit;
08 }
09
10 $Sql = "SELECT Codigo, Nombre FROM Alumnos";
11 $Resultado = mysqli_query($Conexion, $Sql);
12 if (!$Resultado) {
13     echo "Error MySQL: " . mysqli_error($Conexion);
14 }
15 else{
16     if (mysqli_affected_rows($Conexion) == 0){
17         echo "Tabla Vacía";
18     }
19     else{
20         while ($Fila = mysqli_fetch_assoc($Resultado)) {
21             echo $Fila['Codigo'] . " " . $Fila['Nombre'] . "<br>";
22         }
23     }
24     mysqli_free_result($Resultado);
25 }
26 mysqli_close($Conexion);
27 ?>
```



## INSERT SQL

```
01 <?php
02     // INSERTAR REGISTRO.
03     $Conexion = mysqli_connect("localhost","root","","datos");
04     if(!$Conexion){
05         echo "Error MySQL: " . mysqli_error($Conexion);
06         exit;
07     }
08
09     $Sql = "INSERT INTO Alumnos(Codigo, Nombre) VALUE (4, 'ALUMNO 4')";
10     $Resultado = mysqli_query($Conexion, $Sql);
11     if (!$Resultado) {
12         echo "Error MySQL: " . mysqli_error($Conexion);
13     }
14     else{
15         if (mysqli_affected_rows($Conexion) <> 0){
16             echo "Inserción Exitosa";
17         }
18         else {
19             echo "Error, No ha sido posible insertar el registro";
20         }
21     }
22
23     mysqli_close($Conexion);
24 ?>
```

## UPDATE SQL

```
01 <?php
02     // MODIFICAR REGISTRO.
03     $Conexion = mysqli_connect("localhost","root","","datos");
04     if(!$Conexion){
05         echo "Error MySQL: " . mysqli_error($Conexion);
06         exit;
07     }
08
09     $Sql = "UPDATE Ejemplo SET Nombre = 'NUEVO NOMBRE' WHERE Codigo = 4";
10     $Resultado = mysqli_query($Conexion, $Sql);
11     if (!$Resultado) {
12         echo "Error MySQL: " . mysqli_error($Conexion);
13     }
14     else{
15         if (mysqli_affected_rows($Conexion) <> 0){
16             echo "Modificación Exitosa";
17         }
18         else {
19             echo "Error, No ha sido posible modificar el registro";
20         }
21     }
22
23     mysqli_close($Conexion);
24 ?>
```

## DELETE SQL

```
01 <?php
02 // ELIMINAR REGISTRO.
03 $Conexion = mysqli_connect("localhost","root","","datos");
04 if(!$Conexion){
05     echo "Error MySQL: " . mysqli_error($Conexion);
06     exit;
07 }
08
09 $Sql = "DELETE FROM Alumnos WHERE Cosigo = '4'";
10 $Resultado = mysqli_query($Conexion, $Sql);
11 if (!$Resultado) {
12     echo "Error MySQL: " . mysqli_error($Conexion);
13 }
14 else{
15     if (mysqli_affected_rows($Conexion) <> 0){
16         echo "Eliminación Exitosa";
17     }
18     else {
19         echo "Error, No ha sido posible eliminar el registro";
20     }
21 }
22
23 mysqli_close($Conexion);
24 ?>
```