

# El llenguatge PHP

---

## Introducció

- Acrònim de *Personal Home Page*
- Llenguatge de propòsit general, encara que el seu fort és el desenvolupament web.
- Sintaxi similar a C / Java
- El codi s'executa en el servidor (en Apache mitjançant *mod\_php*)
- El client rep el resultat generat després d'interpretar el codi al servidor.
- El codi s'emmagatzema en arxiu amb extensió `.php`.

L'última versió és la 8.0, de novembre de 2020 (i en breu tindrem la versió 8.1). La versió 7.0 va sortir al desembre de 2015. A més de nombroses noves funcionalitats que anirem veient durant el curs, té més de dues vegades millor rendiment que PHP5.

!!! important "PHP 7.4" Nosaltres farem ús de la versió 7.4 de PHP

La seva documentació és extensa i està traduïda: <https://www.php.net/manual/es/>.

## Funcionament i estructura bàsica

### El nostre primer codi PHP

- El codi PHP sempre va entre els símbols `<?php` i `?>`.
- Les instruccions PHP acaben sempre amb `;`.
- Per a generar codi HTML des de PHP podem utilitzar el mètode `echo` passant-li el text del codi que volem generar.
- El codi pot anar entre les etiquetes d'HTML.

```
<html>
<head></head>
<body>
    <h1><?php echo "Hola món" ?></h1>
</body>
```

- També podem utilitzar l'etiqueta `<?=` fa el echo al mateix temps.

```
<html>
<head></head>
<body>
    <h1><?= "Hola món" ?></h1>
</body>
```

## Comentaris

- De bloc entre `/* i */`.
- De linea, començant per `//` o per `#`.

## Codi incrustat

El documents que contenen codi PHP s'han d'anomenar amb l'extensió **.php**

## Variables i tipus de dades

Una de les característiques de php és que és un llenguatge **no fortament tipat**. De fet no cal declarar la variable ni indicar el tipus de dades si la declare. Encara que si volem codi de qualitat ho hauriem de fer.

### Declaració

- Els noms de les variables sempre comencen per `$`
- Després del `$` els noms de les variables han d'anar seguits per una lletra o el caràcter `_` i poden contenir també números.
- No és necessari declarar una variable ni especificar-li un tipus (sencer, cadena,...) concret.

Per crear una variable que continga el text a generar i mostrar-la:

```
<html>
<head></head>
<body>
  <h1>
    <?php
      $salutacio = "Hola món";
      echo $salutacio;
    ?>
  </h1>
</body>
```

### Tipus de les variables

- El tipus de la variable es decideix en funció del context en què s'utilitzi.
- En assignar-li el valor 7, la variable és de tipus "sencer"

```
$la_meua_variable = 7; // ara és un número
$la_meua_variable = "set"; // ara és cadena
```

Si li canviem el contingut passa a ser de tipus "cadena"

### Variable no inicialitzades

Si s'intenta utilitzar una variable abans d'assignar-li un valor, es genera un error de tipus **E\_NOTICE** però no s'interromp l'execució de l'script. L'eixida mostrarà un avís cada volta que s'intente.

## Tipus de dades en PHP

- booleà (**boolean**). Els seus possibles valors són true i false. A més, qualsevol nombre enter es considera com true, excepte el 0 que és false.
- sencer (**integer**). Qualsevol nombre sense decimals. Es poden representar en format decimal, octal (començant per un 0), o hexadecimal (començant per 0x).
- real (**float**). Qualsevol nombre amb decimals. Es poden representar també en notació científica.
- cadena (**string**). Conjunts de caràcters delimitats per cometes simples o dobles.
- vector (**array**). Conjunt de variables del mateix tipus ordenades.
- Objecte (**object**). Utilitzat per les instàncies de classes.
- **null**. És un tipus de dades especial, que s'usa per a indicar que la variable no té valor.  
(<http://php.net/manual/es/language.types.null.php>)

## Àmbit de les variables

L'àmbit d'una variable és la part del codi en que és visible. Una variable declarada en un fitxer de PHP està disponible en eixe fitxer i en els que l'incloguen. Les funcions definixen un àmbit local, separat de la resta del codi. Es poden definir variables globals amb la paraula reservada **global**, encara que no són aconsellables.

## Variables predefinides

Són variables internes predefinides de PHP que poden usar-se des de qualsevol àmbit. Tomem forma d'**arrays associatius** que contenen un conjunt de valors.

- **\$\_SERVER**. Conté informació sobre l'entorn del servidor web i d'execució.
- **\$\_GET**, **\$\_POST** i **\$\_COOKIE** contenen les variables que s'han passat al script actual utilitzant, respectivament, els mètodes GET (paràmetres en la URL), HTTP POST i Cookies HTTP
- **\$\_REQUEST** junta en un solament el contingut dels tres \*arrays anteriors, **\$\_GET**, **\$\_POST** i **\$\_COOKIE**.
- **\$\_ENV** conté les variables que es puguin haver passat a PHP des de l'entorn en què s'executa.
- **\$\_FILES** conté els fitxers que es puguin haver pujat al servidor utilitzant el mètode POST.
- **\$\_SESSION** conté les variables de sessió disponibles per al guió actual.

<http://es.php.net/manual/es/language.variables.superglobals.php>

## Constants

Per a definir constants s'utilitza **define()**, que reb el nom de la constant i el valor que li volem donar

```
define("LIMITE",1000);
```

És habitual utilitzar identificadors en majúscules per a les constants.

## Operadors

### Arimètics

Exemple	Nom	Resultat
<code>+\$a</code>	Identitat	Conversió de <code>\$a</code> a <code>int</code> o <code>float</code> segons el cas.
<code>-\$a</code>	Negació	Oposat de <code>\$a</code> .
<code>\$a + \$b</code>	Suma	Suma de <code>\$a</code> i <code>\$b</code> .
<code>\$a - \$b</code>	Resta	Diferència de <code>\$a</code> i <code>\$b</code> .
<code>\$a * \$b</code>	Multiplicació	Producte de <code>\$a</code> i <code>\$b</code> .
<code>\$a / \$b</code>	Divisió	Quocient de <code>\$a</code> i <code>\$b</code> .
<code>\$a % \$b</code>	Mòdul / Residu	Residu de <code>\$a</code> dividit per <code>\$b</code> .
<code>\$a ** \$b</code>	Potència	Resultat de <code>\$a</code> elevat a <code>\$b</code> . PHP >= 5.6.

En el caso de **cadenas**, si queremos concatenarlas, se utiliza el operador `.`:

```
$x = 33;
$y = 11;
$z = $x + $y;
echo "La suma de 33 y 11 es ".44."<br />";
echo "La suma de ".$x." y ".$y." es ".(33 + 11)."<br />";
echo "La suma de ".$x." y ".$y." es ".$z."<br />";
```

Realment en lloc de concatenar cadenas con variables, podem imprimir-les directament ja que s'expandeixen automàticament:

```
echo "La suma de $x y $y es $z <br />";
```

En ocasiones, necesitamos envoltar el nom de la variable entre claus per poder un més text al resultat:

```
$color = "rojo";
echo "El plural de $color el ${color}s";
?>
```

Més endavant estudiarem algunes funcions per al tractament de cadenes.

## Comparació

Exemple	Nom	Resultat
<code>\$a == \$b</code>	Igual	<code>true</code> si <code>\$a</code> és igual a <code>\$b</code> després de la conversió de tipus.

Exemple	Nom	Resultat
<code>\$a === \$b</code>	Idèntic, Comparació estricta	<code>true</code> si <code>\$a</code> és igual a <code>\$b</code> , i són del mateix tipus de dada.
<code>\$a != \$b</code> , <code>\$a &lt;&gt; \$b</code>	Diferent	<code>true</code> si <code>\$a</code> no és igual a <code>\$b</code> després de la conversió de tipus.
<code>\$a !== \$b</code>	No idèntic	<code>true</code> si <code>\$a</code> no és igual a <code>\$b</code> , o si no són del mateix tipus.
<code>\$a &lt; \$b</code>	Menor que	<code>true</code> si <code>\$a</code> és estrictament menor que <code>\$b</code> .
<code>\$a &gt; \$b</code>	Major que	<code>true</code> si <code>\$a</code> és estrictament major que <code>\$b</code> .
<code>\$a &lt;= \$b</code>	Menor o igual que	<code>true</code> si <code>\$a</code> és menor o igual que <code>\$b</code> .
<code>\$a &gt;= \$b</code>	Major o igual que	<code>true</code> si <code>\$a</code> és major o igual que <code>\$b</code> .
<code>\$a &lt;=&gt; \$b</code>	Nau espacial	Torna <code>-1</code> , <code>0</code> o <code>1</code> quan <code>\$a</code> és respectivament menor, igual, o major que <code>\$b</code> . PHP <code>&gt;= 7</code> .
<code>\$a ?? \$b</code> <code>?? \$c</code>	Fusió de <i>null</i>	El primer operano d'esquerra a dreta que existisca i no siga <code>null</code> . <code>null</code> si no hi ha valors definits i no són <code>null</code> . PHP <code>&gt;= 7</code> .

## Lògics

Exemple	Nom	Resultat
<code>\$a and \$b</code> , <code>\$a &amp;&amp; \$b</code>	<i>And</i> (i)	<code>true</code> si tant <code>\$a</code> com <code>\$b</code> són <code>true</code> .
<code>\$a or \$b</code> , <code>\$a    \$b</code>	<i>Or</i> (o inclusiu)	<code>true</code> si qualsevol de <code>\$a</code> o <code>\$b</code> és <code>true</code> .
<code>\$a xor \$b</code>	<i>Xor</i> (o exclusiu)	<code>true</code> si <code>\$a</code> o <code>\$b</code> és <code>true</code> , pero no ambdós.
<code>!\$a</code>	<i>Not</i> (no)	<code>true</code> si <code>\$a</code> no és <code>true</code> .

## Assignació

Exemple	Nom	Resultat
<code>\$a = \$b</code>	Assignació	Assigna a <code>\$a</code> el valor de <code>\$b</code>
<code>\$a += \$b</code>	Assignació de la suma	Le suma a <code>\$a</code> el valor de <code>\$b</code> . Equivalent a <code>\$a = \$a + \$b</code>
<code>\$a -= \$b</code>	Assignació de la resta	Le resta a <code>\$a</code> el valor de <code>\$b</code> . Equivalent a <code>\$a = \$a - \$b</code>
<code>\$a *= \$b</code>	Assignació del producte	Assigna a <code>\$a</code> el producte de <code>\$a</code> por <code>\$b</code> . Equivalent a <code>\$a = \$a * \$b</code>

Exemple	Nom	Resultat
<code>\$a /= \$b</code>	Assignació de la divisió	Assigna a <code>\$a</code> el quocient de <code>\$a</code> entre <code>\$b</code> . Equivalent a <code>\$a = \$a / \$b</code>
<code>\$a %= \$b</code>	Assignació del residu	Assigna a <code>\$a</code> el residu de dividir <code>\$a</code> entre <code>\$b</code> . Equivalent a <code>\$a = \$a % \$b</code>
<code>\$a .= \$b</code>	Concatenació	Concatena a <code>\$a</code> la cadena <code>\$b</code> . Equivalent a <code>\$a = \$a . \$b</code>
<code>\$a++</code>	Increment	Incrementa <code>\$a</code> en una unitat. Equivalent a <code>\$a = \$a + 1</code>
<code>\$a--</code>	Decrement	Decrementa <code>\$a</code> en una unitat. Equivalent a <code>\$a = \$a - 1</code>

!!! Tip "Prioridad de los operadores" Recorda la prioritat. Primer els parèntesis, després la negació (!), productes/divisions, sumes/restes, comparacions, lògics i finalment es realitza l'assignació. Més informació a <https://www.php.net/manual/es/language.operators.precedence.php>

!!! question "Autoevaluación" Si `$a=5` i `$b=4`, esbrina el valor de `$c` si `$c = $a*2 > $b+5 && !($b<>4)`

## Operador ternari

Funciona com un condicional **condició ? valor si true : valor si false** i que es pot simplificar **\*\* condició**

<https://www.php.net/manual/es/language.operators.php>

Farem alguns **exercicis**: [Exercisi 2.0 Conceptes bàsics](#)

## Estructures de control de flux

- Les instruccions de control de flux en PHP funcionen exactament igual que en altres llenguatges de programació.
- Les més habituals són:
  - Condicionals: if, if else, switch
  - Bucles: while, do while, for

seguint les estructures:

```
if (condició) {
    // instruccions
}
else {
    // instruccions
}
```

```
switch ($variable) {
    case valor:
```

```
        //instrucció1
        break;
    case valor:
        //instrucció1
        break;
    default:
        //instrucció1
}
```

```
while (condició) {
    //instruccions
}
```

```
do {
    //instruccions
} while (condició);
```

```
for ($i=1;$i<10;$i++){
    //instruccions
}
```

<http://php.net/manual/es/language.control-structures.php>

## Expansió de variables

- Podem introduir una variable dins d'un text sempre que usem les cometes dobles per a delimitar el text. Açò farà que el contingut de la variable s'expandisca i es concatene amb el text existent en la cadena.

```
echo "<p>Mòdul: $module</p>"
```

- A voltes, és necessari envoltar-la entre claus

```
echo "<p>Mòdul: {$module}DAW</p>"
```

- Si no posàrem les claus l'interpret cercaria una variable que es cride \$moduleDAW

## Maneig de cadenes de text

- Podem utilitzar tant cometes simples com a cometes dobles.

- Per a concatenar cadenes utilitzarem l'operador punt (.)

```
$fullName = $name.' '.$surname;
```

## Funcions de cadena

### !!! todo "Activitat 1"

Crea una pàgina anomenada `cadenes.php`, copia el contingut de la pàgina `index.php`, activa la directiva de tipus estricta (`declare( strict\_types = 1 );`) i mostra el següent:

1. Elimina els espais del principi i el final del nom si els hi haguera (`_trim_`).
2. Elimina la lletra a del principi i el final del nom si els hi haguera (`_trim_`).
3. Mostra la variable nom en majúscules, minúscules i amb la primera lletra en majúscula i les altres en minúscules (`_strtoupper_`, `_strtolower_`, `_ucfirst_`).
4. Mostra el codi ascii de la primera lletra del nom (`_ord_`).
5. Mostra la longitud del nom (`_strlen_`).
6. Mostra el nombre de vegades que apareix la lletra a (majúscula o minúscula, `_substr_count_`).
7. Mostra la posició de la primera `a` existent en el nom, siga majúscula o minúscula (`_strpos_`). Si no hi ha cap mostrarà -1.
8. El mateix, però amb l'última a.
9. Mostra el nom substituint la lletra o pel número zero, siga majúscula o minúscula (`_str_replace_`).
10. Indica si el nom comença per `al` o no.

### !!! todo "Activitat 2"

En el mateix document `cadenes.php` i a partir d'una variable que continga una url:

```
```php
```

```
$url='http://username:password@hostname:9090/path?arg=value#anchor';  
```
```

Utilitza la funció `parse\_url` per a extraure de la url les següents parts:

1. El protocol utilitzat (en l'exemple "http").
2. El nom d'usuari (en l'exemple "username").
3. El path de la url (en l'exemple "/path").
4. El querystring de la url (en l'exemple "arg=value").

## Maneig de dates

Funció `time()`



En PHP les dates s'emmagatzemen com a números enters. La funció `time()` retorna el nombre de segons transcorreguts des de l'1 de gener de 1970 (instant conegut com a època Unix). A aquesta forma d'expressar data i hora se li denomina **timestamp**.

## Funció `date()`

```
date (string $format [, int $timestamp=time()]): string
```

La funció `date` retorna una cadena formatada segons els codi de format. Si no li passem la variable `timestamp` ens retorna la cadena formatada per a la data i l'hora actual.

Els codis de format més habituals per a la funció `date` són:

| CODI | DESCRIPCIÓ                                    |
|------|---|
| a    | am o pm                                       |
| A    | AM o PM                                       |
| d    | Dia del mes amb zeros                         |
| D    | Abreviatura del dia de la setmana (en anglès) |
| F    | Nom del mes (en anglès)                       |
| h    | Hora en format 1-12                           |
| H    | Hora en format 0-23                           |
| i    | Minuts  |
| j    | Dia del mes sense zeros                       |
| l    | Dia de la setmana                             |
| m    | Número de mes (1-12)                          |
| M    | Abreviatura del mes (en anglès)               |
| s    | Segons  |
| y    | Any amb 2 dígit                               |
| Y    | Any amb 4 dígit                               |
| z    | Dia de l'any (1-365)                          |

La informació completa la pots trobar en el [manual oficial de PHP: date](#)

Suposant que hui és 15 de setembre de 2019 i les 19 hores 20 minuts i 23 segons et mostrem alguns exemples:

```
date("d-m-Y"); // 15-09-2019
date("H:i:s"); // 19:20:23
```

```
date("Y"); // 2019
date("YmdHis"); // 20190915192023
date("d/m/y H:i a"); // 15/09/19 19:20 pm
date("d-m-Y H:i", time()); // Moment actual
```

**Funció** `mktime(hora, min, seg, mes, dia, any)`

La funció `mktime` permet obtenir la marca de temps Unix (*timestamp*) d'una data.

Per exemple:

```
$data = mktime(0, 0, 0, 1, 1, 2020);
echo date('d-m-Y', $data); // mostrarà 01-01-2020
```

Més informació en <https://www.php.net/manual/es/function.mktime.php>

Un altra forma de crear dates és mitjançant la funció `strtotime()` on podem indicar les dates mitjançant una cadena de text. Per exemple:

```
// posarem la data en el format any-mes-dia per a evitar confusions.
$date = strtotime("2020-01-01");
```

## Operar amb dates

Com hem dit al principi les dates s'emmagatzem com a enters on cada unitat representa un segon. Així podem sumar i restar dates (*timestamp*) per a afegir, llevar o calcular diferències entre ells.

Per exemple, per poder determinar els dies que falten per a l'1 de gener de 2020 faríem:

```
$data=mktime(0, 0, 0, 1, 1, 2020); // timestamp que representa l'1 de gener
de 2020
$diferencia=$data - time(); // restem a l'1 de gener de 2020 el temps
actual,
                                // el resultat estarà en segons.
$diferenciaEnDies =((( $diferencia / 60) / 60 ) / 24 );
/*
    dividim els segons entre 60 i obtenim els minuts,
    els dividim entre 60 i obtenim les hores,
    els dividim entre 24 i obtenim els dies
*/
```

## Validar dates

En la funció `checkdate` podem validar una data.

```
checkdate (int $month, int $day ,int $year): bool
```

La funció torna **true** si la data es vàlida, si no, torna **false**.

## Classe DateTime

PHP disposa també de la classe **DateTime** per a representar les dates, la veurem més avant.

### !!! todo "Activitat 3: Treballar amb dates"

Crea un fitxer anomenat `dates.php` i realitza les següents tasques:

1. Mostra la data i hora actuals amb el format: ``dd/mm/yyyy hh:mm:ss``
2. Mostra el nom de la zona horària que s'utilitza per defecte.
3. Mostra la data de que serà d'ací 45 dies.
4. Mostra el nombre de dies que han passat des de l'1 de gener.
5. Mostra la data i hora actuals de Nova York.
6. Mostra el dia de la setmana que era l'1 de gener d'enguany.

En acabar penja el document a Aules.

### !!! todo "Activitat 4"

``200dadesPersonals.php``: Escriu un programa que emmagatzeme en variables teu nom, primer cognom, segon cognom, email, any de naixement i telèfon. Després mostra'ls per pantalla dins d'una taula.

```
Nom	Bruce	
Cognoms	Wayne	
Email	batman@dccomics.com	
Any de naixement	1939	
Telèfon	555666777	
```

``206anyos.php``: Després de llegir l'edat d'una persona, mostrar l'edat que tindrà d'aquí a 10 anys i fa 10 anys. A més, mostra quin any serà en cada un dels casos. Finalment, mostra l'any de jubilació suposant que treballaràs fins als 65 anys.

Tip: ``$anyoActual = date("Y")``;

## Maneig d'arrays

Un array és un tipus de dades que ens permet emmagatzemar diversos valors. Per tal d'accedir a un valor utilitzarem una *clau*. Les claus poden ser nombres o textos (*arrays associatius*). Si no indiquem cap clau, a

cada element se li associarà una clau numèrica correlativa.

## Array indexat

### Crear array

- Podem crear un array buit de dues formes:

```
$noms = array();  
$noms = [];
```

- Si volem assignar valors al array en el moment de crear-ho:

```
$noms = array('Sandra', 'Pedro', 'Andrea');  
$noms = ['Sandra', 'Pedro', 'Andrea'];
```

### Mostrar el contingut d'un array

Si intentem mostrar el contingut d'un array amb **echo** no obtindrem les dades que hi ha dins del array. En lloc d'açò podem utilitzar la funció `print_r`:

```
print_r($noms);
```

Però si necessitem donar-li format als continguts del array, haurem de recórrer-ho amb un bucle i anar mostrant element a element.

### Afegint elements al array

Podem afegir elements al array utilitzant claudàtors buits. La clau d'aquests elements serà el següent índex numèric disponible.

```
$noms []='Raul'; $noms []='Marta';
```

### Accedir als elements del array

Accedirem als elements del array indicant la clau de l'element entre claudàtors:

```
echo $noms[2];
```

També podem modificar el seu valor:

```
$noms[0] = 'Sara';
```

## Recórrer arrays

### Arrays associatius

Si el array conté dades diverses i/o ens interessa accedir a ells amb claus més específiques que un simple índex numèric podem utilitzar arrays associatius:

```
$alumne = array('nom' => 'Sara', 'cognom' => 'García', 'edat' => 22 );  
$alumne = ['nom' => 'Sara', 'cognom' => 'García', 'edat' => 22 ];  
$alumne['nom'] = 'Verónica';  
echo $alumne['nom'];
```

## Recórrer arrays associatius

### Recórrer claus i valors

### Arrays multidimensionals

Imagina que volem tenir un array els elements del qual són noms d'esports. Volem dividir-los en esports d'hivern i esports d'estiu. Podem crear un array esports que continga dos elements que al seu torn també seran arrays.

```
$esports = ['hivern' => ['esquí de fons', 'hoquei sobre gel'], 'estiu' => [  
    'natació', 'voley platja'] ];
```

Per a accedir al primer esport d'hivern:

```
echo $esports['hivern'][0];
```

## Funcions d'arrays

Podem obtenir la mida de l'array mitjançant la funció `count(array)`. Per recórrer l'array farem ús d'un bucle `for`:

```
$tam = count($fruites); // mida de l'array
for ($i=0; $i<count($fruites); $i++) {
    echo "Element $ i: $ fruites [$ i] <br />";
}
```

Les operacions més importants que podem realitzar amb *arrays* són:

- `print_r($ array)`: mostra el contingut de tot el `$array`
- `$elem = array_pop($array)`: elimina l'últim `$element`
- `array_push ($ array, $ elem)`: afegeix un `$element` al final.
- `$bool = in_array($elem, $array)`: esbrina si `$elem` està en el `$array`

=== "PHP"

```
```php
$fruits = [ "orange", "pear", "apple"];

array_push ($fruites, "pinapple");
print_r($fruits);

$lastFruit = array_pop($fruits);
if (in_array("pineapple", $fruits)) {
    echo "<p> Pineapple left </ p>";
} else {
    echo "<p> No pineapple left </ p>";
}
print_r($fruites);
?>
```
```

=== "Consola"

```
```Console
Array
(
    [0] => orange
    [1] => pear
    [2] => apple
    [3] => pinapple
)
<p>No pineapple left</ p>
Array
(
    [0] => orange
    [1] => pear
    [2] => apple
)
```
```

- `$claus = array_keys($array)`: torna les claus de l'\$ array associatiu.
- `$tam = count ($array)`: retorna la mida de \$ array.
- `sort($array)`: ordena els elements de l' \$ array.
- `isset($array[element])`: indica si existeix / té valor element dins del array.
- `unset($array[element])`: elimina l'element de l'array (deixa un buit).

=== "PHP"

```
```php
$capitals = array ("Italy" => "Rome",
    "France" => "Paris",
    "Portugal" => "Lisbon");

$countries = array_keys($capitals);
print_r($countries);
sort($countries);
print_r($countries);

unset($capitals["France"]);
print_r($capitals);
```
```

=== "Consola" Console Array ( [0] => Italy [1] => France [2] => Portugal ) Array ( [0] => France [1] => Italy [2] => Portugal ) Array ( [Italy] => Rome [Portugal] => Lisbon )

Existeixen moltíssimes més funcions per treballar amb arrays. Tota la informació en el [documentació oficial](#).

!!! tip "Articles per aprofundir en les operacions amb arrays" \* Un article molt complet (en anglès) de [Com treballar amb arrays en PHP de la manera correcta](<https://code.tutsplus.com/tutorials/working-with-php-arrays-in-the-right-way-cms-28606>). \* Un altre article recomanable (en anglès) és [Com ordenar arrays en PHP](#).

Altres recursos:

- <https://www.studytonight.com/php/indexed-array>
- [https://aprende-web.net/php/php6\\_1.php](https://aprende-web.net/php/php6_1.php)

## Funcions

### Funcions predefinides

Algunes de les funcions predefinides en PHP són les següents, però hi ha moltíssimes més vinculades als diferents mòduls que podem instal·lar.

- `is_null($var)` determina si una variable és nula o no.
- `isset($var)` determina si una variable estan definida i no és NULL.
- `unset($var)` destrueix les variables especificades.

- `empty($var)` torna **true** si no existeix o és **FALSE**
- `is_int($var)`, `is_float()`, `is_bool()`, `is_array()`
- `var_dump($var)`, mostra informació de la variable.

## Funcions definides per l'usuari

Per a crear les teues pròpies funcions, hauràs d'usar la paraula **function**:

```
function suma($a, $b) {
    return $a + $b;
}
```

Per tal d'invocar la funció:

```
$resultat = suma(5, 7);
```

Si una funció no té una sentència **return**, retorna **null** en finalitzar.

!!! important "Command-query separation" Sí bé és un principi de la programació orientada a objectes és útil usar-lo també quan definim funcions. El principi afirma que cada mètode ha de ser un comandament que realitza una acció o una consulta que retorna dades al que la crida, pero no ambdues coses.

## Valors per defecte en els paràmetres

Podem indicar valors per defecte per als paràmetres. Si quan cridem a la funció no indiquem el valor d'un paràmetre es prendrà el valor per defecte indicat.

```
function preuAmbIva($preu, $iva=0.21){
    return $preu * (1 + $iva);
}
$preu = 10;
$preuIva = preuAmbIva($preu);
```

- Pot haver-hi més d'un paràmetre amb valor per defecte, però sempre han d'estar al final.

## Pas de paràmetres per referència

Per defecte els paràmetres es passen per valor. Per a passar un paràmetre per referència afegirem el símbol **&** davant del seu nom.

```
function preuAmbIva(&$preu, $iva=0.18){
    $preu *= (1 + $iva);
}
```



## Declaracions de tipus (Type Hinting)

Les funcions obliguen a que els paràmetres siguin de cert tipus. Si el valor donat és d'un tipus incorrecte, es generarà un error. Per això s'ha d'anteposar-se el nom del tipus al nom del paràmetre. Es pot fer que una declaració accepti valors **NULL** si el valor predeterminat del paràmetre s'estableix a NULL.

### Tipus vàlids

| Type                 | Description  | Version   |
|----------------------|--|-----------|
| Class/interface name | The value must be an instance of the given class or interface.   |           |
| self                 | The value must be an instance of the same class as the one in which the type declaration is used. Can only be used in classes. |           |
| parent               | The value must be an instance of the parent of the class in which the type declaration is used. Can only be used in classes.   |           |
| array                | The value must be an array.  |           |
| callable             | The value must be a valid callable. Cannot be used as a class property type declaration.                                       |           |
| bool                 | The value must be a boolean value.   |           |
| float                | The value must be a floating point number.   |           |
| int                  | The value must be an integer.  |           |
| string               | The value must be a string.  |           |
| iterable             | The value must be either an array or an instance of Traversable.   | PHP 7.1.0 |
| object               | The value must be an object.   | PHP 7.2.0 |
| mixed                | The value can be any value.  | PHP 8.0.0 |

### Exemple

```
function suma(int a, int b):int
{
    return $a + $b;
}
$resultado = suma(3,5);
```

## Funcions com a paràmetres

En PHP és possible passar funcions com a paràmetres a altres funcions. Només cal passar el nom de la funció entre cometes. Exemple:

```
function calculator($operation,$numA,$numB) {  
    return $operation($numA,$numB);  
}  
function sumar($a,$b) { return $a+$b; }  
function restar($a,$b) {return $a-$b; }  
  
$a=4;$b=6;  
echo calculator('sumar',$a,$b);  
echo calculator('restar',$a,$b);
```

## Funcions anònimes (closures)

- Estan implementades usant la classe **Closure**
- Permeten la creació de funcions que no tenen un nom específic
- Podem assignar la funció a una variable o passar-la com a paràmetre a una altra funció.
- Exemple

Sense paràmetres:

```
$anonima = function () {  
    echo "Hola";  
};  
$anonima();
```

Amb paràmetres:

```
$anonima = function ($nom) {  
    echo "Hola {$nom}";  
};  
$anonima('Vicent');
```

## Usar variables de l'àmbit superior

- Una funció anònima pot usar variables de l'àmbit superior mitjançant la paraula reservada **use**:

```
function saluda(callable $fnSaluda) {  
    $fnSaluda('Vicent');  
}  
$salutacio = 'Hola';  
$anonima = function ($nom) use ($salutacio) {  
    echo "{$salutacio} {$nom}";  
};
```

```
};  
saluda($anonima);
```

## Lliberies

Podem fer lliberies de funcions guardant-les en un fitxer que després importem des d'on les utilitzem. Ho podem fer amb include, o utilitzant composer per a fer-ho.

## Noves característiques PHP 7

### Spaceship operator (<=>)

Compara dues expressions \$a i \$b i torna -1 si \$a és menor que \$b, 0 si són iguals i 1 si \$b és major que \$a.

Exemple:

```
// Integers  
echo 1 <=> 1; // 0  
echo 1 <=> 2; // -1  
echo 2 <=> 1; // 1
```

### Operador de fusió de null ??

Torna el primer operand si existeix i no és NULL o el segon operand.

Exemple:

```
// Fetches the value of $_GET['user'] and returns 'nobody'  
// if it does not exist.  
$username = $_GET['user'] ?? 'nobody';  
  
// This is equivalent to:  
$username = isset($_GET['user']) ? $_GET['user'] : 'nobody';
```

## Més recursos:

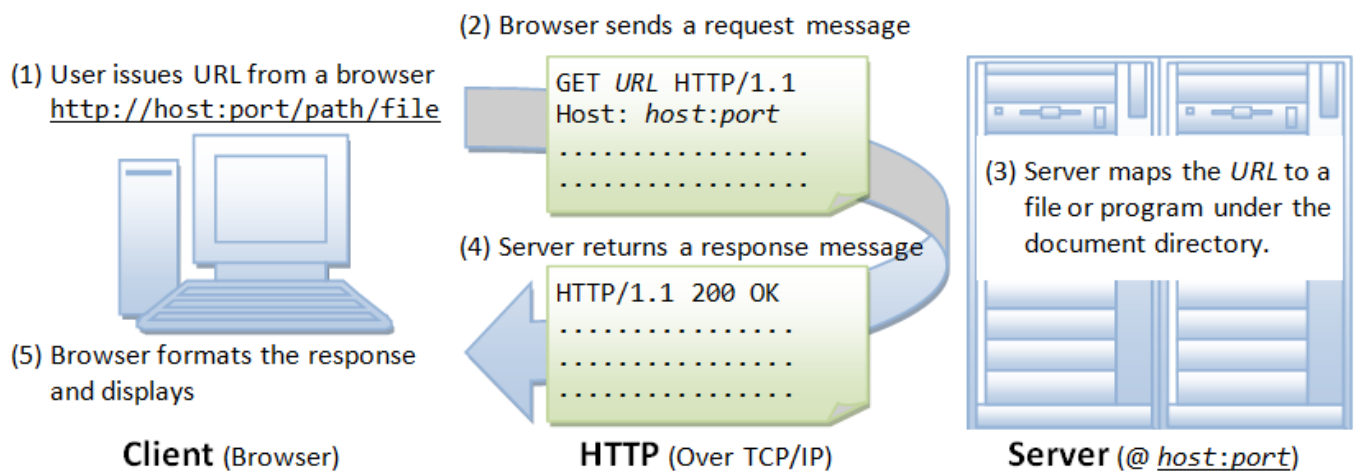
- [Noves característiques PHP 7.0](#)
- [Noves característiques PHP 7.1](#)
- [Noves característiques PHP 7.2](#)
- [Noves característiques PHP 7.3](#)
- [Noves característiques PHP 7.4](#)
- [Noves característiques PHP 8.0](#)

## Processament de formularis

Què és una petició HTTP?

Una petició HTTP és una sol·licitud d'un recurs a un servidor. La petició es realitza a través d'una URL. Amb la petició s'envien també paràmetres.

Hi ha diferents mètodes (METHOD) per a realitzar una petició (GET, POST, PUT, DELETE, PATCH, etc.). Els més habituals són GET i POST. La resta són més utilitzats en les RESTful API.

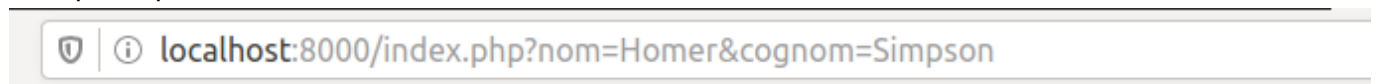


## Mètodes GET i POST

### Mètode GET

S'utilitza per a sol·licitar dades d'un recurs. Mostren els paràmetres que s'envien en la url. Es poden utilitzar directament en enllaços. El resultat es pot emmagatzemar en cache. Romanen en l'historial del navegador. La grandària dels paràmetres està limitat a 255 caràcters.

#### Exemple de petició GET



## Inspecció de la petició

CapçaleresGaletesParàmetresRespostaTempsStack Trace

URL de la sol·licitud: http://localhost:8000/index.php?nom=Homer&cognom=Simpson  
Mètode de la sol·licitud: GET  
Adreça remota: 127.0.0.1:8000  
Codi d'estat: 200 OK ⓘ  
Versió: HTTP/1.1

Edita i torna a enviar

Filtra capçaleres

Capçaleres de la resposta (180 B)Capçaleres sense processar

ⓘ Connection: close  
 ⓘ Content-type: text/html; charset=UTF-8  
 ⓘ Date: Tue, 24 Mar 2020 16:42:47 GMT  
 ⓘ Host: localhost:8000  
 X-Powered-By: PHP/7.2.24-0ubuntu0.19.04.2

Capçaleres de la sol·licitud (487 B)Capçaleres sense processar

ⓘ Accept: text/html,application/xhtml+xml;q=0.9,image/webp,\*/\*;q=0.8  
 ⓘ Accept-Encoding: gzip, deflate  
 ⓘ Accept-Language: ca,en-US;q=0.7,en;q=0.3  
 ⓘ Cache-Control: max-age=0  
 ⓘ Connection: keep-alive  
 ⓘ Cookie: XDEBUG\_SESSION=XDEBUG\_ECLIPSE;...ID=mn6i0cljn6l2volav7jh4jvtol  
 ⓘ DNT: 1  
 ⓘ Host: localhost:8000  
 ⓘ Upgrade-Insecure-Requests: 1  
 ⓘ User-Agent: Mozilla/5.0 (X11; Ubuntu; Linu...) Gecko/20100101 Firefox/72.0

CapçaleresGaletesParàmetresRespostaTempsStack Trace

Filtra paràmetres de la sol·licitud

Cadena de consulta

nom: Homer  
cognom: Simpson

## Inspecció de la petició

### Accedir a les dades de la petició GET

Per a accedir a les dades usem la variable superglobal `$_GET`.

`$_GET` és un array associatiu les claus del qual coincidiran amb els noms que li hem donat als paràmetres.

Per a accedir als paràmetres de la petició anterior:

```
echo $_GET['nom'].' '.$_GET['cognom']; // Homer Simpson
```

### Mètode POST

El mètode POST té les següents característiques:

- S'utilitza per a enviar dades a un recurs.
- Els paràmetres van en el cos de la petició, no són visibles per a l'usuari.
- La petició no es guarda en cache.
- No es pot utilitzar en un enllaç.
- No roman en l'historial.
- No tenim la limitació de grandària dels paràmetres.
- Es solen utilitzar en els formularis.

### Evitar el CSRF

En tota pàgina que reba paràmetres GET has de comprovar el **HTTP referer** del navegador, i que aquest siga de dins de la teua web. En php el referer que envia el navegador s'emmagatzema en

```
$_SERVER['HTTP_REFERER'].
```

Seria tal com:

```
if(parse_url($_SERVER['HTTP_REFERER'],
PHP_URL_HOST) != $_SERVER['HTTP_HOST'])
    die('Anti-CSRF');
```

!!! important Amb aquest codi estem obligant al fet que el navegador envie un *referer* si o sí. Per tant només ha d'utilitzar-se en pàgines a les quals el navegador accedisca des d'una altra pàgina de la nostra web.

Òbviament no podem col·locar-ho en la primera pàgina a la qual s'accedeix a la nostra web (`index.php` o similar), ja que si l'usuari a escrit l'adreça a mà en la barra del navegador no s'enviarà referer cap i saltarà el sistema.

### Definició de formularis

Com hem dit abans el mètode POST s'empra en els formularis. El formulari següent enviarà les dades a la pàgina `index.php` (atribut `action` de l'element `form`).

Utilitza el mètode `post` indicat en l'atribut `method`:

```
<form action="index.php" method="post">
  <label for="nom">Nom</label>
  <input type="text" name="nom" value="">
  <br />
  <label for="cognom">Cognom</label>
  <input type="text" name="cognom" value="">
  <br />
  <input type="submit" value="Enviar">
</form>
```

## Inspecció de la petició: Headers

Capçaleres

Galetes

Paràmetres

Resposta

Temps

URL de la sol·licitud: http://localhost:8000/index.php  
Mètode de la sol·licitud: POST  
Adreça remota: 127.0.0.1:8000  
Codi d'estat: 200 OK ⓘ  
Versió: HTTP/1.1  
Referrer Policy: no-referrer-when-downgrade

Edita i torna a enviar

Filtra capçaleres

▼ Capçaleres de la resposta (180 B)

Capçaleres sense processar

ⓘ Connection: close  
ⓘ Content-type: text/html; charset=UTF-8  
ⓘ Date: Tue, 24 Mar 2020 16:50:48 GMT  
ⓘ Host: localhost:8000  
X-Powered-By: PHP/7.2.24-0ubuntu0.19.04.2

▼ Capçaleres de la sol·licitud (579 B)

Capçaleres sense processar

ⓘ Accept: text/html,application/xhtml+xml;q=0.9,image/webp,\*/\*;q=0.8  
ⓘ Accept-Encoding: gzip, deflate  
ⓘ Accept-Language: ca,en-US;q=0.7,en;q=0.3  
ⓘ Connection: keep-alive  
ⓘ Content-Length: 24  
ⓘ Content-Type: application/x-www-form-urlencoded  
ⓘ Cookie: XDEBUG\_SESSION=XDEBUG\_ECLIPSE;...ID=mn6i0cljn6l2volav7jh4jvtol  
ⓘ DNT: 1  
ⓘ Host: localhost:8000

## Inspecció de la petició: Paramètres

Capçaleres

Galetes

Paràmetres

Resposta

Temps

Filtra paràmetres de la sol·licitud

▼ Dades del formulari

nom: Homer  
cognom: Simpson

▼ Càrrega útil de la sol·licitud

1 nom=Homer&cognom=Simpson

## Accedir a les dades de la petició POST

Useu la variable superglobal `$_POST`. Funciona igual que `$_GET`, però amb els noms (atribut `name`) que li hem donat als camps del formulari.

En depuració podem mostrar totes les dades rebudes:

```
var_dump($_POST);  
print_r($_POST)
```

Per a mostrar les dades individualment:

```
echo $_POST['nom'];  
echo $_POST['cognom'];
```

## Accedir a paràmetres no existents

Errors del tipus

```
Notice: Undefined index: nom in /home/ubuntu/index.php on line 6
```

ens indiquen que la clau `nom` no existeix en l'array `$_POST`. La variable superglobal `$_POST` està buida si no s'ha enviat el formulari.

Per a evitar aquest tipus d'errors és important verificar que s'haja enviat el formulari prèviament:

```
if ($_SERVER['REQUEST_METHOD'] === 'POST')  
{  
  
}
```

## Validació de formularis

Hem de comprovar que les dades que envia el formulari són correctes.

Les validacions a realitzar són les següents:

1. Els camps requerits no han de quedar buits. Ni contenir espais en blanc a l'inici i al final.
2. Els camps email i data han de tenir el format esperat.
3. Tots els camps s'han de filtrar amb `htmlspecialchars` per a evitar atacs de *Cross-site Scripting* (XSS).

### Valors buits

1. Els camps requerits no deurien quedar-se buits.
2. Per a verificar que un valor no queda buit podem utilitzar la funció `empty()` de PHP.

### Espais en blanc

1. Hem d'eliminar els espais en blanc del principi i final dels camps.
2. S'utilitza la funció `trim`

### Escapar l'entrada



Sempre hem de filtrar l'entrada amb `htmlspecialchars` abans de mostrar el camp amb `echo` o similar. Acò convertirà qualsevol caràcter especial d'html en la entitat, així no interferirà en el programa.

## Comprovar l'email

Per a verificar si un email és correcte podem utilitzar la funció `filter_var`

```
filter_var($email, FILTER_VALIDATE_EMAIL)
```

o la funció `filter_input` quan obtenim les dades directament d'una variable extern.

```
// suposem que rebem les dades d'un formulari que té un quadre de text de  
nom 'email'  
$email = filter_input(INPUT_POST, 'email', FILTER_VALIDATE_EMAIL)
```

## filter\_input

```
filter_input(int $type, string $variable_name[, int  
$filter=FILTER_DEFAULT[, mixed $options]]):mixed
```

La funció `filter_input` agafa una variable externa (`$_GET`, `$_POST`, etc) concreta pel seu nom i aplica el filtre indicat.

Els filtres poden [sanejar](#) o [validar](#) les variables externes.

Per exemple, si volem agafar el valor del paràmetre *nom* del querystring (`http://localhost/index.php?nom=`

# Homer

---

) usarem el tipus `INPUT_GET`.

El filtre `FILTER_SANITIZE_STRING` elimina etiquetes, i opcionalment elimina o codifica caràcters especials.

```
$nom = filter_input(INPUT_GET, 'nom', FILTER_SANITIZE_STRING). // $nom =  
Homer
```

Més informació en:

- [filter\\_var](#)
- [filter\\_input](#)

Comprovar la data

- Per a comprovar la data hem de crear una funció a aquest efecte.
- Podem utilitzar el mètode estàtic `createFromFormat` de la classe `DateTime`

Més informació: <http://php.net/manual/es/datetime.createfromformat.php>

## Exemple

```
$data1 = "2001-05-02"; // Data és de tipus string
// DateTime::createFromFormat converteix una cadena de text a on objecte
DateTime
// ens tornarà una instància de DateTime o FALSE en cas d'error.

$dt1 = DateTime::createFromFormat('Y-m-d', $data1);

if ($dt1 === false) {
    echo "La data d'inici és incorrecta";
}
```

<https://www.php.net/manual/es/function.strptime.php>

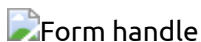
També és habitual trobar els controls de data separats en 3 elements input. Un per a l'any, l'altre per a mes i l'altre per al dia.

## Bones pràctiques en l'obtenció de dades des de l'exterior

Cal seguir les següents bones pràctiques:

1. No confieu mai (mai) en l'entrada des de l'exterior del vostre PHP.
2. Sanegeu i valideu l'entrada de dades sempre.
3. Les funcions `filter_var()` i `filter_input()` poden sanejar el text i validar els formats de text (per exemple, adreces de correu electrònic, enters).
4. Recordeu que l'entrada de dades no es limita a formularis enviats per l'usuari. Els fitxers carregats i descarregats, els valors de sessió, les dades de galetes i les dades de serveis web de tercers també són d'entrada estrangera.

A mode de resum podíem resumir la gestió de formularis en el següent diagrama de flux.



## Formularis en HTML5

En els següents recursos trobaràs informació addicional sobre els controls de formularis en HTML5:

- [Formularios en HTML](#) en MDN web docs.
- [Formularios en HTML5](#) en MDN web docs.
- [HTML Forms](#) en W3CSchools.

## Pujada de fitxers

Per a pujar fitxers PHP implementa un mecanisme senzill a través de la variable superglobal `$_FILES`.

En [Pujada d'arxius](#) disposeu de tota la informació necessària per a gestionar la pujada d'arxius.

## Formulari

Per poder utilitzar el tipus `file` en l'element `input` cal que el formulari incloga l'atribut `enctype` amb el valor `multipart/form-data`.

El valor de l'atribut `name` de l'element `input` serà l'índex de l'array associatiu `$_FILES` que ens permetrà obtenir tota la informació del procés de pujada del fitxer.

En el següent exemple:

```
<form action="upload.php" enctype="multipart/form-data" method="POST">
  <input type="hidden" name="MAX_FILE_SIZE" value="10240">
  <input type="file" name="image" />
  <input type="submit" value="Upload" />
</form>
```

Emprarem `$_FILES['image']` per obtenir les dades de l'arxiu penjat.

## Variable superglobal `$_FILES`

Cada element en `$_FILES` és un array que aporta informació sobre el fitxer pujat. Les claus més importants són:

- `name`. El nom original del fitxer pujat. No és massa útil perquè el sistema original pot tindre convencions diferents i pot generar col·lisions si l'utilitzem per a emmagatzemar-lo en la seua ubicació definitiva.
- `type`. El tipus MIME del fitxer deduït pel navegador.
- `size`. La grandària en bytes del fitxer. Si el fitxer és massa gran el valor enviat serà 0.
- `tmp_name`. El nom temporal del fitxer en el servidor on s'ha emmagatzemat el fitxer pujat.

## Moure el fitxer

Com que el fitxer pujat es guarda en una carpeta temporal hem d'emprar la funció `move_uploaded_file` per a guardar-lo en la ubicació definitiva.

La funció `is_uploaded_file` ens permet assegurar-nos que el fitxer ha estat pujat usant HTTP POST i no es tracta d'un fitxer maliciós.

`move_uploaded_file` ja fa eixa comprovació en executar-se.

## Gestió d'errors

PHP torna un codi d'error en `$_FILES`. El codi es pot tragar en la clau `error`. Per exemple: `$_FILES['image']['error']`.

Els missatges d'error més importants són:

- `UPLOAD_ERR_OK`: La pujada ha sigut correcta.

- `UPLOAD_ERR_INI_SIZE`: La grandària del fitxer que s'intenta pujar és major que el valor indicat en la directiva `upload_max_filesize`.
- `UPLOAD_ERR_FORM_SIZE`: La grandària del fitxer que s'intenta pujar és major que el valor indicat en el formulari en `max_file_size`.
- `UPLOAD_ERR_NO_FILE`: No s'ha enviat cap fitxer.

En [Explicació dels missatges d'error](#) teniu més informació.

## Sentències per a incloure Fitxers

Les sentències **`include()`** i **`include_once()`** i **`require()`** i **`require_once()`** inclouen i avaluen l'arxiu especificat. **`include_once()`** i **`require_once()`** verifica que l'arxiu no haja sigut inclòs abans i és preferible a `include`. Cal ser curòs amb el path de l'arxiu a incloure. La diferència entre `require` i `include` és el tractament de l'error quan el fitxer no existeix.

=== "fruits.php"

```
<?php
$color = 'green';
$fruta = 'apple';
include('fruit.view.php')
```

=== "fruits.view.php"

```
<html>
<head>
<title>Fruites</title>
</head>
<body>
  <h3>
    <?= "Una $fruta $color" ?>
  </h3>
</body>
</html>
```

## Activitats

### Arrays indexats

231. `231Arrays.php`: Crea una pàgina, copia el contingut de la pàgina `index.php` i resol els exercicis següents utilitzant funcions d'arrays:

1. Crea un array amb els noms de diversos alumnes de la classe incloent el teu.
2. Mostra el nombre d'elements que té l'array (`count`).

3. Crea una cadena de text que continga els noms dels alumnes existents en l'array separats per un espai i mostra-la (*implode*).
4. Mostra l'array en un ordre aleatori diferent al que ho vas crear (*shuffle*).
5. Mostra l'array ordenat alfabèticament (*sort*).
6. Mostra els alumnes el nom dels quals continga almenys una "a" (*array\_filter*).
7. Mostra l'array en l'ordre invers al que es va crear (*rsort*).
8. Mostra la posició que té el teu nom en l'array (*array\_search*).

232. [232Ciutats.php](#): Segons l'INE les 7 ciutats més grans d'Espanya (per habitants) el 2018 van ser les següents:

- Madrid, MAD, 3.223.334
- Sevilla, AN, 688.711
- Murcia, MU, 447.182
- Málaga, AN, 571.026
- Zaragoza, AR, 666.880
- València, CV, 791.413
- Barcelona, CAT, 1.620.343

Copia `index.php` i crea un nou document `ciutats.php`. Defineix un array que continga aquesta informació sobre ciutats i

habitants. Imprimeix una taula d'ubicacions i habitants que incloga la població total de les 7 ciutats.

Opcional:  
Modifica la solució de l'anterior exercici perquè mostre les ciutats ordenades per habitants. Després mostra-les per ordre alfabètic.

## Arrays multidimensionals

233. [233Alumnes.php](#): Crea una pàgina anomenada [arrays2.php](#), copia el contingut de la pàgina `index.php` i resol els exercicis següents utilitzant funcions d'arrays:

1. Crea un array d'alumnes on cada element siga un altre array que continga nom i edat de l'alumne.
2. Crea una taula HTML en la qual es mostren totes les dades dels alumnes.
3. Utilitza la funció `array_column` per a obtenir un array indexat que continga únicament els noms dels alumnes i mostra'ls per pantalla.
4. Crea un array amb 10 números :
  - utilitza la funció `array_sum` per a obtenir la suma dels 10 nombres.

234. [234CiutatsOpcional.php](#): Modifica la solució del exercici [ciutats.php](#) perquè la taula continga també la columna del total d'habitants de la comunitat autònoma de les ciutats de la llista i el percentatge sobre els habitants de la comunitat autònoma que representa.

Per exemple:

```
| Ciutat | Habitants | Habitats CA | % sobre CA |  
| -- | --: | --: | --: |  
| València | 791.413 | 5.003.769 | 15.81% |
```

Pista: Caldrà modificar l'array ``$ciutats`` i convertir-lo en multidimensional. Les dades de comunitats autònomes hauran d'estar en un altre array.

Dades: [Municipis de España]  
([https://es.wikipedia.org/wiki/Anexo:Municipios\\_de\\_Espa%C3%B1a\\_por\\_poblaci%C3%B3n](https://es.wikipedia.org/wiki/Anexo:Municipios_de_Espa%C3%B1a_por_poblaci%C3%B3n))

## Funcions

241. `241funcions.php`: Escriu una funció per retornar una etiqueta HTML `<img />`.

La funció hauria d'acceptar com a argument obligatori l'URL de la imatge i arguments opcionals per a un text alternatiu, alçada i amplada.

242. `242funcions.php`: Copieu la funció de l'exercici anterior i modifiqueu-la de manera que només es passe el nom de fitxer a la funció en lloc de l'URL completa. Dins de la funció, farem ús d'una variable global per fer l'URL completa.

Per exemple, si passem ``photo.png`` a la funció, i la variable global conté ``/images``, llavors l'atribut ``src`` de

l'etiqueta retornada serà `/images/photo.png`.

Una funció com aquesta és una forma senzilla de mantenir correctes les vostres etiquetes d'imatges, fins i tot si les imatges es mouen a un nou camí o servidor. Només cal canviar la variable global, per exemple, de ``/images`` a ``http://images.example.com/.``

243. `243funciocolors.php`: Els colors web com `#ffffff` i `#cc3399` es realitzen concatenant els valors hexadecimals de color per a vermell, verd i blau.

Escriu una funció que accepti 3 arguments: roig, verd i blau, i que retorne un string que conté el color adequat per

utilitzar-lo en una pàgina web.

Per exemple, si els arguments són 255, 0, i 255, llavors la cadena retornada hauria de ser #FF00FF.

Pot resultar útil utilitzeu la funció ``dechex()`` integrada, que es troba documentada a [\[http://www.php.net/\]](http://www.php.net/) (<http://www.php.net/>)

Assegureu-vos que els paràmetres reben valors enters i que són colors vàlids.

Implementa 3 exemples d'ús.

#### 244. [244funcionsSQL.php](#): Crea una funció anomenada `insert` que ens genere una sentència INSERT INTO en SQL.

Per a açò la funció rebrà dos paràmetres:

1. El nom de la taula
2. Un array associatiu que contindrà els noms i valors dels camps de la taula.

La sentència resultant tindrà la següent forma:

```
...
```

```
"INSERT INTO nom_taula (nom dels camps separats per comes)
  VALUES (noms dels camps separats per comes amb el caràcter ":" davant)
..."
```

De moment, no farem res amb els valors dels camps.

Ajuda: utilitza les funcions ``sprintf``, ``implode`` i ``array_keys``

#### 245. [245funcionsSQL2.php](#): A partir de l'exercici anterior crea una altra funció que reba els mateixos paràmetres més un paràmetre booleà per a indicar si volem generar la query amb els noms dels camps o no.

El paràmetre tindrà el valor ``true`` per defecte.

Si el seu valor és ``true`` generarà la consulta igual que en l'exercici anterior, però si és ``false`` la generarà així:

```
```sql
INSERT INTO nom_taula
  VALUES (valors dels camps separats per comes amb el caràcter ':' davant)
```
```

#### 246. [246funcionsSQLReferencia.php](#): Repeteix l'exercici anterior amb els següents canvis:

La cadena resultant es passarà per referència.

Passarem la cadena de la següent forma:

```
```sql
INSERT INTO taula (camps) VALUES (valors)
```
```

Dins de la funció substituïrem el següent:

1. El text taula pel nom de la taula.
2. El text camps pels noms dels camps separats per comes
3. El text valors pels noms dels camps separats per comes i el caràcter `:` davant.

## Formularis

261. [261ExempleGet.php](#): Crea una pàgina que reba com a paràmetre un nom i mostre el text 'Benvingut [nom]!!!' sent [nom] el nom has passat com a paràmetre.

262. [262Formulari.php](#): Crea un formulari que tinga els següents camps:

- Nom i cognoms
- Data de naixement
- Correu electrònic
- Assumpte
- Missatge

En l'atribut `action` del formulari posarem el següent:

```
```html
<form action="262Formulari.php" .../>
```
```

Açò farà que siga la pròpia pàgina del formulari la que processe les dades del mateix.

En prémer `Enviar` han d'aparèixer sota el formulari les dades que s'han introduït en el mateix.

263. [263FormulariReparat.php](#): Soluciona el problema dels paràmetres no enviats de l'exercici anterior.

264. [264FormulariValidat.php](#): Modifica l'exercici anterior realitzant les validacions oportunes.

## Pujada de fitxers



271. `271Formularimatge.php`: Modifica l'activitat `264FormulariValidat.php` afegint un camp de tipus `FILE` per a pujar una imatge al servidor. Es guardarà en la carpeta `uploads` i es mostrarà amb la resta de dades.

## Crèdits

- Aitor Medrano. (setembre de 2021) *Desarrollo Web en Entorno Servidor* disponible en: <https://aitor-medrano.github.io/dwes2122/index.html>