

# Programación de aplicaciones Web con PHP

Lic. Jorge Arroyo Hernández

[jarroy@una.ac.cr](mailto:jarroy@una.ac.cr)

MSc. Federico Mora Mora

[fmoram@una.ac.cr](mailto:fmoram@una.ac.cr)

Universidad Nacional, Costa Rica

## Resumen

PHP es uno de los lenguajes de scripting más utilizados para la creación de páginas Web dinámicas debido a su enorme potencial, facilidad de aprendizaje y excelente rendimiento. Esta ponencia hace un recorrido sobre los principios básicos de PHP. Se abordan los conceptos básicos de programación tales como tipos de variables, operadores, estructuras condicionales y de ciclos, funciones y la orientación a objetos.

## Palabras claves

PHP, lenguaje de programación web, scripting, programación orientada a objetos.

## 1. Introducción

En los últimos años, una de las causas de la creciente popularidad de la Internet es aumento de sitios Web dinámicos y de sus múltiples funcionalidades. Las nuevas características y flexibilidad de las poderosas herramientas han dado pie para la implementación de éstos sitios electrónicos que han venido a mejorar y a digitalizar nuevos y mejores servicios e información.

Muchas tecnologías están disponibles para la creación de páginas Web dinámicas, entre los que se pueden citar ASP (Active Server Pages), JSP (Java Server Pages), ColdFusion y PHP (Hipertext Pre-Procesor).

Sin embargo, por sus múltiples características, fácil uso, ser de código abierto y multiplataforma, PHP ha es el lenguaje más popular para la programación de sitios Web dinámicos.

El propósito principal de este documento es realizar una descripción de los principios básicos del lenguaje PHP orientado a la creación de sitios Web dinámicos. Además, describir aspectos más importantes de PHP para programar en este lenguaje e implementar sitios Web dinámicos.

## **2. Generalidades sobre PHP**

El sistema PHP fue diseñado en 1994, por Rasmus Lerdorf en un CGI en el lenguaje C que permitió la interpretación de algunos comandos básicos. El sistema fue denominado Personal Home Page Tools y creció en popularidad rápidamente al permitir ser utilizado para otras aplicaciones Web. Los israelíes del Technion, Zeev Suraski y Andi Gutmans, reescribieron el analizador sintáctico en el año 1997 y crearon la base del PHP3, cambiando el nombre del lenguaje a la forma actual. Inmediatamente comenzaron experimentaciones públicas de PHP3 y fue publicado oficialmente en junio del 1998.

Para 1999, Suraski y Gutmans reescribieron el código de PHP, produciendo lo que hoy se conoce como Zend Engine o motor Zend, un portmanteau de los nombres de ambos, Zeev y Andi. También fundaron Zend Technologies en Ramat Gan, Israel.

En mayo de 2000, PHP 4 fue lanzado bajo el poder del motor Zend Engine 1.0. El día 13 de julio de 2007 se anunció la suspensión del soporte y desarrollo de la versión 4 de PHP a pesar de lo anunciado se ha liberado una nueva versión con mejoras de seguridad, la 4.4.8 publicada el 13 de Enero del 2008. Según esta noticia se dará soporte a fallos críticos hasta el 2008-08-08.

El 13 de julio de 2004, fue lanzado PHP 5, utilizando el motor Zend Engine II (o Zend Engine 2). La versión más reciente de PHP es la 5.2.5 (8 de noviembre de 2007), que incluye todas las ventajas que provee el nuevo Zend Engine. Está previsto el lanzamiento en breve de la rama 6 de PHP.

¿Qué se necesita para que PHP funcione?

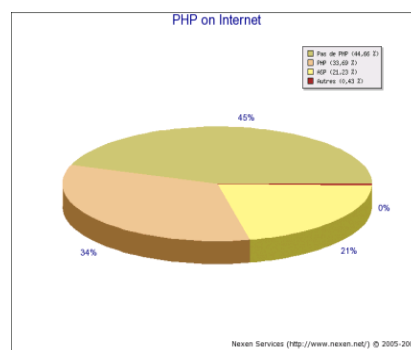
PHP es un lenguaje de programación Web diseñado para que un intérprete que lo analice y ejecute. El servidor web, cuando detecta una extensión asociada a PHP, envía esta página al intérprete y éste se la devuelve la respuesta según la solicitud. Para que PHP funcione es necesario que se versión compilada de un procesador de PHP y un servidor Web. Paralelamente se puede trabajar con una base de datos si es necesario.

En este documento la infraestructura para el desarrollo de ejemplos, la proporciona WampServer<sup>1</sup>. Éste es un sistema para Internet en un entorno integrado y configurado listo para su uso, utiliza las siguientes herramientas: Windows, el sistema operativo, Apache, el servidor web, MySQL, el gestor de bases de datos, Perl, PHP, o Python, lenguajes de programación, y PhpMyAdmin.

Para digitar el código en PHP no hace falta un editor de texto muy sofisticado, puede ser en el bloc de notas. Hay editores de para PHP libres en Internet que ayudan a digitar el código PHP y HTML.

La siguiente grafica muestra las el uso de PHP en agosto del 2007 alrededor del mundo. Se observa que un 34% de las aplicaciones Web están programadas con PHP.

Figura 1. La gráfica uso de PHP por encima de otros lenguajes de programación<sup>2</sup>.



PHP es un lenguaje de "código abierto" interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor. Se caracteriza por la amplia gama de posibilidades que brinda, es soportada por la mayoría de los servidores, integración con múltiples motores de bases de datos, portabilidad, administración de datos e integración con muchos paquetes especializados.

#### Ventajas de PHP

- Es un lenguaje multiplataforma.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL

<sup>1</sup> Para obtener la documentación y el archivo ejecutable de WampServer diríjase a <http://www.wampserver.com/>

<sup>2</sup> Tomado de [http://www.nexen.net/chiffres\\_cles/phpversion/17504-php\\_statistics\\_for\\_august\\_2007.php#sglobal](http://www.nexen.net/chiffres_cles/phpversion/17504-php_statistics_for_august_2007.php#sglobal)

- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su página oficial
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite las técnicas de Programación Orientada a Objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables.
- Tiene manejo de excepciones.

#### Desventajas de PHP

- No posee una abstracción de base de datos estándar, sino bibliotecas especializadas para cada motor (a veces más de una para el mismo motor).
- Por su diseño dinámico no puede ser compilado y es muy difícil de optimizar.
- Por sus características promueve la creación de código desordenado y complejo de mantener.

Debido al interés creciente de sobre este lenguaje de programación, este trabajo nace con el propósito principal de realizar una descripción de los principios básicos del lenguaje PHP orientado a la creación de sitios Web dinámicos.

En este documento, se hace referencia a aspectos y conceptos básicos como la sintaxis, tipos de variables, estructura de control, funciones estándares en PHP, uso de array, aspectos de la programación de objetos (POO) en PHP como clases y declaración de métodos, atributos y métodos de clase, herencia, clases abstractas, Interfaces y polimorfismos.

### **3. Principios básicos para el lenguaje PHP**

PHP es un lenguaje de programación Web diseñado para que un intérprete lo analice y ejecute. El servidor web, cuando detecta una extensión asociada a PHP, envía esta página al intérprete y éste se la devuelve la respuesta según la solicitud. Para que PHP funcione es necesario que se versión compilada de un procesador de PHP y un servidor Web. Paralelamente se puede trabajar con una base de datos si es necesario.

Éste lenguaje que está muy relacionado con el HTML, por lo que es muy común que este insertado dentro de un documento de este tipo. Debido a esto, la forma más común para que el intérprete de PHP pueda reconocer el código es delimitando éste con las etiquetas `<?PHP y ?>`. Existen otras formas de hacerlo,

sin embargo para efectos de este documento se utilizará ésta, pues es la más utilizada y no necesita ninguna configuración por defecto.

Figura 2. Sintaxis básica de HTML y PHP

```
<html>
<head>
<title> Página de
prueba</title>
</head>
<?php
Instrucciones
?>
</html>
```

Los archivos de este software utilizan la extensión php<sup>3</sup>. A continuación se presenta un ejemplo del código fuente de un documento.

Figura 3. Script: Mi primer página de con PHP

```
<html>
<head>
<title> Página de prueba</title>
</head>
¡Hola mundoj! <br><br>
<?php

// obtener la fecha en el formato día mes y año
$fecha = date("d.m.y");

// obtener la hora en el formato hora minuto y segundo
$hora = date("h.i.s");

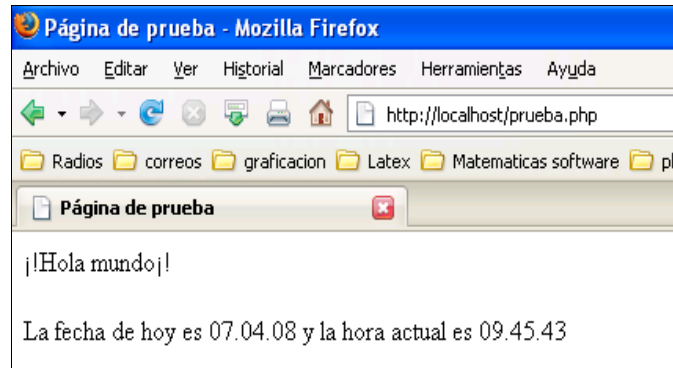
// despliega entonces
echo "La fecha de hoy es ".$fecha. " y la hora actual es ". $hora;
?>
</html>
```

Si se guarda el archivo anterior con el nombre prueba.php y se digita en la barra de direcciones del browser <http://localhost/prueba.php>, se despliega en la pantalla el siguiente página.

---

<sup>3</sup> Recuerde que los archivos PHP son ejecutados en WAMP. Para mayor información puede consultar a la página <http://www.wampserver.com/>

Figura 4. Ejemplo de una página con PHP



En detalle línea por línea, la figura 3 presenta en dos funciones para obtener la fecha y la hora. Para documentar el código, se comenta de varias maneras siguiendo el mismo estilo de java:

Figura 5. Formas de comentar código en PHP

Función	Código
Comenta sólo una línea	// texto
Comenta el texto en una o varias líneas.	/* texto a comentar */

Para declarar cualquier variable no hace falta definirle el tipo, sin embargo hay que anteceder al nombre de las variables el signo de dólar. Para desplegar las líneas de código se puede hacer con el comando *echo* y se utiliza el punto (.) para concatenar las líneas de código estático y código dinámico tal y como se muestra en la siguiente línea de código:

```
echo "La fecha de hoy es ".$fecha. " y la hora actual es ". $hora;
```

#### 4. Tipo de datos y su manipulación en PHP<sup>4</sup>

El lenguaje de PHP no requiere la definición explícita de tipos en la declaración de variables pues lo determina el contexto en el que es usada. Así se genera una conversión automática de un tipo a otro dependiendo del valor asignado.

Para ilustrarla, un ejemplo interesante es la conversión que hace el operador “+” el cual cambia la forma de evaluación. Si cualquiera de los dos sumandos es flotante el resultado también lo va a hacer, en cualquier otro caso los operandos van a ser

<sup>4</sup> Tomado de <http://www.php.net/manual/es/language.types.php>

tomado como enteros tal. PHP soporta ocho tipos de datos primitivos. En detalle son:

**Booleanos:** Un boolean expresa un valor de verdad. Puede ser TRUE or FALSE.

**Enteros:** Un integer es un número del conjunto de los números enteros.

**Números de punto flotante:** Los números de punto flotante representan números reales.

**String:** Un valor string es una serie de caracteres.

**Matrices:** Una matriz en PHP es en realidad un mapa ordenado. Un mapa es un tipo de datos que asocia valores con claves. Este tipo es optimizado en varias formas, de modo que puede usarlo como una matriz real, o una lista (vector), tabla asociativa (caso particular de implementación de un mapa), diccionario, colección, pila, cola y probablemente más.

**Objetos:** Tipos que variable que sirve para instanciar clases.

**Recurso:** Un recurso es una variable especial, que contiene una referencia a un recurso externo. Los recursos son creados y usados por funciones especiales.

**Null:** El valor especial null representa que una variable no tiene valor.

## Constantes

Al igual que las funciones de validación para variables, existen las respectivas la definición de constantes. Sólo son consideradas para los casos en los que no presente problemas de seguridad tales como en password, direcciones a archivos o algunas expresiones matemáticas. Se distinguen en dos formas. La primera, sólo se le puede asignar un valor durante todo el programa; la segunda distinción, no inicia con el carácter \$. En el siguiente script se describe como se define una constante.

Figura 6. Script: Definición de constante en PHP.

```
<html>
<head> <title> Página de prueba</title> </head>
<?php

    define("PI", 3.1415); // se define la constante PI como una
    aproximación
?>
```

```
</html>
```

### Expresiones

PHP basa su sintaxis en expresiones. Las expresiones son objetos que se le asigna o produce un valor. O en otra forma, se puede afirmar que todo lo que se le asigna o produce un valor es una expresión. Las expresiones pueden ser desde variables, constantes, operadores hasta funciones.

### 5. Operadores

Los operadores se utilizan para determinar o para obtener un valor a partir de una solicitud, operación matemática o para determinar la veracidad de una expresión. Entre esto encontramos los operadores aritméticos, los operadores de asignación, los operadores de cadenas, los operadores de incremento o decremento, los operadores de comparación, los operadores a nivel de bit, los operadores de ejecución y los operadores para la omisión de mensajes de error. Los operadores de asignación se utilizan para asignarle valor a las variables en un script.

Figura 7. Operadores de asignación PHP

Operador	Ejemplo	Descripción
=	\$a = \$b	\$a toma el valor de \$b
+=	\$a += \$b	\$a aumenta el valor en \$b
-=	\$a -= \$b	\$a disminuye en \$b
*=	\$a *= \$b	\$a se aumenta en \$b veces
/=	\$a /= \$b	\$a se divide en \$b veces
%=	\$a %= \$b	\$a es el residuo del cociente con \$b
.=	\$a .= \$b	\$a se concatena con \$b

Los operadores aritméticos funcionan de la misma manera que la aritmética convencional tanto para constantes y variables.



Figura 8. Operadores aritméticos PHP

Operador	Ejemplo	Descripción
+	$\$a + \$b$	Suma de dos operandos
-	$\$a - \$b$	Resta de dos operandos
*	$\$a * \$b$	Producto de dos operandos
/	$\$a / \$b$	División de dos operandos
%	$\$a \% \$b$	Residuo de dos operandos

Los operadores de cadena sirven se utilizan para concatenar los operados.

Figura 9. Operadores de asignación PHP

Operador	Ejemplo	Descripción
.	"hola " . " mundo"	se concatena ambas expresiones
.=	$\$a .= \$b$	$\$a$ se concatena con $\$b$

PHP soporta los operandos de Incremento y decremento, además la ejecución va a depender de la posición en que se ubique la variable con respecto al operador que es aplicado.

Figura 10. Operadores de decremento e incremento en PHP

Operador	Ejemplo	Descripción
++	++ $\$a$	<b>Preincremento:</b> Incrementa a $\$a$ en 1 y después devuelve en $\$a$
	$\$a$ ++	<b>Postincremento:</b> devuelve en $\$a$ y después Incrementa a $\$a$ en 1
--	-- $\$a$	<b>Predecremento:</b> decrementa a $\$a$ en 1 y después devuelve en $\$a$
	$\$a$ --	<b>Postdecremento:</b> devuelve en $\$a$ y después decrementa a $\$a$ en 1

Los operadores lógicos, también denominados operadores booleanos se utilizan para evaluar las expresiones. Devuelven falso o verdadero.

Figura 11. Operadores de decremento e incremento en PHP

Operador	Ejemplo	Descripción
And &&	\$a and \$b	Devuelve verdadero cuando \$a y \$b son verdaderas
Or 	\$a or \$b	Devuelve verdadero cuando \$a o \$b es verdadera
Xor	\$a xor \$b	Devuelve verdadero cuando \$a ó \$b es verdadera. Si ambos son verdaderos devuelve falso.
!	!\$a	Devuelve el valor de verdad de la expresión negado. Falso si \$a es verdadero, o verdadero si \$a es falso.

Los operadores de comparación se utilizan para confrontar expresiones sobre dos o más operandos. Usualmente se utilizan para preguntar y devuelven falso o verdadero según corresponda.

Figura 12. Operadores de comparación en PHP

Operador	Ejemplo	Descripción
==	\$a == \$b	Devuelve verdadero si los operandos son iguales
!=	\$a != \$b	Devuelve verdadero si los operandos son diferentes
<	\$a < \$b	Devuelve verdadero si \$a es menor que \$b
<=	\$a <= \$b	Devuelve verdadero si \$a es menor o igual que \$b
>	\$a > \$b	Devuelve verdadero si \$a es mayor que \$b
>=	\$a >= \$b	Devuelve verdadero si \$a es mayor o igual que \$b

## 6. Estructuras de programación en PHP

Cualquier lenguaje de programación posee instrucciones que se ejecutan una detrás de otras en el orden que han sido pensadas. PHP posee tres estructuras de control bien definidas. If - else, if - else y la instrucción switch que sirven para ejecutar o no una acción. En detalle se presentan algunas de estas con scripts.

## 6.1. If-else

El intérprete de PHP evalúa la condición booleana, si ésta es verdadera se ejecutarán las instrucciones del primer bloque en caso, sino evalúa las instrucciones del segundo bloque. La expresión booleana puede ser una proposición simple o compuesta.

Figura 13. Estructura básicas de if-else en PHP

```
if (condición booleana) {  
    Instrucciones del  
    bloque 1  
} else {  
    Instrucciones del  
    bloque 2  
}
```

## 6.2. If elseif

Esta es una variante de la instrucción if - else, pero que permite más condicionales.

Figura 14. Estructura básicas de if-elseif en PHP

```
if (condición booleana 1) {  
    Sentencias del bloque 1  
} elseif (condición booleana 2) {  
    Sentencias del bloque 2  
} else {  
    Sentencias del bloque N  
}
```

### 6.3. Switch

A menudo es más conveniente utilizar la instrucción switch-case, que se puede entender como múltiples instancias del condicional if-else.

Figura 15. Estructura básica del switch-case en PHP

```
Switch( $variable ) {  
  
    Case valor1 :      Sentencias 1;  
    break;  
  
    Case valor2 :      Sentencias 2;  
    break;  
    .....  
  
    Case valor N :      Sentencias N;  
    break;  
    default: Sentencias ; break;  
}
```

A continuación se presenta un ejemplo de ésta estructura de control.

Figura 16. Script: Estructura básica switch-case en PHP

```
<html>  
<head> <title> Página de prueba</title> </head>  
<?php  
  
$MedidaAnguloRepresentativo = 45;  
  
if( ( ($MedidaAnguloRepresentativo > 0 ) && ( $MedidaAnguloRepresentativo < 180 ) )  
) {  
  
    Switch($MedidaAnguloRepresentativo ) {  
  
        case (( $MedidaAnguloRepresentativo > 90 ) and ( $MedidaAnguloRepresentativo  
< 180 ) ):  
  
            echo " es un triangulo obtusángulo"; break;  
  
        case ( $MedidaAnguloRepresentativo == 90 ):  
  
            echo " es un triangulo rectángulo"; break;  
  
        default: echo "Es un triangulo acutángulo"; break;  
  
    }  
  
}
```

```

    }} else {
    echo "El numero no corresponde a la medida un ángulo representativo de un
    triangulo";
    }
?>
</html>

```

## 7. Estructuras de ciclos

Los ciclos son estructuras de repeticiones o estructuras que permiten ejecutar de manera repetida instrucciones de acuerdo a alguna condición. Entre estas se encuentran las siguientes: for, foreach, while, Do while. Un par de sentencias que ayudan al manejo de ciclo que son Break y Continue.

### 7.1. For

Permite realizar un conjunto de instrucciones en un determinado número de veces. La sintaxis es la siguiente:

Figura 17. Estructura básica del switch-case en PHP

```

for( $variable = valorInicial; $variable <= valorFinal;
contador ){
    Sentencias;
}

```

El bucle puede estar determina el valor de la variable. Se ejecuta cada vez que finaliza una iteración. La sintaxis del bucle puede cambiar según se desee. El siguiente cuadro presenta algunos ejemplos:

Figura 18. Sintaxis de los contadores en el ciclo for en PHP

Sintaxis	Descripción
\$variable++	Aumenta en 1 la variable contador
\$variable--	Disminuye en 1 la variable contador
\$variable += N	Aumenta en N la variable contador
\$variable -= N	Disminuye en N la variable contador

El siguiente Script presenta ejemplifica el uso de esta estructura de repeticiones, éste calcula el factorial de un número:

Figura 19. Script: Factorial de un numero con la estructura for en PHP

```
<html>
<head> <title> Página de prueba</title>
</head>
<?php
$factorial = 1;
for($numero = 1; $numero <= 5;
$numero++) {
    $factorial = $factorial* $numero;
}
echo "El factorial de 5 es ". $factorial;
?>
</html>
```

## 7.2. Forecha

La estructura del Forecha permite recorrer las estructuras de tipo array, obteniendo en cada iteración cada uno de los elementos. Sirve para construir o interactuar con tablas de tipo hash. Un ejemplo a continuación permite ver múltiplos de 3 con el forecha:

Figura 20. Script: Estructura básica foreach en PHP

```
<html>
<head> <title> Página de prueba</title> </head>
<?php
echo "<table border='1'>";
echo "<tr><td align = 'center'> número </td>";
echo "<td align = 'center'> múltiplo </td> </tr>";

// llenamos la matriz con múltiplos de 3

for($i=0; $i < 11; $i++){

    $matriz[$i]= 3*$i;
}

// desplegamos los valores de la matriz
foreach ($matriz as $numero => $multiplo){

    echo "<tr><td align = 'center'>".$numero."</td>";
    echo "<td align = 'center'>".$multiplo ."</td>
</tr>";
}
echo "</table>";
?>
```

```
</html>
```

### 7.3. While

La estructura while utiliza un condición booleana que le indica hasta donde debe de iterarse. Para finalizar el ciclo, es necesario declarar un condicional o un contador de manera que la condición booleana se vuelva falsa. La sintaxis elemental es la siguiente:

Figura 21. Estructura básica del while en PHP

```
While (condición booleana){  
    Sentencias;  
}
```

El script muestra un ciclo iterado por con la estructura while para las potencias de 2:

Figura 22. Script Estructura básica while en PHP

```
<html>  
<head> <title> Página de prueba</title> </head>  
<?php  
$potencia = 1;  
$i = 0;  
while($ i < 11) {  
  
    echo "<br> 2 &nbsp;  <sup>". $i."</sup> es igual a ".$potencia;  
    $potencia = $potencia*2;  
    $i++; }  
  
?>  
</html>
```

### 7.4. Do while

La instrucción do-while cumple la misma función que en while, la única diferencia se da en que la condición booleana se evalúa hasta el final del ciclo. La sintaxis básica es

Figura 23. Estructura básica del do-while en PHP

```
do{  
  
    Sentencias;  
  
} while (condición booleana);
```

Por ejemplo, podemos evaluar la sumatoria  $\sum_{i=1}^5 \frac{i+2}{i}$  a través de un ciclo do-while.

Script N° 24. Script de la estructura básica do-while en PHP

```
<html>  
<head> <title> Página de prueba</title> </head>  
<?php  
$total = 0; $i = 0;  
do{  
    $total =$total + (($i+2)/5) ;  
    $i++;  
  
} while($i <= 6);  
echo " el total es ". $total;  
?>  
</html>
```

## 7.5. Break y continue

El uso del break y el continue permiten interrumpir o seguir ejecutando un ciclo de acuerdo a alguna condición. Estas condiciones se pueden utilizar en cualquiera de las estructuras de ciclos anteriores. En el ejemplo anterior, el efecto se puede visualizar si se cambia la sentencia continue por el break.

Figura 25. Script: Uso del break en PHP

```
<html>  
  
<head> <title> Página de prueba</title> </head>  
  
<?php  
  
echo "<table border='1'>";  
  
    for ($i = 0; $i < 3 ; $i++) {  
  
        echo "<tr>";
```



```

for ($j=0 ; ; $j++) {

    echo "<td>";

    if ($j >2) continue 2; {

        echo $i + $j;
    }

    echo "</td>";

}
echo "</tr>";
}

echo "</table>";
?>

</html>

```

## 8. Inclusión de ficheros

Por la naturaleza del cómo son programadas las paginas dinámicas, se hace necesario cargar archivos con el objetivo de no repetir código. Esta reutilización permite optimizar el código y facilitar el mantenimiento del mismo. La siguiente tabla muestra las distintas funciones para la inclusión de ficheros.

Figura 26. Funciones para la inclusión de ficheros en PHP

Función	Descripción <sup>5</sup>
require()	Incluye y evalúa el archivo especificado. Produce un Warning cuando hay un no encuentra el archivo solicitado.
Include()	Incluye y evalúa el archivo especificado. Produce un Errol Fatal cuando hay un no encuentra el archivo solicitado.
require_once()	Incluye y evalúa el fichero especificado durante la ejecución del script. Se comporta de manera similar a require() con la única diferencia que si el código ha sido ya incluido, no se volverá a incluir.

<sup>5</sup> Tomado de <http://www.php.net/manual/es/langref.php>

include_once()	Incluye y evalúa el fichero especificado durante la ejecución del script. Se comporta de manera similar a include_once() con la única diferencia que si el código ha sido ya incluido, no se volverá a incluir.
----------------	---

## 9. Funciones

Al igual que en las matemáticas, el lenguaje PHP soporta funciones. Se puede llamar tantas veces a las funciones como se necesite. Así de esta manera los scripts son más legibles y fáciles de interpretar.

Las funciones pueden recibir parámetros o no y devuelven algún otro.

Figura 27. Estructura básica de una función en PHP

```
function nombreFunción (parámetro1 , parametro2,...) {
    Sentencias;
}
```

Para llamar la función basta con seguir la siguiente sintaxis:

Figura 28. Llamado de una función en PHP

```
nombreFunción (parámetro1 , parametro2,...);
```

Para ilustrar el concepto de función en PHP, el script siguiente muestra la aproximación del número **e** como la serie  $e = \sum_{n=0}^{\infty} \frac{1}{n!}$ .

Figura 29. Script de aproximación del número **e** en PHP

```
<html>
<head> <title> Página de prueba</title> </head>
<?php

$n = 100;
while($i <= $n){
    $total = $total + ( 1 / factorial($i) );
    $i++;
}
echo $total;
function factorial ($j){
```

```
$factorial = 1;
$i=1;

if($j==0){
    return 1;
}else{
    while($i <= $j){
        $factorial = $factorial *$i;
        $i++;
    }
    return $factorial;
}
?>
</html>
```

### 10. Orientación a objetos

A partir de la versión de 5 de PHP, se incluye el soporte para la programación orientada a objetos (POO). La noción principal de POO es una clase que se puede interpretar como un tipo de variable. Un objeto contiene una serie de propiedades y/ métodos que tienen funcionalidad. Por ejemplo, el tipo **double** puede ser considerada como una clase; entonces la variable de esta clase tiene una propiedad (que el valor sea real) y una lista de métodos (suma, sqrt, etc.).

La sintaxis básica de una clase en PHP es la siguiente:

Figura 30. Estructura básica crear una clase en PHP

```
class nombreClase {
    sentencias;
    function(){
        .....
    };
}
```

Para instanciar un objeto de la clase **nombreClase** según el ejemplo anterior, se utiliza un operador denominado *new*. La sintaxis para crear un objeto es

Figura 31. Estructura básica para declarar un objeto en PHP

```
$objeto = new $nombreClase;
$objeto -> function();
```

Esto permite que cada objeto maneje sus propiedades sin el temor que sean modificadas. Las propiedades en un objeto son una lista de atributos con un valor definido. Para acceder a estas propiedades se utiliza la palabra reservada **\$this**, ésta es una referencia al objeto concreto en el método que se está ejecutando la cual permiten acceder a sus métodos y propiedades.

En algunos casos, se puede usar la encapsulación para acceder a las propiedades de un objeto únicamente por medio de sus métodos quedando estas como ocultas o inaccesibles. De esta manera, los métodos se convierten en un puente de comunicación. Algunas de estas propiedades se definen como privadas para no alterar su valor. Esta se declara con la palabra reservada **private**.

Otros métodos comunes en la implementación de objetos en el control de sus métodos y variables son:

Figura 32. Métodos para POO en PHP

Método	Descripción
<code>_set()</code> y <code>_get()</code>	Permiten capturar los valores de los procesos en propiedades que no están declaradas en un objeto.
<code>_call()</code>	Permite llamar métodos no declarados
<code>_construct()</code>	Inicializa las variables, objetos, conexiones, etc.
<code>_destruct()</code>	Destruye las variables, objetos, conexiones, etc.

Los atributos y métodos estáticos son aquellos que pueden utilizarse desde la clase sin necesidad de crear un objeto. Para que un método o una variable estática se declare se antecede la palabra **static**.

La herencia permite de forma jerárquica la relación de dos o más clases, de las cuales una es la clase es "padre" y la otra es la clase "hija" a la cual se le hereda todas las atributos. A la clase "padre" usualmente se denomina la Superclase y a

la clase hija se le llama "subclase". Para que una subclase herede las características de un la superclase se debe declarar con la palabra reservada **extends**. La sintaxis en PHP es la siguiente:

Figura 33. Estructura básica para declarar herencia en PHP

```
<?php
class superClase{
    atributos;
    métodos;
    .....
}
?>
<?php
class subClase extends superClase {
    .....
}
?>
```

Cuando una variable se necesite que sea privada para cualquier otro método pero no para las clases que son heredadas de ésta, se puede cambiar la palabra reservada `private` por **protected**. Así, sólo las clases heredadas pueden tener acceso a los atributos y métodos de la superclase. Si se define una clase como final, esto indica que no se pueden declarar subclases de este. La sintaxis para declarar una clase final es:

Figura 34. Estructura básica para declarar final una clase en PHP

```
<?php
final class miClase{
    .....
}
?>
```

El lenguaje PHP permite declarar súper clases que no pueden ser instanciadas por si solas. Únicamente son instanciadas por clases heredan sus atributos. A éstas se les denomina clases abstractas. Cabe la posibilidad de que haya múltiples niveles de herencia de clases abstractas las cuales solo se podrán

instanciar en los niveles inferiores en clases concretas. De igual manera se pueden declarar funciones abstractas. La sintaxis es la siguiente:

Figura 35. Estructura básica para declarar abstracta una función y una clase en PHP

```
<?php
    abstract class miClase{
        abstract function mifunción( );
    }
?>
```

Una variante de la clase abstracta es la **interface**. Su funcionalidad es muy similar con respecto a la instanciación de subclases, se diferencia en que en ésta sólo se pueden declarar sólo métodos. Los atributos no se pueden declarar en esta clase. Para declarar la herencia de las clases interface se utiliza la palabra reservada **implements**. La sintaxis es la siguiente:

Figura 36. Estructura básica para declarar abstracta una función y una clase en PHP

```
<?php
Interface SuperInterfaz{
    function mifunción( );
}
Interface SubInterfaz implements SuperInterfaz {
    function mifunción( ){
        sentencias;
    };
};
//Instanciación del objeto
$p = new SubInterfaz;
$p -> mifunción();
?>
```

Como cualquier otro lenguaje de programación orientado a POO, PHP soporta el polimorfismo. Esto es que significa que clases diferentes tendrán un

comportamiento distinto para la misma operación. Esto se debe a que posee a la habilidad de un objeto de determinar que método invocar para un objeto pasado como argumento en tiempo de ejecución real.

## **11. Consideraciones finales**

El lenguaje PHP es un programa orientado a la programación y creación de sitios Web dinámicos, por medio del cual el usuario es capaz de hacer uso de su creatividad y habilidad para implementarlo en sus actividades profesionales. Es un lenguaje fácil de aprender que permite de forma gratuita poder construir sus propios recursos y subirlos a Internet, sin necesidad de recurrir a recursos técnicos sofisticados.

Existen muchos frameworks diseñados con PHP de código abierto que pueden ser modificados y adaptados de acuerdo a las necesidades, lo cual permite compartir los programas elaborados con otras personas e intercambiar experiencias. Además, existen actualmente una gran cantidad de librerías matemáticas tales como JpGraph que permite elaborar de manera interactiva gráficas estadísticas.

La guía anterior es apenas el comienzo para tener un conocimiento básico para trabajar con el programa, es tarea del usuario profundizar los conocimientos con ayuda de manuales o explorando de manera autodidacta.

Las aplicaciones de estos programas en la educación están comenzando a tomar fuerza por las grandes ventajas que poseen. Es importante que los educadores se actualicen constantemente con los recursos tecnológicos que estén a su alcance, para ampliar sus recursos didácticos y aprovecharlos en metodologías que le permitan mejorar su quehacer diario.

## 12. Referencias

- [1] PHP Home Page: [www.php.net](http://www.php.net)
- [2] Koterov, D. Creating your web site with PHP. 2002. Editorial Jessica Mroz.
- [3] Gutiérrez, A. Bravo, G. PHP5 a través de ejemplos. 2005. Editorial Alfaomega.
- [4] Ullman, L. PHP y MySQL for dynamics web sites. 2003. Editorial Pearson Education.
- [5] Wikipedia. En Internet: <http://es.wikipedia.org/>