

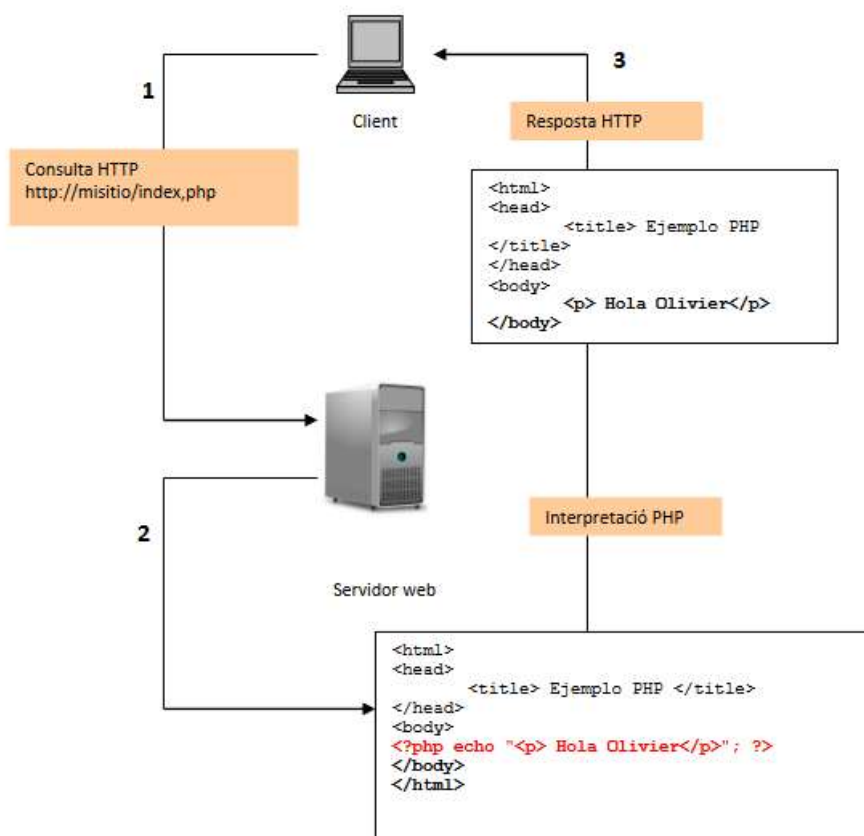
1. Introducció

PHP és un llenguatge d'script que s'executa al costat del servidor.

Els scripts s'incrusten en els documents HTML i el servidor web els interpreta i executa abans de servir les pàgines als clients.

El client (navegador) no veu el codi PHP sinó el resultat que produeix.

Vegem el següent esquema per veure com es tracta un arxiu PHP pel servidor web:



2. Què és pot fer amb PHP

A nivell més bàsic, PHP pot fer qualsevol cosa que es pugui fer amb un script CGI, com processar la informació de formularis, generar, pàgines amb contingut dinàmic, enviar i rebre cookies o interactuar amb el sistema: borra arxius, crear...

A més la majoria de les funcions més útils ja estan predefinides:

- Connectivitat: HTTP, FTP, COM, YP/NIS, SNMP, Sockets, CORBA, LDAP.
- Serveis de correu i notícies: POP, IMAP, SMTP, NNTP
- Text i gràfics: XML, HTML, PDF, GD, Flash.
- Funcions matemàtiques.
- POSIX: semàfors, memòria compartida, accés a fitxers, expressions regulars, cronòmetres.
- Comerç electrònic: Cybercash, Verisign.
- Formularis.
- Encriptació i compressió: MD5, SHA, Gzip, Bzip2, OpenSSL.

Però unes de les característiques més destacades és el suport per a un gran nombre de bases de dades. Escriure una interfície via web per a una base de dades és tasca senzilla amb PHP.

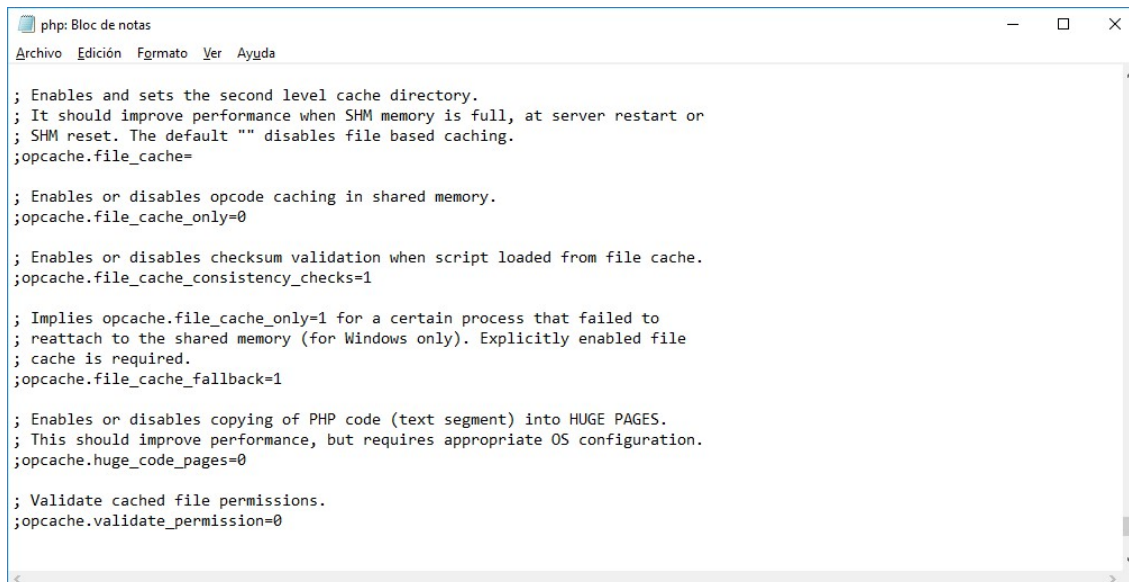
Algunes d'aquestes bases de dades suportades són:

- dBase
- Adabas D
- IBM DB2
- Informix
- InterBase
- mSQL
- MySQL
- ODBC
- Oracle (OCI7 i OCI8)
- PostgreSQL
- SyBase

PHP també suporta l'ús d'altres serveis que utilitzin protocols com IMAP, SNMP, NNTP, POP3, HTTP, IRC i derivats. També es poden obrir sockets de xarxa directes (rawsockets) i interactuar amb altres protocols.

3. Configuració de PHP

Al arxiu **php.ini** de configuració es troben les directives de configuració que s'utilitzen per modificar el comportament del PHP.



```
php: Bloc de notes
Archivo Edición Formato Ver Ayuda

; Enables and sets the second level cache directory.
; It should improve performance when SHM memory is full, at server restart or
; SHM reset. The default "" disables file based caching.
;opcache.file_cache=

; Enables or disables opcode caching in shared memory.
;opcache.file_cache_only=0

; Enables or disables checksum validation when script loaded from file cache.
;opcache.file_cache_consistency_checks=1

; Implies opcache.file_cache_only=1 for a certain process that failed to
; reattach to the shared memory (for Windows only). Explicitly enabled file
; cache is required.
;opcache.file_cache_fallback=1

; Enables or disables copying of PHP code (text segment) into HUGE PAGES.
; This should improve performance, but requires appropriate OS configuration.
;opcache.huge_code_pages=0

; Validate cached file permissions.
;opcache.validate_permission=0
```

4. Sintaxis del llenguatge PHP bàsica

4.1. Etiqueta PHP i normes de denominació

A la versió 7 PHP accepta dues sintaxis per les etiquetes:

```
<?php .... ?>
<? .....?>
```

La primera és la més habitual i més recomanada.

La segona només és possible si al arxiu de configuració php.ini la directiva `short_open_tag` està en on.

Qualsevol entitat en PHP amb nom (constant, variable, funció, ...) ha de tenir un nom que respecti les següents regles:

- ✓ Començar amb lletra o _
- ✓ A continuació ha de tenir lletres, números o guions baixos.

4.2. HTML generat amb PHP

Per imprimir

```
echo "Hola a tots";  
print "Hola a tots";
```

Si posem `\n` genera codi HTML més llegible.

codi PHP

```
print("<p> paragraf 1 </p>");  
print("<p> paragraf 1 </p>");
```

codi HTML generat

```
<p> paragraf 1 </p> <p> paragraf 1 </p>;
```

codi PHP

```
print("<p> paragraf 1 </p>\n");  
print("<p> paragraf 1 </p>\n");
```

codi HTML generat

```
<p> paragraf 1 </p>  
<p> paragraf 1 </p>;
```

4.3. Comentaris

PHP ofereix dues sintaxis:

- `//` o `#` per inserir comentaris a una sola línia o després d'una instrucció.
- `/* */` per inserir comentaris en diverses línies.

```
1 <?php  
2 print "Hola mon"; // comentari hasta el final de la línia  
3 echo "Primeros pasos amb PHP" #comentari hasta el final de la línia  
4 /*comentari en  
5 varies línies*/  
6 echo "Clase de 2ASI";  
7 ?>
```

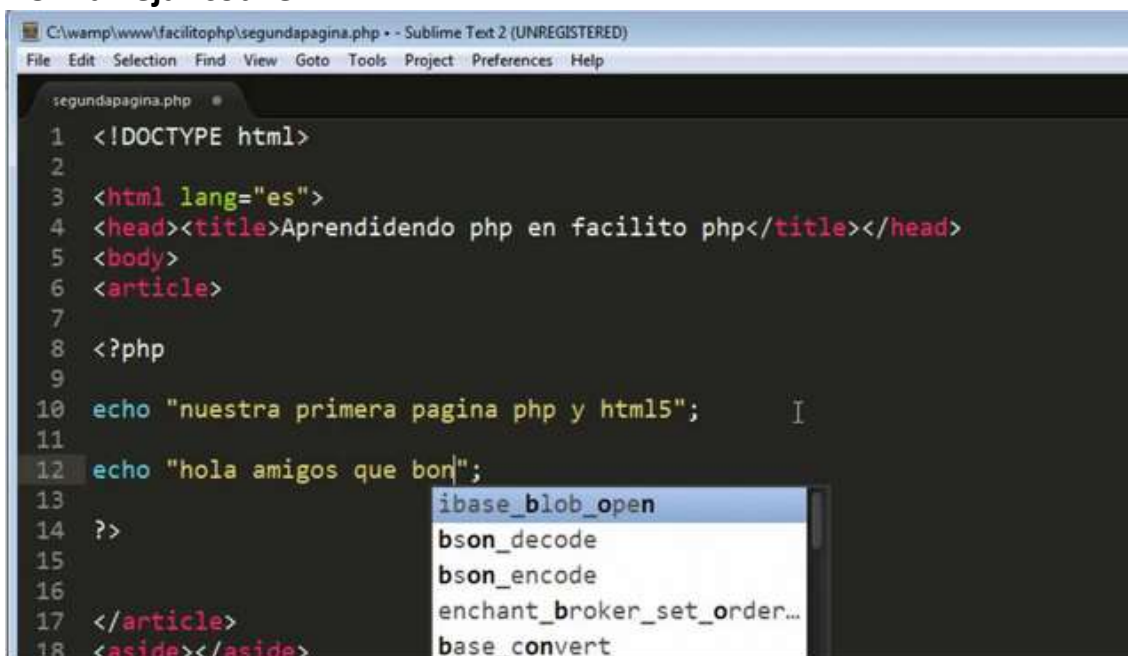
4.4. Separador d'instruccions

En PHP totes les instruccions han d'acabar amb punt i coma.

Exemple:

```
1  <?php
2  echo "Hola ";
3  echo "Olga!";
4  ?>
```

4.5. Barrejar codi en PHP i HTML



```
C:\wamp\www\facilito php\segundapagina.php - Sublime Text 2 (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

segundapagina.php
1  <!DOCTYPE html>
2
3  <html lang="es">
4  <head><title>Aprendiendo php en facilito php</title></head>
5  <body>
6  <article>
7
8  <?php
9
10 echo "nuestra primera pagina php y html5";
11
12 echo "hola amigos que bon!";
13
14 ?>
15
16
17 </article>
18 <aside></aside>
```

4.6. Constants

La funció **define** o la paraula clau **const** permeten definir una constant.

Una constant té un àrea de memòria identificada per un nom que conté un valor llegible, però no modificable pel programa.

Sintaxi

```
1 <?php
2 define('COLOR', 'vermell');
3 const ROPA = "bufanda";
4 echo COLORES;
5 echo ROPA;
6 ?>
```

Àmbit

L'àmbit d'una constant és l'script on es defineix . És a dir es pot definir en un primera secció de codi PHP i utilitzar-la en un altra secció del mateix script.

Exemple:

```
1 <?php
2 //Definim una constant
3 define('NOM', 'Olga');
4 ?>
5 <!DOCTYPE html>
6 <html>
7 <head>
8
9 </head>
10 <body>
11
12 <h1> Hola <?php echo NOM ?></h1>
13
14 </body>
15
16
17
18 </html>
```

4.7. Variables

Una variable té un àrea de memòria identificada per un nom que conté un valor llegible i modificable pel programa.

Les variables tenen el prefix \$ seguit d'un nom que compleixi amb les regles de denominació.

Aquest nom és sensible a majúscules i minúscules.

Exemple:

```
1  <?php
2
3  $nombre= "Olga";
4  $cognom = "Domène";
5
6  echo "El meu nom és $nombre $cognom";
7  |
8  ?>
```

Les variables PHP es defineixen automàticament i se les assigna valor amb =.

El tipus de la variable es defineix i redefineix de manera automàtica. depenen de la última assignació realitzada, adoptant un o altre tipus. La funció **gettype(nom_var)** permet obtenir el tipus d'aquesta variable en forma de cadena

Per obtenir el tipus d'una variable s'utilitzen les següents funcions:

[is_float\(\\$varname\).](#)
[is_array\(\\$varname\).](#)
[Is_String\(\\$varname\)](#) i
[Is_Object\(\\$varname\).](#)
[Is_int\(\\$varname\)](#)
[is_bool\(\\$varname\)](#)

Àmbit

L'àmbit d'una variable és l'script on es defineix . És a dir es pot definir en un primera secció de codi PHP i utilitzar-la en un altra secció del mateix script.

4.8. Tipus de dades

PHP disposa de quatre tipus bàsics i dos tipus compostos de dades.

Tipus escalars

1. Número enter
2. Número de punt flotant
3. Cadena de caràcters.
4. Booleà.

Tipus compostos

1. Matriu
2. Objecte

Tipus especials

1. NULL: tipus d'una variable sense haver sigut inicialitzada.
2. Recurs: és una mica particular i fa referència a un recurs extern: arxiu obert, connexió a base de dades, etc.

```
1  <?php
2
3  //Números enters
4  $numeroEnter = 5;
5
6  //Números amb punt flotant
7  $numeroEnter = 5.5;
8
9  //Cadena de caràcters
10
11 $cadena = "HoLa mon";
12 // Booleà
13
14 $esCert = true;
15 ?>
16
```

seqüència	significat
\n	nova línia
\t	tabulació horitzontal
\\	barra invertida
\\$	signe dòlar
\"	cometes dobles

4.8.1 Enters

El tipus enter (integer) permet emmagatzemar números entre -2^{31} i 2^{31} .

Els valor més gran i més petit admesos per la plataforma son els facilitats per les constants PHP_INT_MAX i PHP_INT_MIN respectivament. Aquesta última està des de la versió PHP7.

4.8.2. Número de punt flotant

El tipus float permet emmagatzemar números entre 10^{-308} i 10^{308} .

En el cas de conversió d'un float a un integer, el número es trunca en comptes d'arrodonir-se.

4.8.3. Cadena de caràcters

El tipus string permet emmagatzemar qualsevol seqüència de caràcters ASCII sense limitació de mida.

Els strings poden estar limitats per cometes dobles o cometes simples.

Quan una cadena està delimitada per cometes dobles, les variables es substitueixen pel seu valor, no succeeix el mateix quan estan delimitades per cometes simples.

```
1 <?php
2 $nombre = 'Pol';
3 echo "Yo me llamo $nombre";
4 echo 'Yo me llamo $nombre'
5 ?>
6
7
```



Yo me llamo Pol
Yo me llamo \$nombre

És possible accedir al enèsim caràcter d'una cadena utilitzant la notació $\$x[i]$, on $\$x$ és la variable de tipus cadena i i el número del caràcter (el primer és el 0) també es poden utilitzar claus.

```
1 <?php
2 $nombre = 'David';
3 echo $nombre[0], $nombre[6];|
4
5 ?>
6
7
```



Dd

4.8.4. Booleans

Aquest tipus pot prendre dos valors: TRUE o FALSE.

PHP és capaç de convertir qualsevol tipus de dades en un booleà

Valor	Resultat de a conversió
Nombre enter 0	FALSE
Nombre decimal 0.000....	
Cadena buida ""	
Cadena igual a "0"	
Matriu buida	
Objecte buit	
Constant NULL	
Tota la resta	TRUE

4.8.5. Matrius

Una matriu és un col·lecció ordenada per la parella clau/valor.

Tipus de matrius segons el tipus de clau:

- Matriu numèrica : la clau és un número enter.
- Matriu associativa: la clau són cadenes i no han d'estar ordenades .

El valor pot ser de qualsevol tipus, incloent el tipus matriu. En aquest cas se les anomena matrius multidimensionals.

Exemples:

Matriu numèrica

Clau/Índex	Valor
0	cero
1	Un
2	Dos
3	Tres

Matriu numèrica (índex no ordenats ni consecutius)

Clau/Índex	Valor
20	Pomes
30	Taronges
10	Olives
50	Peix

Matriu multidimensional (llista de ciutats per país)

Clau/Índex	Valor	
Espanya	Clau/Índex	Valor
	0	Barcelona
	1	Madrid
	2	Asturies
Itàlia	Valor	
	Clau/Índex	Valor
	0	Roma
	1	Venècia
	3	Milà

4.8.5.1 Creació

Es pot declarar utilitzant la funció array, o implícitament utilitzant [].

Amb una assignació del tipus \$matriu = valor, PHP busca l'índex enter major utilitzat i associa el valor al índex immediatament superior. Si la matriu està buida, l'element és col·locarà a l'índex 0.

Amb una assignació del tipus \$matriu[clau] = valor, PHP associa el valor a la clau especificada (que pot ser de tipus enter o cadena).

Ambdues notacions es poden barrejar en una seqüència de comandes.

Exemple de creació d'un array unidimensional

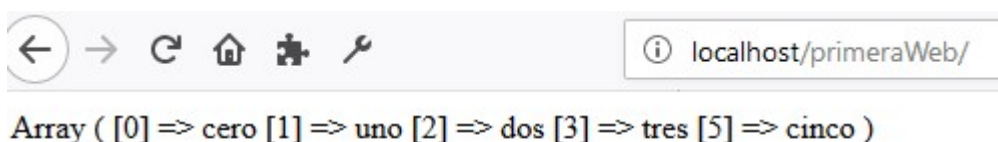
```
1  <?php
2
3  $numeros[] = 'AAAA'; // => index 0
4  $numeros[] = 'BBBB'; // => index 1
5  $numeros[] = 'CCCC'; // => index 2
6  $numeros[] = 'DDDD'; // => index 3
7  $numeros[5] = 'EEEE'; // => index 5
8  $numeros[] = 'FFFF'; // => index 6
9  $numeros[1] = 1; // => index 6
10
11 print_r($numeros);
12 ?>
13
14
```



Array ([0] => AAAA [1] => 1 [2] => CCCC [3] => DDDD [5] => EEEE [6] => FFFF)

```
1  <?php
2
3  $numeros = array('cero', 'uno', 'dos', 'tres', 5=>'cinco');
4  print_r($numeros);
5  ?>
6
7
```

```
1  <?php
2
3  $numeros = ['cero', 'uno', 'dos', 'tres', 5=>'cinco'];
4  print_r($numeros);
5  ?>
6
7
```



Array ([0] => cero [1] => uno [2] => dos [3] => tres [5] => cinco)

Exemple array multidimensional

```
1 <?php
2 //creació d'una taula amb les ciutats d'Espanya
3
4 $ciutats_espanya[] = 'Barcelona';
5 $ciutats_espanya[] = 'Madrid';
6 $ciutats_espanya[] = 'Asturies';
7 //enmagatzematge de la taula de ciutats d'Espanya a la taula de ciutats
8
9 $ciutats['ESPANYA'] = $ciutats_espanya;
10
11 //idem amb les ciutats d'Italia
12 $ciutats_italia[] = 'Roma';
13 $ciutats_italia[] = 'Venècia';
14 $ciutats_italia[] = 'Milà';
15
16 $ciutats['ITALIA'] = $ciutats_italia;
17
18 print_r($ciutats);
19
20 ?>
21
```

```
1 <?php
2
3 $ciutats_espanya = array('Barcelona', 'Madrid', 'Asturies');
4 $ciutats_italia = array('Roma', 'Venècia', 'Milà');
5
6 $ciutats= array('ESPANYA'=>$ciutats_espanya, 'ITALIA'=>$ciutats_italia);
7 print_r($ciutats);
8
9 ?>
10
11
```

```
1 <?php
2
3 $ciutats= ['ESPANYA'=> ['Barcelona', 'Madrid', 'Asturies'], 'ITALIA'=> ['Roma', 'Venècia', 'Milà']];
4 print_r($ciutats);
5
6 ?>
```

localhost/primeraWeb/

Array ([ESPANYA] => Array ([0] => Barcelona [1] => Madrid [2] => Asturies) [ITALIA] => Array ([0] => Roma [1] => Venècia [2] => Milà))

4.8.5.2. Accedir a un element de la matriu

```
1  <?php
2
3  echo $numeros['un']; //Si la clau s'ha definit com un string
4  echo $numeros[0]; //Si la clau s'ha definit com un integer
5  echo $ciutats['ESPANYA'][0];
6
7  ?>
```

4.8.5.3. Recórrer una matriu

Es poden utilitzar multitud de mètodes per recórrer una matriu:

- L'estructura de control iterativa **for**
- L'estructura de control iterativa **while**
- L'estructura d'examen de matriu **foreach**

```
foreach (matriu as variable_valor){  
    Instruccions  
}
```

```
foreach (matriu as variable_calve => variable_valor){  
    Instruccions  
}
```

```
1  <?php
2
3  foreach ($numeros as $clau => $numero){
4
5      echo "$numero $clau <br>";
6  }
7
8  ?>
```

```
foreach (matriu as list(variable [,])){  
    Instruccions  
}
```



```
1 <?php  
2 $capitales = [['ESPANYA', 'Madrid'], ['ITALIA', 'ROMA']];  
3 print_r($capitales);  
4 foreach ($capitales as list($pais, $ciudad)){  
5  
6     echo "<br>$pais $ciudad <br>";  
7 }  
8  
9 ?>
```

Array ([0] => Array ([0] => ESPANYA [1] => Madrid) [1] => Array ([0] => ITALIA [1] => ROMA))
ESPANYA Madrid

ITALIA ROMA

A la funció **list** si n'hi han menys variables que elements a la taula anuada, els elements sobrants s'ignoraran. Pel contrari si n'hi ha més variables a la llista **list**, es generarà una alerta i les variables sobrants no s'inicialitzaran.

4.9. Operadors

4.9.1. Operadors d'assignació per valor i per referència

4.9.1.1. Per valor

\$variable1 = expressió;

4.9.1.2. Per referència

\$variable 2 = &\$variable1;

En aquest cas , el valor de la variable1 no es copia a la variable2, sinó que la variable2 fa referència a la variable1. És a dir, totes dues apunten a la mateixa zona de memòria així que la modificació d'una afecta a l'altra.

4.9.2. Operadors aritmètics

Operadores aritméticos		
Ejemplo	Nombre	Resultado
$+ \$a$	Identidad	Conversión de $\$a$ a int o float según el caso.
$- \$a$	Negación	Opuesto de $\$a$.
$\$a + \b	Adición	Suma de $\$a$ y $\$b$.
$\$a - \b	Sustracción	Diferencia de $\$a$ y $\$b$.
$\$a * \b	Multiplicación	Producto de $\$a$ y $\$b$.
$\$a / \b	División	Cociente de $\$a$ y $\$b$.
$\$a \% \b	Módulo	Resto de $\$a$ dividido por $\$b$.
$\$a ** \b	Exponenciación	Resultado de elevar $\$a$ a la potencia $\$b$ ésima. Introducido en PHP 5.6.

4.9.3. Operadors de comparació

Operadores de comparación		
Ejemplo	Nombre	Resultado
$\$a == \b	Igual	TRUE si $\$a$ es igual a $\$b$ después de la manipulación de tipos.
$\$a === \b	Idéntico	TRUE si $\$a$ es igual a $\$b$, y son del mismo tipo.
$\$a != \b	Diferente	TRUE si $\$a$ no es igual a $\$b$ después de la manipulación de tipos.
$\$a <> \b	Diferente	TRUE si $\$a$ no es igual a $\$b$ después de la manipulación de tipos.
$\$a !== \b	No idéntico	TRUE si $\$a$ no es igual a $\$b$, o si no son del mismo tipo.
$\$a < \b	Menor que	TRUE si $\$a$ es estrictamente menor que $\$b$.
$\$a > \b	Mayor que	TRUE si $\$a$ es estrictamente mayor que $\$b$.
$\$a <= \b	Menor o igual que	TRUE si $\$a$ es menor o igual que $\$b$.
$\$a >= \b	Mayor o igual que	TRUE si $\$a$ es mayor o igual que $\$b$.
$\$a <=> \b	Nave espacial	Un integer menor que, igual a, o mayor que cero cuando $\$a$ es respectivamente menor que, igual a, o mayor que $\$b$. Disponible a partir de PHP 7.
$\$a ?? \$b ?? \$c$	Fusión de null	El primer operando de izquierda a derecha que exista y no sea NULL . NULL si no hay valores definidos y no son NULL . Disponible a partir de PHP 7.

4.9.4. Operadors de increment / decrement

Operadores de incremento/decremento		
Ejemplo	Nombre	Efecto
++\$a	Pre-incremento	Incrementa \$a en uno, y luego retorna \$a.
\$a++	Post-incremento	Retorna \$a, y luego incrementa \$a en uno.
--\$a	Pre-decremento	Decrementa \$a en uno, luego retorna \$a.
\$a--	Post-decremento	Retorna \$a, luego decrementa \$a en uno.

4.9.5. Operadors de concatenació d'Strings

L'únic operador d'String és l'operador de concatenació, i és el punt(.)

```

1  <?php
2  $nom = "Olga";
3  $cognom = "Domene";
4  echo "El meu nom és $nom". " i el meu cognom és $cognom";
5
6
7  ?>

```

4.9.6. Operadors lògics

Operadores lógicos		
Ejemplo	Nombre	Resultado
\$a and \$b	And (y)	TRUE si tanto \$a como \$b son TRUE.
\$a or \$b	Or (o inclusivo)	TRUE si cualquiera de \$a o \$b es TRUE.
\$a xor \$b	Xor (o exclusivo)	TRUE si \$a o \$b es TRUE, pero no ambos.
! \$a	Not (no)	TRUE si \$a no es TRUE.
\$a && \$b	And (y)	TRUE si tanto \$a como \$b son TRUE.
\$a \$b	Or (o inclusivo)	TRUE si cualquiera de \$a o \$b es TRUE.

4.9.7. Operadors ternari

Condicció ? Expressió2: Expressió 3

S'executa l'expressió 2 si la condició avalua com a cert, en cas contrari s'executa l'expressió 3.

```
1  <?php
2
3  $pomes = 5;
4  $taronges = 19;
5
6  $major = ( $pomes > $taronges ) ? $pomes : $taronges;
7  echo $major;
8  ?>
9
```

4.10. Estructures de control

4.10.1. Estructura if

Sintaxis

```
if (condicion 1){
    instrucciones_1;
}
elseif (condicion 2){
    instrucciones_2;
}
elseif (condicion _3){
    instrucciones_3;
}
[.....]
else{
    instrucciones_n;
}
```

Exemple

```
1  <?php
2
3  $anys = 38;
4  ▼ if ($anys < 18){
5      echo "Ets menor d'edat";
6  }
7
8  ▼ elseif ($anys >= 18 && $anys < 30){
9      echo "Ets jove";
10 }
11
12 ▼ elseif ($anys >= 30 && $anys < 64){
13     echo "Ets adult";
14 }
15 }
16 ▼ else{
17     echo "Estàs jubilat";
18 }
19
20 ?>
21
```

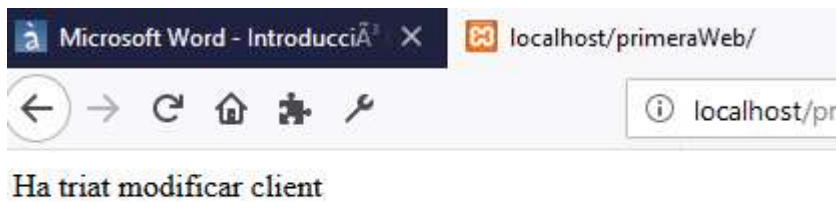
4.10.2. Estructura switch

Sintaxis

```
switch (expressió){
    case expressió 1:
        instrucciones_1;
        [break];
    case expressió 2:
        instrucciones_1;
        [break];
        [...]
        [break];
    instrucciones_n;
}
```

Exemple

```
1  <?php
2
3  $opcio =2;
4  switch ($opcio){
5      case 1:
6          echo "Ha triat insertar client <br>";
7          break;
8      case 2:
9          echo "Ha triat modificar client <br>";
10         break;
11     case 3:
12         echo "Ha triat esborrar client <br>";
13         break;
14     default:
15         echo "opció no valida <br>";
16     }
17 ?>
18
```



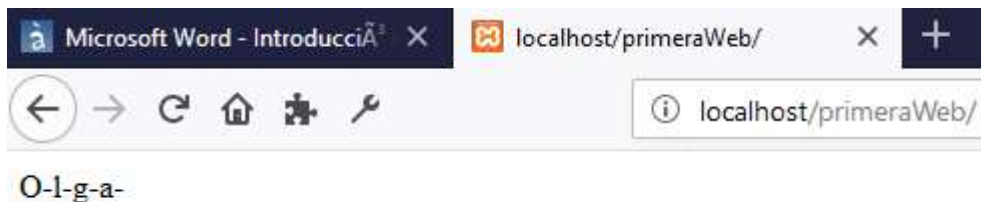
4.10.3. Estructura while

Sintaxis

```
while (condició){
    Instruccions;
}
```

Exemple

```
1  <?php
2
3  $nom = "Olga";
4  $longitud = strlen($nom);
5  $index = 0;
6
7  while ($index < $longitud){
8      echo "$nom[$index]-";
9      $index++;
10
11  }
12  ?>
13
```



4.10.4. Estructura do while

Sintaxis

```
do {  
    Instruccions;  
} while (condició)
```

Exemple

```
1  <?php
2
3  $nom = "Olga";
4  $longitud = strlen($nom);
5  $index = 0;
6
7  do{
8      echo "$nom[$index]-";
9      $index++;
10
11  }while ($index < $longitud)
12  ?>
13
```

4.10.5. Estructura for

Sintaxis

```
for (expressió1; expressió2; expressió3) {  
    Instruccions;  
}
```

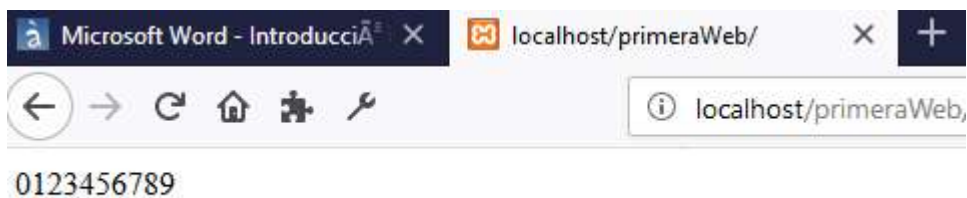
expressió1: s'executa al inici del bucle.

expressió2: s'executa i el resultat s'avalua com a booleà abans de cada iteració. Si aquest resultat s'avalua com a TRUE les instruccions s'executa, si s'avalua com a FALSE, el bucle es para.

expressió3: s'executa al final de cada iteració

Exemple

```
1  <?php  
2  
3  for($n= 0; $n<10; $n++){  
4      echo $n;  
5  }  
6  ?>  
7
```




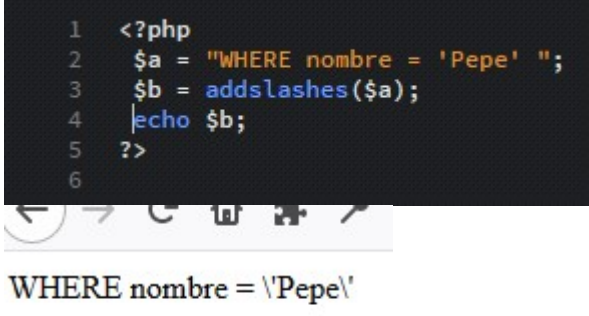


4.11. Utilització de funcions PHP

4.11.1. Manipulació de cadenes

Les funcions més útils per la manipulació de cadenes de caràcters son les següents:

Funció	Descripció	Exemple
strlen	Retorna el nom de caràcters d'una cadena.	<pre> 1 <?php 2 \$x = "HOLA mon"; 3 echo "Mida de x es ".strlen(\$x); 4 ?> 5 </pre> <p>Mida de x es 8</p>
strtolower strtoupper ucfirst ucwords lcfirst	Conversions de majúscules a minúscules.	<pre> 1 <?php 2 \$x = "hola mon"; 3 \$y = "HOLA MON"; 4 echo strtolower(\$y)."
"; 5 echo strtoupper(\$x)."
"; 6 ?> 7 </pre> <p>hola mon HOLA MON</p>
strcmp strcasecmp	Comparació de cadenes. (sensible a majúscules o minúscules o no). Retornen un negatiu si la primera cadena és menor que la segona, un positiu si cadena 1 és major que cadena 2, retorna un 0 si són iguals.	<pre> 1 <?php 2 \$x = "Olga"; 3 \$y = "OLGA"; 4 echo strcmp(\$x,\$y)."
"; 5 echo strcasecmp(\$x, \$y)."
"; 6 ?> 7 </pre> <p>1 0</p>
ltrim rtrim trim	Retorna una cadena amb l'eliminació de caràcters en blanc, o altres caràcters al principi de la cadena, al final o ambdós costats.	<pre> 1 <?php 2 \$x = " hola mundo "; 3 echo trim(\$x)."
"; 4 ?> 5 </pre>

		
nl2br	Retorna un string on transforma els finals de línia propis d'un fitxer de text en propis d'HTML, .	 <pre> 1 <?php 2 \$x = "M'agradaria anar 3 de vacances \n a 4 New York \n amb la meva familia"; 5 echo nl2br(\$x); 6 7 ?> </pre> <p>M'agradaria anar de vacances a New York amb la meva familia</p>
substr	Extracció d'una subcadena d'una cadena	 <pre> 1 <?php 2 \$x = "M'agradaria anar de vacances a New York"; 3 echo substr(\$x, 3)."
"; 4 echo substr(\$x, 3, 8)."
"; 5 ?> </pre> <p>gradaria anar de vacances a New York gradaria</p>
addslashes	Funció que retorna una cadena de caràcters que protegeix amb una \ els caràcters problemàtics.	 <pre> 1 <?php 2 \$a = "WHERE nombre = 'Pepe' "; 3 \$b = addslashes(\$a); 4 echo \$b; 5 ?> </pre> <p>WHERE nombre = \'Pepe\'</p>
stripslashes	Retorna una cadena de caràcters on elimina les \ que protegeixen els caràcters problemàtics.	

explode	Transforma una cadena de caràcters en un array, donada una cadena presa com separador 'elements. Si s'especifica límit, la matriu retorna un màxim d'elements i l'últim contindrà la resta de la cadena.	<pre> 1 <?php 2 \$casa = "salón comedor cocina baño"; 3 \$zonas = explode(" ", \$casa); 4 echo \$zonas[0]; 5 ?> 6 </pre> 
implode	Realitza la operació inversa a explode, és dir, agafa un Array i l'uneix en una cadena de caràcters, afegint el separador indicat.	<pre> 1 <?php 2 \$lenguajes = array("C","Perl","PHP"); 3 \$cadena = implode(";", \$lenguajes); 4 echo \$cadena; 5 ?> 6 </pre> 
strstr	La funció strstr retorna una subcadena de cadena des de la primera aparició de la subcadena fins al final. Si no està la subcadena retorna false	<pre> 1 <?php 2 \$email = 'yo@xtec.cat'; 3 \$dominio = strstr(\$email, '@'); 4 print \$dominio; 5 ?> 6 </pre> 
strpos	realitza la búsqueda d'esquerra a dreta d'una subcadena i retorna la posició.	<pre> 1 <?php 2 \$texto = "cadena de caracteres"; 3 echo strpos(\$texto, "ca"), "
"; 4 echo strpos(\$texto, "ca", 6); 5 ?> 6 </pre> 

4.11.2. Manipulació de matrius

Funció	Descripció	Sintaxi
count	Retorna el número d'elements d'una matriu.	enter count(mixto variable)
in_array	Comprova si un valor està present a una matriu.	boolea in_array(mixto valor_buscat, matriu [,boolea mateix_tipus])
array_search	Busca un valor a una matriu	mixto array_search(mixto valor_buscat, matriu matriu[, boolea mateix_tipus])
[a][k][r]sort	<p>Ordena una matriu (diverses variants)</p> <p>sort: ordre ascendent el valor, sense conservació de les claus.</p> <p>rsort: ordre descendent el valor sense conservació de les claus.</p> <p>asort: ordre ascendent el valor, amb conservació de les claus.</p> <p>arsort: ordre descendent el valor amb conservació de les claus.</p> <p>ksort: ordre ascendent de la clau, amb conservació dels valors.</p> <p>krsort: ordre descendent, amb conservació dels valors.</p>	<p>Boolea [a][k][r]sort (matriu matriu [, enter indicador])</p> <p>Indicador:</p> <p>SORT_REGULAR: Compara elements sense canviar tipus</p> <p>SORT_NUMERIC: compara els elements de manera numèrica.</p> <p>SORT_STRING: compara els elements coma cadenes de caràcters</p> <p>SORT_FLAG_CASE: s'afegeix el valor al valor anterior per no tenir en compte majúscules ni minúscules.</p> <p>....</p>
explode	Divideix una cadena segons el separador i emmagatzema els elements dins d'una matriu.	Ja explicat a les funcions d'String
implode	Re agrupa els elements d'una matriu en una cadena mitjançant un separador.	Ja explicat a les funcions d'String
max	Retorna el valor més alt emmagatzemat a una matriu	Mixte max(matriu matriu)
min	Retorna el valor més petit emmagatzemat a una matriu	Mixte min(matriu matriu)

Existeixen moltes altres funcions que podeu consultar en www.php.net

```
1  <?php
2  $edats = array(5,7,59,36,45,8,0);
3  /*
4   Sintaxis:
5   enter count(mixto variable)
6   */
7  echo "La matriu té ".count($edats)." posicions.<br>";
8  /*
9   Sintaxis:
10  boolea int_array(mixto valor_buscat, matriu [,boolea mateix_tipus])
11  */
12  $trobat = in_array(59, $edats);
13  if($trobat){
14      echo "L'element s'ha trobat";
15  }
16  }else{
17      echo "L'element no es troba a la matriu<br>";
18  }
19  $trobat = in_array("59", $edats, true);
20  if($trobat){
21      echo "L'element s'ha trobat<br>";
22  }
23  }else{
24      echo "L'element no es troba a la matriu<br>";
25  }
26  }
27  /*Sintaxi;
28  mixto array_search(mixto valor_buscat, matriu matriu[, boolea mateix_tipus])
29  */
30
31  echo "El valor $trobat està a la posició ".array_search(59, $edats)."<br>";
32
33  sort($edats);
34  print_r($edats)."<br>";
35
36
37  echo "El valor mínim de l'array es " .min($edats)."<br>";
38  echo "El valor màxim de l'array es " .max($edats)."<br>";
39  ?>
```

4.12. Declaració de funcions

Sintaxi

```
function nombre_funcion([parametres]){
    instruccions;
}
```

Ejemplo

```
1  <?php
2
3  ▼ function producto($valor1, $valor2){
4      return $valor1*$valor2;
5  }
6  //Crida a la funció
7  $resultado = producto(2,4);
8  echo "2 X 4 = $resultado <br>";
9
10 ?>
```

Transformació dels valors d'una matriu en una llista de paràmetres

```
1  <?php
2
3  ▼ function suma($valor1, $valor2, $valor3){
4      return $valor1 + $valor2 + $valor3;
5  }
6  $valores = [1,2,3];
7  echo "1+2+3 = ".suma(...$valores). "<br>";
8
9  ?>
10
```



1+2+3 = 6

Una funció només es pot utilitzar a l'script on es defineix. Per utilitzar-la en varis scripts es necessari bé copiar-la en varis scripts, o bé definir-la en un arxiu i incloure-la on sigui necessari.

Exemple

Arxiu funciones.inc

```
1  <?php
2
3  function suma($valor1, $valor2, $valor3){
4      return $valor1 +$valor2+ $valor3;
5  }
6
7  ?>
```

Arxiu prueba.php

```
1  <?php
2  //Inclusión del archivo de funciones
3  include('funciones.inc');
4  //Utilización de la función
5
6  echo suma(5, 7, 9);
7  ?>
8
```