

3. Referència del llenguatge I

Per escriure els scripts PHP hem d'utilitzar un editor. Aquest pot ser des d'un editor de text pla genèric, com Gedit d'Ubuntu, fins a un IDE, com NetBenas. Per iniciar-vos, que és el que estem fent en aquesta unitat formativa, el millor és utilitzar un editor de pàgines web simple que ens oferirà ajudes per escriure els nostres scripts, cosa que no ens oferirà un editor de text pla genèric i, per un altre costat, no té la complexitat d'un IDE que ara mateix no necessitem. Un editor que compleix aquestes característiques és Brackets.

Brackets és un editor de pàgines web de programari lliure, multiplataforma (disponible per Linux, Windows i MAC) i fàcil d'utilitzar desenvolupat principalment per l'empresa Adobe Systems.

Per treballar amb PHP, un cop instal·lat, és molt aconsellable afegir-li dues extensions:

- **Brackets PHP-Beautify**: aquesta extensió ajuda a mantenir el codi sempre ben formatat, indentat i llegible.
- **Brackets Link File**: quan implementem scripts PHP, a vegades necessitem vincular arxius externs com altres scripts PHP al document actiu, és a dir, a l'script que estem implementant. Aquesta extensió ens facilita aquesta vinculació.

Instal·lació de Brackets

L'enllaç www.brackets.io és el lloc oficial de l'editor Brackets des d'on podreu descarregar-lo per instal·lar-lo.

3.1 Sintaxi

Tenim el fitxer _1Sintaxi.php amb el següent contingut:

```

1 <?php
2
3 //Això és un comentari d'una línia en php.
4 /*Això també és un comentari d'una línia en php*/
5 #Això també és un comentari d'una sola línia.
6
7 /* Això és un comentari
8   PHP de
9   més d'una línia */
10
11 //Integració dins d'un document HTML
12 echo "Això és un missatge escrit amb PHP abans d'escriure el document HTML";
13
14 ?>
15
16 <html>
17
18   <head>
19     <title>Integració php</title>
20     <meta charset="UTF-8"/>
```

```

21   </head>
22
23   <body>
24     <br/>
25
26   <?php
27     //Integració dins del cos del document HTML
28     echo "Això és un missatge escrit amb PHP integrat en el cos del document
29       html";
30   ?>
31
32   <!— Integració de la funció echo dins d'un element HTML—>
33   <p><?= "Això és un missatge echo integrat en el paràgraf del document html
34     ";?></p>
35
36   </body>
37
38 </html>

```

El contingut del fitxer _1Sintaxi.php està format per tres scripts PHP, els que podem veure en la figura 8.1:

FIGURA 3.1. Scripts de l'exemple

```

<?php
//Això és un comentari d'una línia en php
/*Això també és un comentari d'una línia en php*/
#Això també és un comentari d'una sola línia.

/* Això és un comentari
   PHP de
   més d'una línia */

//Integració dins d'un document HTML
echo "Això és un missatge escrit amb PHP abans d'escriure el document HTML";
?>

<html>
  <head>
    <title>Integració php</title>
    <meta charset="UTF-8"/>
  </head>
  <body>
    <br />
    <?php
      //Integració dins del cos del document HTML
      echo "Això és un missatge escrit amb PHP integrat en el cos del document html";
    ?>
    <!— Integració de la funció echo dins d'un element HTML—>
    <p><?= "Això és un missatge echo integrat en el paràgraf del document html";?></p>
  </body>

```

The code is annotated with three red boxes labeled "Script PHP" highlighting the following sections:

- The first box highlights the multi-line comment starting with /* and ending with */.
- The second box highlights the single-line comment starting with #.
- The third box highlights the PHP block within the body of the HTML document, specifically the line echo "Això és un missatge escrit amb PHP integrat en el cos del document html";.

Com podem observar en la figura 8.1, tots els scripts PHP estan escrits dins del bloc `<?php...?>`. Doncs bé, perquè el sistema pugui interpretar el codi PHP, necessàriament s'ha d'escriure dins d'aquest bloc sempre que l'script PHP comparteixi fitxer amb un altre tipus de codi, com en el cas del fitxer _1Sintaxi.php, on els scripts PHP comparteixen fitxer amb codi HTML.

En el cas que els scripts PHP no comparteixin fitxer amb altres llenguatges, no cal l'etiqueta de tancament `?>`, com podem veure en el contingut del següent fitxer _2Sintaxi.php:

```

1 <?php
2
3 echo "FITXER EXTERN PHP";
4
5 //Elements html dins d'un script PHP
6 echo "<br/>";
7
8 echo "Això és un missatge echo del fitxer extern _2integracio.php";

```

3.1.1 Sentències

Les sentències són els **fragments del codi de l'script PHP** que l'intèrpret llegirà fins al final i executarà. En el cas del llenguatge PHP, el final de la sentència ve indicat pel punt i coma (;). Una sentència del fitxer _1Sintaxi.php és:

```
1 echo "Això és un missatge escrit amb PHP abans d'escriure el document HTML";
```

Així doncs, en aquest cas l'intèrpret llegirà la sentència començant per la construcció echo, continuarà amb la cadena de caràcters que imprimirà echo i quan llegeixi el punt i coma (;), executarà la sentència, amb el resultat de la impressió per pantalla mitjançant echo de la cadena de caràcters “*Això és un missatge escrit amb PHP abans d'escriure el document HTML*”.

echo

És una construcció del llenguatge PHP que el que fa és imprimir per pantalla una cadena de caràcters, la que troben a continuació d'echo. Aquesta cadena l'escrivim entre cometes o entre parèntesis.

L'intèrpret de PHP sempre llegirà les sentències **fins a trobar el punt i coma** (;) i les executarà.

L'última sentència del codi d'un script PHP, pot finalitzar sense punt i coma (;), però en aquest cas l'script sempre haurà de finalitzar amb l'etiqueta de tancament ?>, encara que no comparteixi fitxer amb cap altra tipus de llenguatge.

Si modifiquem el contingut del fitxer _2Sintaxi.php eliminant el punt i coma (;) de l'última sentència, el codi resultant serà el següent:

```
1 <?php
2
3 echo "FITXER EXTERN PHP";
4
5 //Elements html dins d'un script PHP
6 echo "<br/>";
7
8 echo "Això és un missatge echo del fitxer extern _2integracio.php"
9
10 ?>
```

3.1.2 Comentaris

Els comentaris en PHP, com en altres llenguatges de programació, serveixen per escriure comentaris referents al codi dels scripts que ajudin a entendre la seva implementació; com per exemple, perquè s'implementa una funcionalitat, o distingir diferents blocs dins de l'script... Així doncs, els comentaris no són codi executable.

És altament aconsellable **incluir comentaris** en els nostres scripts, ja que ens ajudaran a entendre el codi implementat tant a nosaltres, com a altres programadors que també el puguin consultar.

En el fitxer _1Sintaxi.php podem veure que els tres tipus de comentaris més habituals que podem trobar en el llenguatge PHP comencen amb:

- **Dues barres inclinades (//).** Són comentaris d'una sola línia. El comentari l'escrivim després de les dues barres inclinades (//):

```
1 //Això és un comentari d'una línia en php.
```

- **Una barra inclinada i un asterisc (*).** finalitzen amb un asterisc i una barra inclinada (*). El comentari es troba entre els símbols d'inici i finalització /* i */ i poden ser comentaris d'una sola línia (com en el primer cas) o comentaris en bloc (de més d'una línia; com en el segon exemple):

```
1 /*Això també és un comentari d'una línia en php*/
```

```
1 /* Això és un comentari
2   PHP de
3   més d'una línia */
```

- **Un coixinet (#).** Són comentaris d'una sola línia. El comentari l'escrivim després del coixinet (#):

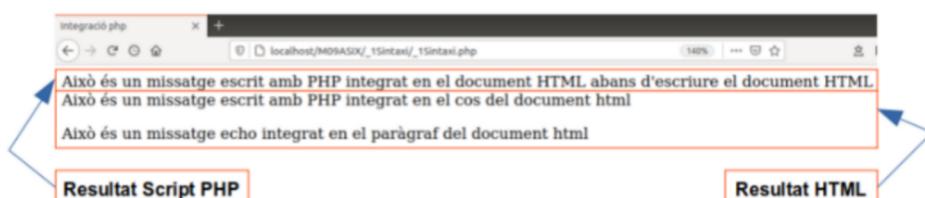
```
1 #Això també és un comentari d'una sola línia.
```

3.1.3 Integració dels scripts PHP dins de documents HTML

Com podem veure en el fitxer _1Sintaxi.php, nosaltres podem fer que scripts PHP comparteixin fitxer amb un document HTML. En aquest cas ens podem trobar en dues situacions:

- Que l'script no estigui integrat dins el document HTML com en el cas del primer script PHP del fitxer _1Sintaxi.php que està escrit abans del codi HTML. En aquest cas el resultat de l'execució d'aquest script PHP no interactua amb el codi HTML, simplement imprimeix per pantalla el missatge “Això és un missatge escrit amb PHP abans d'escriure el document HTML” abans de la impressió del resultat del document HTML, tal com podem veure en la figura 5.3

FIGURA 3.2. Resultat de l'execució del fitxer _1Sintaxi.php



Per tant, quan tenim un script PHP compartint un fitxer amb codi HTML, però aquest no està integrat dins el document HTML, la seva execució no té per què interactuar amb el codi HTML, tant estigui escrit abans o després del codi HTML.

- Que l'script estigui integrat dins el document HTML com en el cas dels altres dos scripts del fitxer _1Sintaxi.php. El resultat de la seva execució serà part del contingut del document HTML. Per integrar un script dins un document HTML, ho podem fer principalment de dues maneres:
 - Escrivint l'script dins el bloc `<?php...?>` com a contingut d'un element del document HTML, com en el cas del segon script del fitxer _1Sintaxi.php que hem escrit l'script com a contingut de l'element `<body>` del document HTML. En aquest cas, podem escriure qualsevol script PHP.
 - Escrivint l'script amb l'estructura del tercer script del fitxer _1Sintaxi.php que hem escrit com a contingut de l'element `<p>` (paràgraf) del document HTML. Aquesta estructura és:

¹ `<?= "contingut"; ?>`

On **contingut** és una cadena de caràcters (string) i el símbol igual (=) equival a la construcció **echo** de PHP. Per tant, aquesta segona manera d'integrar un script en PHP només és vàlida quan el que volem és imprimir per pantalla una cadena de caràcters com ho faríem amb l'estructura echo de PHP.

3.2 Variables

Els noms de les variables en PHP sempre han de començar amb el **símbol de dòlar** (\$), seguit d'una lletra o guió baix (_) i després poden contenir lletres, números i el símbol del guió baix (_). Exemples de noms de variables PHP són: \$inicial, \$_unaAltraVariable i \$_1234_abc.

PHP és “**case sensitive**”, és a dir, que distingeix majúscules de minúscules; per tant, no és el mateix la variable \$unaVariable que \$UnaVariable, ja que el nom de la primera comença per “u” i el segon per “U”.

PHP és un llenguatge no tipificat, això vol dir que quan declarem les variables no cal indicar el tipus de dades que li assignarem com passa amb altres llenguatges com JAVA.

Per fer ús d'una variable en PHP, no cal ni declarar-la ni inicialitzar-la abans del seu ús.

Si volem imprimir per pantalla el contingut d'una variable, ho podem fer mitjançant l'estructura echo, posant entre les cometes o parèntesis el nom de la variable, és a dir, si tenim el següent script:

```

1 <?php
2 $inicial = "Això és el contingut de la variable";
3 echo "$inicial";

```

Per pantalla s'imprimirà “Això és el contingut de la variable”, que és el contingut de la variable \$inicial.

En canvi, si vull imprimir el nom de la variable, perquè l'intèrpret interpreti el símbol del dòlar (\$) com un caràcter ASCII, i no com un caràcter especial del llenguatge PHP, haurem de posar davant el dòlar el símbol (\) per crear una seqüència d'escapament, és a dir, si tenim el següent script:

```

1 <?php
2 $inicial = "Això és el contingut de la variable \$inicial";
3 echo "$inicial";

```

Per pantalla s'imprimirà *Això és el contingut de la variable \$inicial*, que és el contingut de la variable \$inicial i com en el contingut de la variable hem precedit el nom de la variable \$inicial pel símbol (\), s'imprimirà el nom de la variable en lloc del seu contingut.

3.2.1 Àmbits de les variables

Tenim el següent fitxer _2variables.php:

```

1 <?php
2     $inicial = "Això és el contingut de la variable \$inicial en l'script
3         extern a HTML";
4
5     //Imprimim el valor de la variable $inicial
6     echo "$inicial";
7 ?>
8
9 <html>
10    <head>
11        <title>Variables en PHP</title>
12        <meta charset="UTF-8"/>
13    </head>
14
15    <body>
16        <?php
17            /* Imprimim la variable $inicial de l'script extern al document
18               HTML */
19
20            echo "$inicial"; //S'imprimeix el seu valor original
21
22            include "_2Inicial.php";
23        ?>
24    </body>
25
26 </html>

```

Quan executem la sentència PHP **echo “\$inicial”;**, que trobem en l'script que hi ha dins el cos del document HTML, el que s'imprimirà per pantalla és la cadena de caràcters *Això és el contingut de la variable \$inicial en l'script extern a HTML*

que és el contingut de la variable \$inicial definida en l'script extern al document HTML. Això és així perquè les dues variables són la mateixa, encara que s'utilitzin en àmbits diferents, una externa al document HTML i l'altre en el document HTML. Per tant, quan es defineix una variable en un fitxer PHP, aquesta definició és vàlida per tot el fitxer, menys en el cas de les funcions, encara que la variable la utilitzem dins d'àmbits diferents, com per exemple dins un document HTML i fora d'ell.

En l'última sentència de l'script que trobem en el cos del document HTML, mitjançant la funció **include** de PHP, el que fem és incloure el contingut del fitxer _2Inicial.php en el fitxer _2variables.php. Si el contingut del fitxer _2Inicial.php és:

```

1 <?php
2
3 echo $inicial;
```

En el cas de les funcions, no es compleix que les variables definides fora de la funció siguin les mateixes que definim dins la funció, i a la inversa. Això ho veurem més detalladament en l'apartat següent: "Referència del llenguatge II".

El contingut que estem incloent en el fitxer _2variables.php mitjançant la funció **include** concretament és **echo \$inicial;**. Per tant, en executar la sentència **include “_2Inicial.php”**; del fitxer _2variables.php, s'imprimirà per pantalla *Això és el contingut de la variable \$inicial en l'script extern a HTML* que de nou torna a ser el contingut de la variable \$inicial definida en l'script extern al document HTML. Això és així perquè la variable del fitxer extern _2Inicial.php té el mateix nom que la variable definida en l'script extern al document HTML, les variables tornen a ser la mateixa.

Finalment, en el llenguatge PHP existeix un conjunt de variables predefinides. Es tracta de variables definides per aquest llenguatge amb un nom i funcionalitat determinats.

Variables predefinides

Podem trobar la seva definició en el [lloc oficial de PHP](#).

3.3 Tipus de dades

Encara que PHP és un llenguatge no tipificat, existeixen tipus de dades en PHP; els més significatius en PHP són:

- Els **valors escalars**: enter, float, booleà i string (cadena de caràcters).
- Els **tipus compostos**: array i objecte.
- Els **tipus especials**: recurs i null.

En PHP, una variable pot emmagatzemar dades de diferents tipus, així si emmagatzemem a una variable una cadena de text i més endavant li assignem un enter, no donarà error.

Encara que el llenguatge PHP no comprova quin tipus de dada té assignada una variable, existeixen funcions que ens ajuden a comprovar el tipus de dada assignat a una variable. Aquestes funcions són:

PHP no fa distinció entre floats i doubles, com sí que ho fan altres llenguatges de programació.

La separació de la part decimal de la part entera d'un float, es fa mitjançant el punt (.), per exemple: 1.13.

- **is_tipusDades(valor)**. Són un conjunt de funcions amb les quals comprovem si el valor passat per paràmetre és del tipus que indica **tipusDades**. Si és així, la funció retornarà 1, en cas contrari 0. Un exemple de funció d'aquest tipus és `is_int()` (o `is_integer()`), en aquest cas si el valor passat per paràmetre és un enter retornarà 1 i en cas contrari 0.
- **gettype(valor)**. Ens retorna el tipus de dada del valor passat per paràmetre. Per exemple, `gettype(5)` ens retornarà “integer”, ja que 5 és un enter.

3.4 Constants

Per definir una constant a PHP, ho podem fer mitjançant:

- **La paraula reservada const**. Amb aquesta opció, l'estruatura que hem de fer servir és **const nomConstant = valor**, on `nomConstant` és el nom que li donarem a la constant i `valor`, el valor constant que li assignarem. Per exemple, en el codi següent, on el nom de la constant és `SALUTACIO` i el valor assignat “Bon dia!”:

```
1 const SALUTACIO="Bon dia!";
```

- **La funció define(nomConstant, valor)**. On el primer paràmetre, **nomConstant**, és el nom que li donarem a la constant i el segon paràmetre, **valor**, el valor constant que li assignarem a la constant. Per exemple, en el codi següent, on el nom de la constant és `PI` i el valor assignat 3.141592:

```
1 define("PI", 3.141592);
```

Tant en un cas com en l'altre, s'ha de **tenir en compte que**:

- Una vegada que la constant està definida, no podem modificar el seu valor.
- Els valors assignats a les constants només poden ser els valors escalars (boolean, integer, float i string) i arrays
- Per convenció els noms de les constants sempre han de ser amb majúscula.
- Els noms de les constants no poden començar pel símbol del dòlar (\$).

PHP també té constants predefinides i, entre elles, unes constants anomenades **màgiques**. A aquestes constants, se'ls assignen els seus valors quan es compila el fitxer on es fa ús d'elles. Aquests valors, també són predefinitos pel llenguatge. Totes comencen per dos guions baixos (`__`). Un exemple de constant màgica és `__FILE__`, a la qual el valor que si li assignarà un cop compilat el fitxer on en fem ús, serà l'adreça i nom d'aquest fitxer.

Constants predefinides i constants màgiques

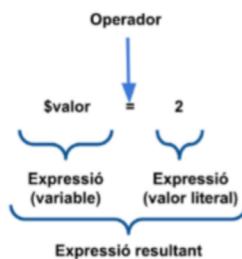
Al web oficial de PHP es poden trobar les [constants predefinides de PHP](#) i les constants màgiques.

3.5 Expressions i operadors

Podem dir que en PHP gairebé tot el que s'escriu és una **expressió**. La manera més simple i encertada de definir el que és una expressió és “qualsevol cosa que té un valor”. Així doncs les formes més bàsiques d'expressions en PHP són les variables i els valors literals o constants com 2, “Hola”, True...

Mitjançant els operadors prenem una o més expressions per retornar un valor. La construcció que sorgeix de la unió dels operadors amb les expressions, és una altra expressió. A la figura 5.4 podem veure un exemple:

FIGURA 3.3. Expressió amb operador



Els operadors es poden agrupar en tres blocs segons el nombre d'expressions sobre les quals operen:

1. Els **operadors unaris** opera sobre una expressió i el que pot fer principalment amb aquesta expressió és negar-la, incrementar-la o decrementar-la.
2. Els **operadors binaris** opera sobre dues expressions i permet assignar-les, sumar-les, restar-les o comparar-les.
3. L'**operador ternari** és un únic operador que permet escollir entre dues expressions en funció d'una altra.

Els operadors amb els quals treballarem els podem classificar en operadors per strings, aritmètics, d'assignació, de comparació, lògics i l'operador ternari.

En el següent script podeu veure com treballar amb els **operadors de cadenes de caràcters (strings)**. L'operador principal és la concatenació (.) que el que fa és unir (concatenar) cadenes de caràcters:

Precedència d'operadors

És important tenir en compte la precedència dels operadors. En el següent enllaç trobareu la [precedència en PHP](#).

```

1 <?php
2
3 /*L'operador concatenació en php és el . */
4
5 $cadena="Cadena";
6 $aixo_es_una="Això és una ";
7

```

```

8 //concatenació dos cadenes
9 echo "Això és una cadena." i això és una altra<br/>";
10
11 //concatenació d'una variable i una cadena
12 echo $cadena." concatenada<br/>";
13
14 //concatenació de cadenes i números de qualsevol tipus
15 echo "Això és ".( 1 )."a cadena<br/>"; //Retorna una cadena
16
17 //concatenació de dues variables
18 echo $aixo_es_una.$cadena;
19
20 echo"<br/>"; //salt de línia
21
22 /*L'assignació sobre concatenació (.=), concatena l'expressió del costat
   dret amb l'expressió assignada a la variable del costat esquerre,
   assignant l'expressió resultant a la variable de l'esquerra.*/
23
24 $aixo_es_una .= $cadena; //Ara el valor de $aixo_es_una és "Això és una
   Cadena"
25 echo $aixo_es_una;

```

El resultat de l'script és el de la figura 5.5:

FIGURA 3.4. Operacions amb strings



En el següent script podeu veure com treballar amb els **operadors aritmètics**:

```

1 <?php
2     $valor1=1.2;
3     $valor2=2.3;
4     $caracter="B";
5
6     //Suma
7     echo "1+2= ".($valor1+$valor2)."<br/>"; //Mostra la suma dels valors de les
       variables
8
9     //Resta
10    echo "1-2= ".($valor1-$valor2)."<br/>"; //Mostra la resta dels valors de les
       variables
11
12    //Producte
13    echo "1*2= ".($valor1*$valor2)."<br/>"; //Mostra el producte dels valors de
       les variables
14
15    //Divisió
16    echo "1/2= ".($valor1/$valor2)."<br/>"; //Mostra la divisió dels valors de
       les variables
17
18    //Mòdul
19    echo "La resta de dividir 2 per 1 és ".($valor2%$valor1); //Mostra el mòdul
       dels valors de les variables

```

Si els operands del mòdul són decimals, abans de l'operació, es convertiran en enters traient la part decimal. Per altra banda, el signe del resultat del mòdul sempre és el signe del dividend.

El resultat de l'script és el de la figura 5.6:

FIGURA 3.5. Operacions aritmètiques

En el següent script podeu veure com treballar amb els **operadors d'increment i decrement**, amb els quals podrem augmentar o disminuir un valor en una unitat:

```

1 <?php
2 //Les operacions d'increment i decrement, permeten augmentar o disminuir en
3 //una unitat un valor determinat
4
5 //Definim les variables amb les quals treballarem
6 $valor1=1.2;
7 $valor2=2.3;
8 $caracter="B";
9
10 //Post-increment: primer retorna el valor de la variable a incrementar i
11 //després fa l'augment modificant el valor original de la variable
12 echo "Primer el retorn: " . $valor1++ . "<br/>"; //s'imprimeix el valor
13 //original de $valor1
14 echo "Després l'augment: " . $valor1 . "<br/>"; //s'imprimeix el valor
15 //original augmentat en una unitat
16
17 //Pre-increment: primer fa l'augment modificant el valor original de la
18 //variable i després retornem el valor augmentat
19 echo "Primer l'augment: " . ++$valor1 . "<br/>"; //s'imprimeix el valor
20 //original de $valor1 augmentat en una unitat
21 echo "Després el retorn: " . $valor1 . "<br/>"; //s'imprimeix el valor
22 //original de $valor1 augmentat en una unitat
23
24 //Post-decrement: primer retorna el valor de la variable a disminuir i despré
25 //s fa la disminució modificant el valor original de la variable
26 echo "Primer el retorn: " . $valor1-- . "<br/>"; //s'imprimeix el valor
27 //original de $valor1
28 echo "Després la disminució: " . $valor1 . "<br/>"; //s'imprimeix el valor
29 //original disminuit en una unitat
30
31 //Pre-decrement: primer fa la disminució modificant el valor original de la
32 //variable i després retornem el valor disminuit
33 echo "Primer la disminució: " . --$valor1 . "<br/>"; //s'imprimeix el valor
34 //original de $valor1 disminuit en una unitat
35 echo "Després el retorn: " . $valor1 . "<br/>"; //s'imprimeix el valor
36 //original de $valor1 disminuit en una unitat
37
38 /*Increment de caràcters transforma el caràcter original assignat a una
39 //variable en el seu successor
40 //i li assigna el nou valor. No es poden decrementar*/
41 $valor=$caracter;
42 echo "++\$caracter val ".(+$caracter)." i \$valor val $valor<br/>"; //Mostra
43 // i $valor conté B

```

El resultat de l'script és el de la figura 5.7:

FIGURA 3.6. Operacions d'increment i decrement

Primer el retorn: 1.2
Després l'augment: 2.2
Primer l'augment: 3.2
Després el retorn: 3.2
Primer el retorn: 3.2
Després la disminució: 2.2
Primer la disminució: 1.2
Després el retorn: 1.2
++\$caracter val C i \$valor val B

En el següent script podeu veure com treballar amb els **operadors d'assignació**. Ja coneixem l'operador d'assignació (=) per assignar un valor a una variable, és a dir, \$variable = valor. L'operador d'assignació es pot combinar amb certes operacions per simplificar les expressions:

```

1 <?php
2
3 /*Tots coneixem l'operador d'assignació per assignar un valor a
4 *una variable, és a dir, $variable = valor.*/
5 $valor1 = 1.2;
6 $valor2 = 2.3;
7 $cadena1 = "Cadena";
8 $cadena2 = " és una cadena";
9
10 /* L'operador d'assignació es pot combinar amb certes operacions per
11 * simplificar les expressions. En tots els casos, l'operació es realitza
12 * entre els valors de les dues expressions i el resultat s'assigna a
13 * l'expressió (variable) de l'esquerra*/
14
15 //Assignació + suma
16 $valor1 += $valor2; //Mateixa que $valor1 = $valor1+$valor2
17 echo "Si \$valor1+=\$valor2, \$valor1 val $valor1 <br/>"; //Mostra 3.5
18
19 //Assignació + resta
20 $valor1 -= $valor2; //Mateixa que $valor1 = $valor1-$valor2
21 echo "Si \$valor1-=\$valor2, \$valor1 val $valor1 <br/>"; //Mostra 1.2
22
23 //Assignació + producte
24 $valor1 *= $valor2; //Mateixa que $valor1 = $valor1*$valor2
25 echo "Si \$valor1*=\$valor2, \$valor1 val $valor1 <br/>"; //Mostra 2.76
26
27 //Assignació + divisió
28 $valor1 /= $valor2; //Mateixa que $valor1 = $valor1/$valor2
29 echo "Si \$valor1=/\$valor2, \$valor1 val $valor1 <br/>"; //Mostra 1.2
30
31 //Inicialitzem variables amb nous valors
32 $valor1 = 1;
33 $valor2 = 2;
34
35 //Assignació + mòdul
36 $valor2% = $valor1; //Mateixa que $valor1 = $valor2%$valor1
37 echo "Si \$valor2%=\$valor1, \$valor2 val $valor2 <br/>"; //Mostra 1.2

```

El resultat de l'script és el de la figura 5.8:

FIGURA 3.7. Operacions d'assignació

Si \$valor1+=\$valor2, \$valor1 val 3.5
Si \$valor1-=\$valor2, \$valor1 val 1.2
Si \$valor1*=\$valor2, \$valor1 val 2.76
Si \$valor1/=\$valor2, \$valor1 val 1.2
Si \$valor2%=\$valor1, \$valor2 val 0
Si \$cadena1.=\$cadena2, \$cadena1 val Cadena és una cadena

En el següent script podeu veure com treballar amb els **operadors de comparació**:

```

1 <?php
2
3 //Compararem els valors assignats a les següents variables
4 $a = 1;
5 $b = 2;
6 $caracterA = "a"; //"a" és equivalent al nombre decimal 97 en el codi ASCII
7 $caracterB = "b"; //"b" és equivalent al nombre decimal 98 en el codi ASCII
8
9 // $a menor que $b
10 echo "\$a < \$b retorna ".( $a < $b )."<br/>"; //Retorna verdader (el navegador
   imprimeix 1)
11
12 // $caracterA menor que $caracterB. Compara el codi ASCII dels caràcters.
13 echo "\$caracterA < \$caracterB retorna ".( $caracterA < $caracterB )."<br/>"; //
   Retorna verdader (el navegador imprimeix 1)
14
15 // $a major que $b
16 echo "\$a > \$b retorna ".( $a > $b )."<br/>"; //Retorna fals (el navegador no
   imprimeix res)
17
18 // $caracterA major que $caracterB
19 echo "\$caracterA > \$caracterB retorna ".( $caracterA > $caracterB )."<br/>"; //
   Retorna fals (el navegador no imprimeix res)
20
21 // $a menor o igual que $b
22 echo "\$a <= \$b retorna ".( $a <= $b )."<br/>"; //Retorna verdader (el
   navegador imprimeix 1)
23
24 // $caracterA menor o igual que $caracterB
25 echo "\$caracterA <= \$caracterB retorna ".( $caracterA <= $caracterB )."<br/>"
   ; //Retorna verdader (el navegador imprimeix 1)
26
27 // $a major o igual que $b
28 echo "\$a >= \$b retorna ".( $a >= $b )."<br/>"; //Retorna fals (el navegador
   no imprimeix res)
29
30 // $caracterA major o igual que $caracterB
31 echo "\$caracterA >= \$caracterB retorna ".( $caracterA >= $caracterB )."<br/>"
   ; //Retorna fals (el navegador no imprimeix res)
32
33 // $a igual que $b
34 echo "\$a == \$b retorna ".( $a == $b )."<br/>"; //Retorna fals (el navegador
   no imprimeix res)
35
36 // $caracterA igual que $caracterB
37 echo "\$caracterA == \$caracterB retorna ".( $caracterA == $caracterB )."<br/>"
   ; //Retorna fals (el navegador no imprimeix res)
38
39 // $a igual que $a i són del mateix tipus
40 echo "\$a === \$a retorna ".( $a === $a )."<br/>"; //Retorna verdader (el
   navegador imprimeix 1)
41
42 // $a igual que "1" i no són del mateix tipus
43 echo "\$a === "1" retorna ".( $a === "1" )."<br/>"; //Retorna fals (el
   navegador no imprimeix res)
44
45 // $a diferent que $b
46 echo "\$a != \$b retorna ".( $a != $b )."<br/>"; //Retorna verdader (el
   navegador imprimeix 1)
47
48 // $caracterA diferent que $caracterB
49 echo "\$caracterA != \$caracterB retorna ".( $caracterA != $caracterB )."<br/>"
   ; //Retorna verdader (el navegador imprimeix 1)
```

El resultat de l'script és el de la figura 3.8:

FIGURA 3.8. Operacions de comparació

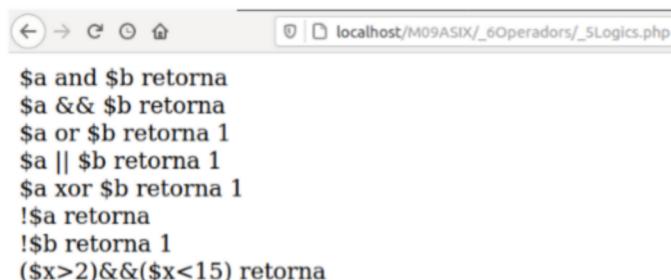
A screenshot of a web browser window. The address bar shows 'localhost/M09ASIX/_6Operadors/_4Comparacio.php'. The page content displays a series of PHP code examples illustrating various comparison and logical operators.

```
$a < $b retorna 1
$caracterA < $caracterB retorna 1
$a > $b retorna
$caracterA > $caracterB retorna
$a <= $b retorna 1
$caracterA <= $caracterB retorna 1
$a >= $b retorna
$caracterA >= $caracterB retorna
$a == $b retorna
$caracterA == $caracterB retorna
$a === $a retorna 1
$a === "1" retorna
$a != $b retorna
$caracterA != $caracterB retorna
```

En el següent script podeu veure com treballar amb els **operadors lògics**:

```
1  <?php
2
3  //Compararem els valors assignats a les variables
4  $a = true;
5  $b = false;
6
7  /*Conjunció: and o && . Retorna verdader si els dos operands són verdaders,
8  si no sempre retorna fals.*/
9  echo "\$a and \$b retorna ".( $a and $b )."<br/>"; //Retorna fals (el navegador
   no imprimeix res)
10 echo "\$a && \$b retorna ".( $a && $b )."<br/>"; //Retorna fals (el navegador
    no imprimeix res)
11
12 /*Disjunció: or o || . Retorna fals si els dos operands són falsos,
13 si no sempre retorna verdader.*/
14 echo "\$a or \$b retorna ".( $a or $b )."<br/>"; //Retorna verdader (el
   navegador imprimeix 1)
15 echo "\$a || \$b retorna ".( $a || $b )."<br/>"; //Retorna verdader (el
   navegador imprimeix 1)
16
17 /*Disjunció exclusiva: xor. Retorna verdader quan un operand és fals
18 i l'altre verdader, sinó sempre retorna fals.*/
19 echo "\$a xor \$b retorna ".($a xor $b)."<br/>"; //Retorna verdader (el
   navegador imprimeix 1)
20
21 /*Negació: !. Retorna verdader quan l'operand és fals i fals quan
22 l'operand és verdader.*/
23 echo "!\$a retorna ".(!$a)."<br/>"; //Retorna fals (el navegador no imprimeix
   res)
24 echo "!\$b retorna ".(!$b)."<br/>"; //Retorna verdader (el navegador imprimeix
   1)
25
26 //Exemple d'expressió amb operadors de comparació i lògics
27 $x = 1; //Nova variable a la qual assignem un valor enter
28 echo "(\$x>2)&&(\$x<15) retorna ".( ( $x>2 ) && ( $x<15 ) ); //Retorna fals (el
   navegador no imprimeix res)
```

El resultat de l'script és el de la figura 3.9:

FIGURA 3.9. Operacions lògiques


The screenshot shows a browser window with the URL `localhost/M09ASIX/_6Operadors/_5Logics.php`. The page displays several PHP code examples for logical operators:

```
$a and $b retorna
$a && $b retorna
$a or $b retorna 1
$a || $b retorna 1
$a xor $b retorna 1
!$a retorna
!$b retorna 1
($x>2)&&($x<15) retorna
```

En el següent script podeu veure com treballar amb l'**operador ternari**, tenint en compte que l'estructura d'aquest operador es **condició ? opció A : opció B**, on si és compleix la **condició** el retorn de l'operació serà l'**opcio A**, i si no l'**opcio B**. La condició serà qualsevol expressió PHP que ens retorni verdader o fals:

```
1 <?php
2
3 /* Operador ternari: si es compleix la condició ( ( $estat == "trist" ) ? )
4   l'opcio vàlida serà la primera ("Estic $estat."), si no la segona
5   ("Em sento bé!!!.")*/
6
7 //Declaració variable
8 $estat = "animat";
9
10 //Realitzem l'operació. El resultat s'assigna a la variable $quinDiaTinc.
11 $quinDiaTinc = ( $estat == "trist" ) ? "Estic $estat." : "Em sento bé!!!.";
12
13 //Mostrem resultat
14 echo $quinDiaTinc; //El resultat és "Em sento bé!!!.", perquè $estat és "animat
  " i no "trist" com la condició.
```

El resultat de l'script és el de la figura 3.10:

FIGURA 3.10. Operador ternari


The screenshot shows a browser window with the URL `localhost/M09ASIX/_6Operadors/_6OperadorTernari.php`. The page displays the output of the PHP script, which is the string "Em sento bé!!!".

3.6 Arrays

Els arrays són estructures que ens permeten **emmagatzemar un conjunt de dades en temps d'execució** (no de manera permanent) de l'script on estan definits. Cadascuna d'aquestes dades té associat un índex que la identifica dins l'array. Les **característiques dels arrays** en PHP són:

- Les dades emmagatzemades en un array poden ser de diferents tipus.
- No cal inicialitzar-los per treballar amb ells.
- No tenen una longitud fixa, si afegim dades aquesta longitud augmentarà, i si les trèiem disminuirà.