

PHP

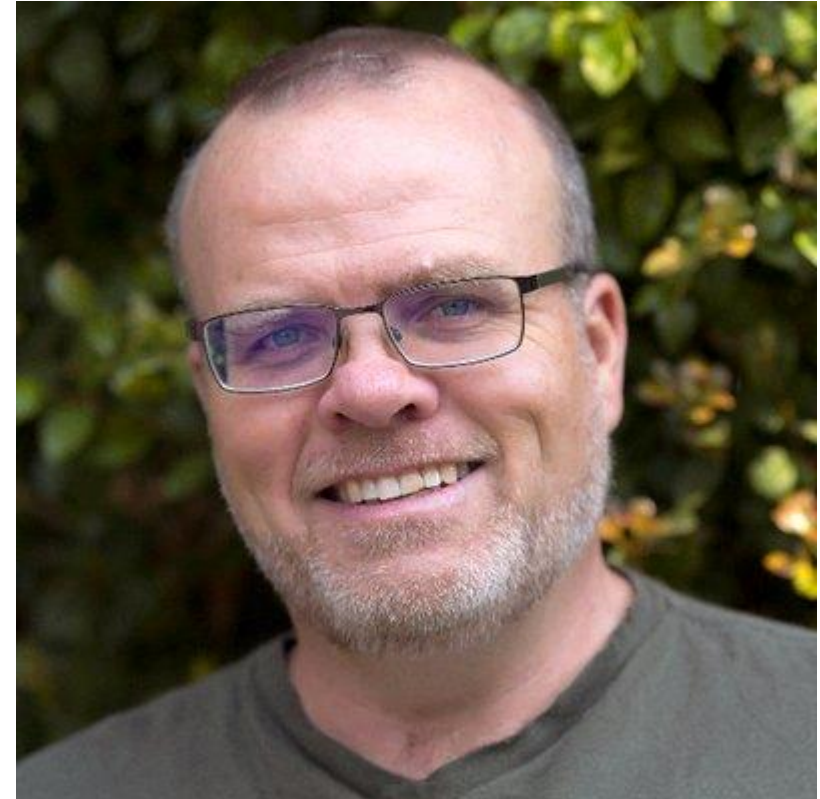
1.1

Introducción

PHP se creó en 1994 por Rasmus Lerdorf.

Su extensión es .php

PHP: Hypertext
Preprocessor



1.1

Introducción

PHP es un lenguaje de programación destinado a desarrollar **aplicaciones para la web y crear páginas web**, favoreciendo la **conexión entre los servidores y la interfaz de usuario**.











Entre los factores que hicieron que PHP se volviera tan popular, se destaca el hecho de que es de código abierto.

Cualquiera puede hacer cambios en su estructura. Esto representa dos cosas importantes:

1. Es de código abierto, no hay restricciones de uso vinculadas a los derechos. El usuario puede usar PHP para programar en cualquier proyecto y comercializarlo sin problemas.
2. Está en constante perfeccionamiento, gracias a una comunidad de desarrolladores proactiva y comprometida.

1.1

Introducción

Sep 2021	Sep 2020	Change	Programming Language		Ratings	Change
1	1			C	11.83%	-4.12%
2	3	▲		Python	11.67%	+1.20%
3	2	▼		Java	11.12%	-2.37%
4	4			C++	7.13%	+0.01%
5	5			C#	5.78%	+1.20%
6	6			Visual Basic	4.62%	+0.50%
7	7			JavaScript	2.55%	+0.01%
8	14	▲▲		Assembly language	2.42%	+1.12%
9	8	▼		PHP	1.85%	-0.64%
10	10			SQL	1.80%	+0.04%

1.1

Introducción

Generalmente es definido como un lenguaje del lado del servidor. Esto significa que se aplica en la programación que tiene lugar en el servidor web responsable de ejecutar la aplicación o, más a menudo, en un sitio web.

El código PHP se ejecuta en el servidor que, al leer los comandos, puede activar todos los elementos funcionales y la interfaz visual del sitio web.

La simplicidad para aprender a usarlo y el desarrollo del código abierto le facilita el trabajo.

A medida que avanzan las configuraciones y ediciones se simplifican aún más.

1.1

Introducción

EL PODER DE PHP

- PHP puede generar contenido de página dinámica.
- PHP puede crear, abrir, leer, escribir, eliminar y cerrar archivos en el servidor.
- PHP puede recopilar datos de formularios.
- PHP puede enviar y recibir cookies.
- PHP puede agregar, eliminar, modificar datos en su base de datos.
- PHP se puede utilizar para controlar el acceso de los usuarios.
- PHP puede cifrar datos.

Con PHP no está limitado a generar HTML. Puede generar imágenes, archivos PDF e incluso películas Flash. **También puede generar cualquier texto, como XHTML y XML.**

1.1

Introducción

EL PODER DE PHP

- PHP puede generar contenido de página dinámica.
- PHP puede crear, abrir, leer, escribir, eliminar y cerrar archivos en el servidor.
- PHP puede recopilar datos de formularios.
- PHP puede enviar y recibir cookies.
- PHP puede agregar, eliminar, modificar datos en su base de datos.
- PHP se puede utilizar para controlar el acceso de los usuarios.
- PHP puede cifrar datos.

Con PHP no está limitado a generar HTML. Puede generar imágenes, archivos PDF e incluso películas Flash. **También puede generar cualquier texto, como XHTML y XML.**

PHP se ejecuta en varias plataformas (Windows, Linux, Unix, Mac OS X, etc.). Además es compatible con casi todos los servidores que se utilizan en la actualidad.

1.2 Conceptos básicos

```
<?php
```

```
// De esta forma se hace un comentario.
```

```
?>
```

- Los ficheros que contengan código PHP se deben de guardar con la extensión .php.
- El código cohabita con HTML.

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<h1>Página principal </h1>
```

```
<?php
```

```
echo "1, 2, 3 probando el código";
```

```
?>
```

```
</body>
```

```
</html>
```


1.2 Conceptos básicos

- Es importante acabar el código de PHP con ;
- Probar escribir el echo de forma diferente, es decir, con mayúsculas y minúsculas.

1.2 Conceptos básicos

Definición de variables

- Una variable definida en el lenguaje PHP **SIEMPRE COMIENZA** con el signo \$ y la precede el nombre de la variable.

```
<!DOCTYPE html>
<html>
<body>

<h1>Página principal </h1>

<?php
echo "1, 2, 3 probando el código";
$usuario= "Horacio";
echo "Bienvenido".$usuario."<br>";

?>

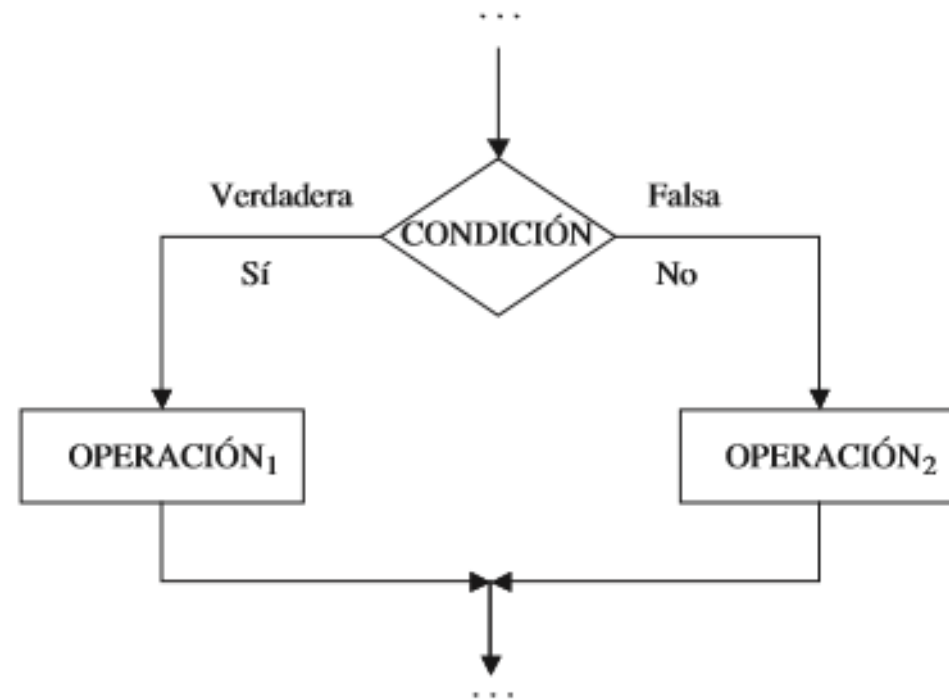
</body>
</html>
```

1.2 Conceptos básicos

- Probar escribir el nombre de la variable de forma diferente cuando la invocamos, es decir, con mayúsculas y minúsculas en la parte del echo.
- Probar a concatenar sin los puntos.

1.3 Algoritmos

IF- ELSE SIMPLE

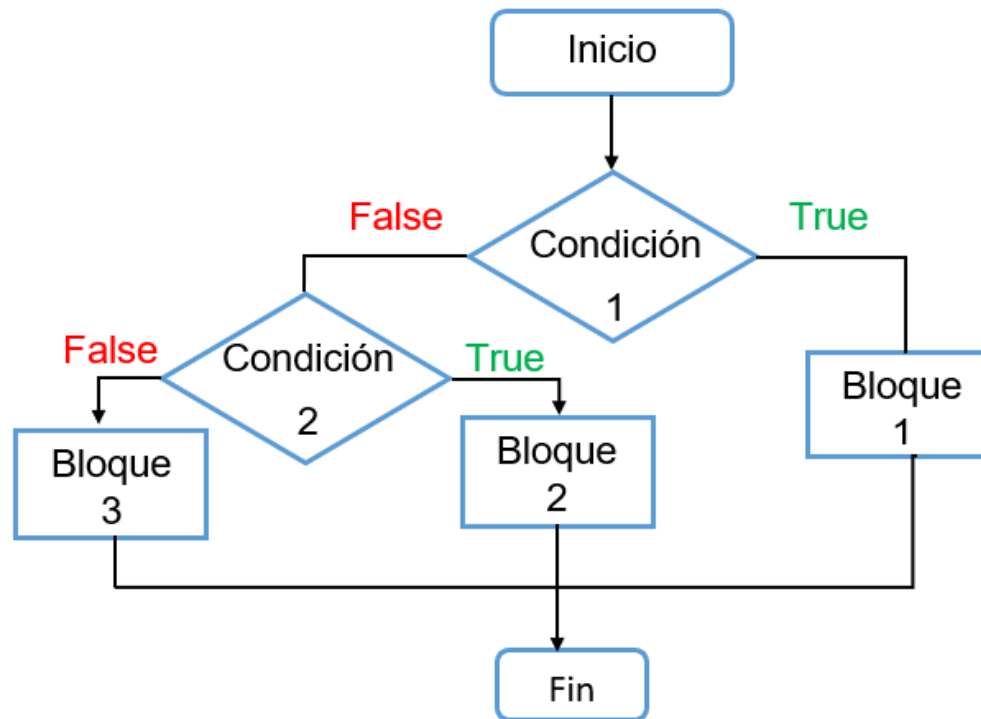


```
if (condición ) {  
    Código operación 1.  
} else {  
    Código operación 2.  
}
```

Mirar el repositorio para ver un ejemplo

1.3 Algoritmos

IF- ELSE MÚLTIPLE

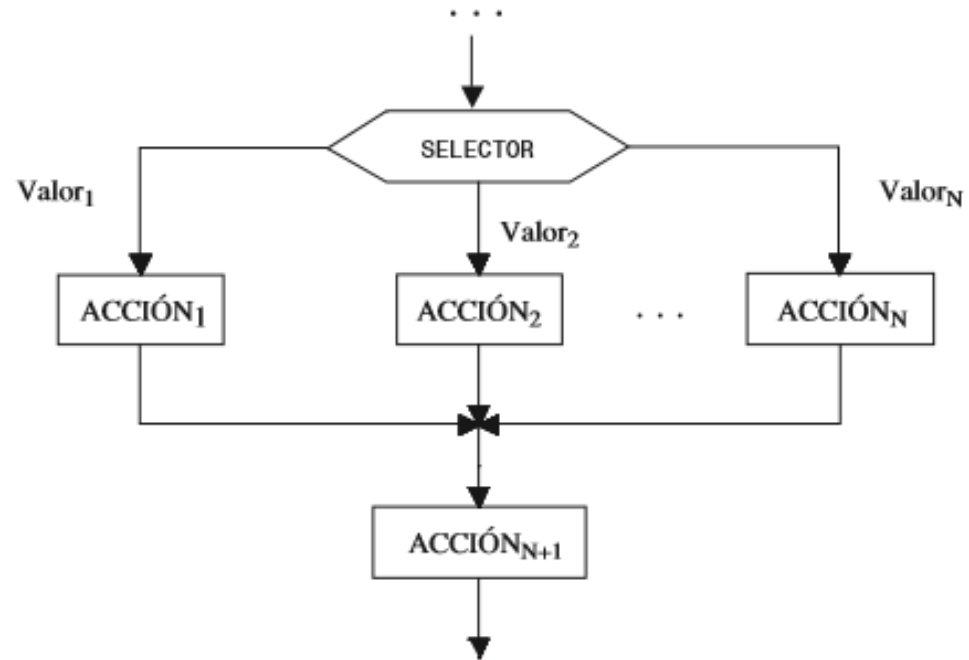


```
if (condición 1) {  
    Código bloque 1.  
} elseif (condición 2) {  
    Código bloque 2.  
} else {  
    Código bloque 3.  
}
```

Mirar el repositorio para ver un ejemplo

1.3 Algoritmos

SWITCH



```
switch (selector ) {
```

```
    case valor1:  
        Código acción 1;  
        break;
```

```
    case valor2:  
        Código acción 2;  
        break;
```

```
    .  
    .  
    .
```

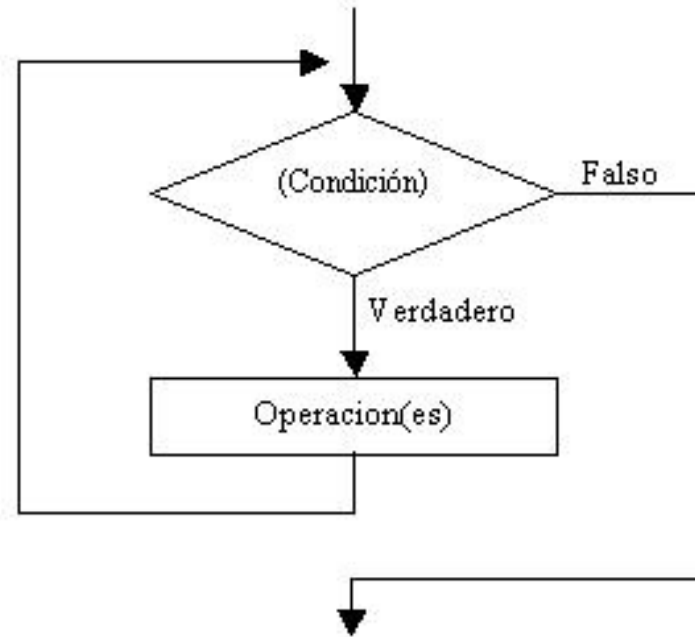
```
    default:  
        Código acción N;  
        break;
```

```
}
```

Mirar el repositorio para ver un ejemplo

1.3 Algoritmos

WHILE

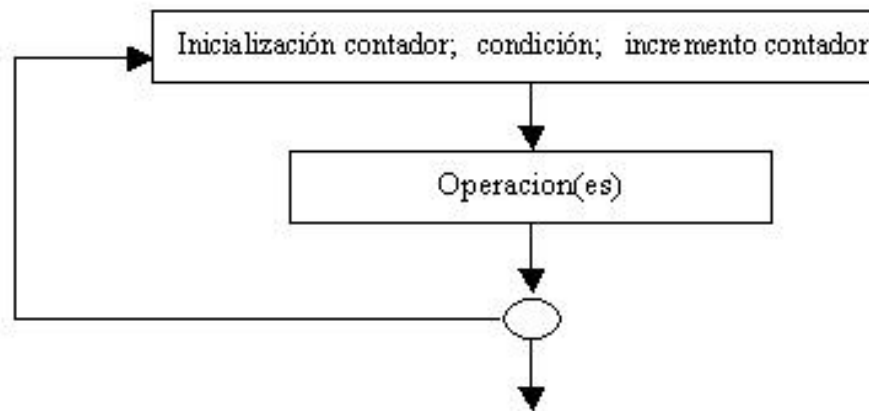


```
while (Condición verdadera) {  
    Código de operación;  
}
```

Mirar el repositorio para ver un ejemplo

1.3 Algoritmos

FOR



```
for (inicialización, condición, incremento) {  
    Código de operación;  
}
```

Mirar el repositorio para ver un ejemplo

1.3 Algoritmos

FUNCIÓN

Función sin argumentos pasados

```
function nombreFunción() {  
  Código a ejecutar;  
}
```

Función con argumentos pasados

```
function nombreFunción($argumento) {  
  Código a ejecutar;  
}
```

Mirar el repositorio para ver un ejemplo

1.4 Recogida de datos

- El atributo `method` especifica el método HTTP que se utilizará al enviar los datos del formulario.
- Los datos del formulario se pueden enviar como variables de URL (con `method="get"`) o como transacción de publicación HTTP (con `method="post"`).
- La longitud de una URL es limitada (2048 caracteres)
- Útil para envíos de formularios donde un usuario quiere marcar el resultado
- GET es bueno para datos no seguros, como cadenas de consulta en Google.
- Agrega los datos del formulario dentro del cuerpo de la solicitud HTTP (los datos del formulario enviado no se muestran en la URL).
- POST no tiene limitaciones de tamaño y se puede utilizar para enviar grandes cantidades de datos.

1.4 Recogida de datos

Recogida de variable mediante el método GET

```
$variable=$_GET["valor del name"];
```

Recogida de variable mediante el método POST

```
$variable=$_POST["valor del name"];
```

Mirar el repositorio para ver un ejemplo

1.5 Algunas funciones vista en clase

Funciones matemáticas

- `pow($base,$exponente)` → Elevar al exponente cualquier base.
- `rand($rangomin,$rangomax)` → Generar un número aleatorio.
- `sqrt($numero)` → Raíz cuadrada de un número.
- `base_convert($numero,baseinicial,basefinal)` → Cambio de base.

Funciones strings

- `strlen($cadena)` → Contar el número de caracteres de la cadena.
- `strrev($cadena)` → Invierte la cadena.
- `str_word_count($numero)` → Raíz cuadrada de un número.
- `str_replace($busqueda,$nuevapalabra,$palabra)` → Reemplaza palabras.

1.6 Inclusión de archivos

Inclusión de archivos en PHP

PHP nos permite insertar cualquier tipo de archivos con formato de texto, dentro de un archivo PHP.

Entre los tipos de archivos que podemos insertar dentro de un fichero .php, se encuentran aquellos con las siguientes extensiones: .php, .txt, .htm, .html, entre otros con formato de texto.

1.6 Inclusión de archivos

Para insertar archivos, PHP dispone de cuatro funciones:

1. `include`
2. `include_once`
3. `require`
4. `require_once`

Estas cuatro funciones, necesitan recibir como parámetro, la ruta local o remota del archivo a ser incluido.

```
<?php  
include("archivo.php");  
include_once("archivo.txt");  
require("archivo.html");  
require_once("archivo.htm");
```

1.6 Inclusión de archivos

Diferencia entre include y require

Si bien las funciones `require()` e `include()` de PHP realizan una acción similar (importar un archivo), no son iguales. `include()` intenta importar al archivo indicado y en caso de no poder hacerlo, arroja un error y continúa ejecutando el resto del script.

Sin embargo, la función `require()`, cuando no logra importar el archivo indicado, arroja un error y finaliza sin permitir que el resto del script continúe ejecutándose.

Include y require "_once"

La única diferencia que existe entre `include` e `include_once` y `require` y `require_once`, es que si el archivo indicado con `"_once"` ya ha sido incluido no volverá a importarse.

1.7 Matrices (arrays)

Una matriz (array) es un mapa de datos ordenado que asocia "claves" a sus valores correspondientes. Es así, que estas matrices, nos son de gran utilidad para crear desde diccionarios de datos hasta árboles de múltiples diccionarios.

En la asignaturas utilizaremos 3 tipos:

1. Arrays normales.
2. Arrays asociativos.
3. Arrays multidimensionales.

1.7 Matrices (arrays)

Arrays normales.

```
<?php  
$numeros = array(38 , 36 , 6 , 2 , 16 , 19);  
?>
```

Arrays asociativos.

```
<?php  
$persona = array(  
    'nombre' => Pepe',  
    'apellido' => 'Martinez',  
    'edad' => 27,  
    'programador' => true  
);  
?>
```

1.7 Matrices (arrays)

Arrays multidimensionales.

```
<?php
$personas = array(
    array(
        'nombre' => Manuel',
        'apellido' => Gutierrez',
        'edad' => 27,
        'programador' => true
    ),
    array(
        'nombre' => 'Tomás',
        'apellido' => Pina',
        'edad' => 20,
        'programador' => true
    ),
    array(
        'nombre' => 'Andrés',
        'apellido' => 'Martínez',
        'edad' => 32,
        'programador' => false
    )
);
?>
```

1.8 foreach

El bucle foreach es un constructor nativo de PHP, que permite realizar operaciones iterativas (cíclicas) con matrices, recorriendo uno a uno los elementos de una matriz, comenzando por el primer elemento.

Sintaxis

```
foreach($array as $valor_del_elemento) {  
    // Código  
}
```

Sintaxis

Es posible también, iterar obteniendo las claves de cada elemento, además de su valor. Para ello, se utiliza la siguiente sintaxis:

```
foreach($array as $clave => $valor) {  
    // Código  
}
```

1.9 Tratamientos de fechas

mktime()

Otra forma de recuperar la fecha y la hora en PHP es en formato timestamp (o marca de tiempo Unix), esto es la cantidad de segundos que va desde de la época Unix (1 de Enero de 1970 a las 00:00:00) hasta el momento en que se llama. Esta función recibirá 6 parámetros: la hora, los minutos, los segundos, el mes, el día y el año (en ese orden):

```
<?php
$mktime = mktime('19', '50', '02', '09', '19', '2013');
echo $mktime;
?>
```

1.9

Tratamientos de fechas

mktime()

Otra forma de recuperar la fecha y la hora en PHP es en formato timestamp (o marca de tiempo Unix), esto es la cantidad de segundos que va desde de la época Unix (1 de Enero de 1970 a las 00:00:00) hasta el momento en que se llama. Esta función recibirá 6 parámetros: la hora, los minutos, los segundos, el mes, el día y el año (en ese orden):

```
<?php
$mktime = mktime('19', '50', '02', '09', '19', '2013');
echo $mktime;
?>
```

1.10 Ficheros

Php nos da la posibilidad de crear, modificar, eliminar y recuperar el contenido de archivos de texto como por ejemplo txt, xml o incluso html.

Creación de archivos

Para crear y editar archivos debemos usar dos funciones, **fopen()** y **fwrite()**, la primer función creará un archivo si no existe, y si existe lo reemplazará por el nuevo. **fwrite()** nos permitirá escribir el contenido del mismo.

Luego usamos la función **fwrite()** para escribir el contenido del archivo y finalmente cerramos el mismo con **fclose()**.

```
<?php
$contenido = 'Texto de prueba';
$archivo = fopen('archivo.txt', 'w');
fwrite($archivo, $contenido);
fclose($archivo);
?>
```

1.10 Ficheros

- **w** → Para escritura, si el archivo no existe lo crea y si ya existe elimina sus contenidos.
- **w+** → Para escritura y lectura, si el archivo no existe lo crea y si ya existe elimina sus contenidos.
- **r** → Para lectura.
- **r+** → Para lectura y escritura
- **a** → Para adjuntar, escribe al final del mismo, si no existe intenta crearlo.

```
<?php
$contenido = 'Texto de prueba';
$archivo = fopen('archivo.txt', 'w');
fwrite($archivo, $contenido);
fclose($archivo);
?>
```

