Resum del curs de PHP

Resum del curs

1. Què hem de recordar

- El PHP genera HTML
- Podem mesclar PHP i HTML:
    - El codi php va entre `<?php ?>`
    - Si volem treure HTML ho hem de fer amb `echo` o `print`
- Podem fer servir `include` per no repetir codi
- La connexió a la base de dades sols s'hauria de fer un cop
- Alerta amb els paths relatius i absoluts
- En desenvolupament activau que us mostri els errors

2. Comentaris

```php
<?php
/*
Això és un comentari llarg
i pot tenir més d'una línea.
*/

// Això és un comentari d'una sola línea.
# ------------------------------------
?>
```

3. Generar codi/imprimir per pantalla

- Podeu fer servir `echo` o `print`
- `echo` és més versàtil ja que ens permet posar variables dins la cadena

```php
echo "<h1>Podem mostrar HTML fent un echo</h1>";
echo "un".$variable ." dos";
echo "$nom, com estàs? 'uep'";
echo "un", "dos o tres", "quatre";
```

Imprimir alguna cosa és molt comú en PHP i par això hi ha una sintaxis ràpida de fer-ho:

```php
// sintaxi compacta
<?= "Hello World" ?>
```

4. Variables

- Les variables ens permeten guardar informació i fer-ne ús quan la necessitem.
- Les variables tenen tipus (sencer, booleà, cadena, float, array)
- les variables són sensibles a majúscules i minúscules. No es permeten espais per anomenar variables
- convé posar sempre noms significatius
- millor en minúscules, ha de començar amb lletra o subratjat (_) seguit de més lletres, números o caràcters (_)

```php
$nom = "Això és una variable";
$num = 29.1;
```

4.1. Variables globals

Són aquelles que ens dona ja el PHP com a predefinides

- `$_GET` : conté les dades que venen com a paràmetres de la url o a un form enviat per GET
- `$_POST` : conté les dades enviades per un form per POST

4.2. Mirar si una variable existeix

Algunes vegades ens instaressa saber si una variable té valor o si està definida.

- `empty($variable)` torna vertader si la variable està definida però no té cap valor, per exemple la cadena buida o una matriu sense elements.
- `is_null($variable)` > torna vertader si el contingut de la variable és el valor null.
- `isset($variable)` ?> torna vertader si la variable ha estat definida.

5. Operacions

5.1. Aritmètiques

```
+ Addition
- Subtraction
* Multiplication
/ Division
% Modulo
** Exponentiation
```

5.2. Números

```php
// Suma/increment ràpid
$value = 10
$value++ // 11
// or
$value += 1 // 11

// Resta ràpida d'un número
$value = 10
$value-- // 9
// or
$value -= 1 // 9

// mirar si és un número
echo is_numeric('59.99'); # true

// Arrodonir
echo round(0.80);  //  1

// Arrodonir a un nombre determinat de decimals
echo round(1.49356, 2));  // retorna 1.49

// Obtenir un nombre aleatori entre dos nombres
echo(rand(10, 100)); # 89
```

5.3. Cadenes

```php
// String can use single quote
$name = 'Mike'
// or double quote
$name = "Mike"

// Double quote string can escape characters \n = new line  \t = tab  \\ = backslash
echo "Hello Mike\nHello David";

// Double quote string can do interpolation
echo "Hello $name";

// string concat
echo 'Hello ' . $name;

// string length
echo strlen($name);

// Remove space(s) before and after
echo trim($text)

// Convert to lowercase / uppercase
echo strtolower($email);
echo strtoupper($name);

// Converts the first character to uppercase
echo ucfirst($name);  // 'Mike'

// Replace text a by text b in $text
echo str_replace('a', 'b', $text);

// String Contains (PHP 8)
echo str_contains($name, 'ke')  # true

// Find numeric position of first occurrence
$pos = strpos($name, 'k'); # 2

// Returns portion of string (offset / length)
echo substr($name, 0, $pos); # Mi
```

6. Depurant codi

Per depurar hi ha dues funcions que ens permeten obtenir informaició sobre una variable:

- `print_r($variable)`
- `var_dump($variable)`

Si voleu mostrar tots els errors posau al principi de la plana

```php
<?php
ini_set('display_errors', 1);
error_reporting(E_ALL);
?>
```

I no oblideu llevar-ho en producció!!!!

7. Condicions

- `>` : major que
- `<` : menor que
- `==` : igual
- `===` : estrictament igual
- `!=` : no igual

i les combinacions de `>=` , `<=`

7.1. Operadors lògics

- or (||)
- and (&&)
- or exclusiu (xor)
- not (!)

7.2. Sintaxis

```php
// simple: si passa això
if (condicio) {
    // codi a executar quan la condició és vertadera
}

// si pasa això i si no ...
if (condicio) {
    // codi a executar quan la condició és vertadera
} else {
    //  codi a executar si tota la resta no es compleix
}

// si passa això, o això altre, o si no ...
if (condicio) {
    //  codi a executar quan la condició és vertadera
} elseif (condicio2) {
    //  codi a executar quan la condicio2 és vertadera
} else {
    // codi a executar si tota la resta no es compleix
}

?>
```

Exemples:

```php
if ($condition == 10) {
    echo 'condition 10'
} elseif  ($condition == 5) {
    echo 'condition 5'
} else {
    echo 'all other conditions'
}

// And condition = &&
if ($condition === 10 && $condition2 === 5) {
    echo '10 and 5'
}

// Or condition = ||
if ($condition === 10 || $condition2 === 5) {
    echo '10 or 5'
}
```

8. Arrays

Els arrays es declaren amb la paraula clau `array` i poden ser indexats o associatius.

Ens permeten guardar informació estructurada i els seus elements poden ser també arrays.

8.1. Indexats

Els elements de l'array s'accedeixen per posició

```php
<?php
$seven = 7;
$arrayname = array( "this is an element", 5, $seven );

// o també
echo $arrayname[0];   //prints: this is an element
echo $arrayname[1];   //prints: 5
echo $arrayname[2];   //prints: 7
?>
```

8.2. Associatius

Els elements de l'array s'accedeixen per la clau de l'element,
també anomenats diccionaris.

```php
<?php
$first_array = array("key1" => "the first element", "key2" => "the second element");

$second_array = array(
    "key3" => "this is the first element of the second array",
    "key4" => "this is the second element of the second array",
);
echo $first_array['key1'];    //prints "the first element."
echo $second_array['key3'];   //prints "the first element of the second array"
echo $first_array['key2'];    //prints "the second element"
echo $second_array['key4'];   //prints "this is the second element of the second array"
```

8.2.1. Saber si un element està a l'array

```php
in_array ( mixed $needle , array $haystack  ) //torna vertader si $needle està a
                                              // l' array $haystack
```

Per exemple

```php
<?php
  $cotxes=array("Volvo"=>"XC90","BMW"=>"X5","Toyota"=>"Highlander");
  $marques = array_keys($cotxes);
  if (in_array('Volvo', $marques)) {
    echo "tenc un volvo";
  }
?>
```

8.2.2. Obtenir la longitud d'un array

`count` ens diu quants elements té un array, ja sigui de tipus associatiu o indexat.

```php
<?php
$comida = array('frutas' => array('naranja', 'plátano', 'manzana'),
                'verduras' => array('zanahoria', 'col', 'guisante'));

echo count($comida); // retorna 2
?>
```

9. Bucles

Els bucles ens permeten repetir una acció, ja sigui recorrer un array o fer alguna cosa mentre es compleixi una condició.

9.1. For Loop

```php
<?php
for (initialization; condition; increment)
{
   code to be executed;
}
?>
```

Exemple:

```php
<?php
for ($i=0; $i <= 10; $i++)
{
   echo "The number is ".$i."<br />";
}
?>
```

9.2. Do While

S'executa al manco un cop, comprovant la condició al final

```
do
{
   codi que volem executar
}
while (condition);
```

Exemple:

```
<?php
$i = 0;
do {
   echo "The number is ".$i."<br/>";
   $i++;
}
while ($i <= 10);
?>
```

9.3. While do

La condició s'avalua al principi, així que el codi es pot executar o no segons es complexi o no la condició.

```
$number = 1;
while ($number < 10) {
    echo 'value : ' . $number ;
    $number += 1;
}
```

Aquest codi mira si el valor de la variable `$number` és menor que deu, en en cas afirmatiu imprimeix el valor de la variable i després l'incrementa, tornant a comprovar la condició.

9.4. Foreach

El `foreach` és la manera més còmoda de recorre un array, ja sigui indexat o associatiu ja que no requereix tractar amb comptadors o

9.4.1. Arrays indexats

El `foreach` reocrre l'array i va posant cada element dins la variable

```
foreach (array as value)
{
   code to be executed;
}
```

Exemple:

```
<?php
$email = array('john.smith@example.com', 'alex@example.com');
foreach ($email as $value) {
   echo "Processing ".$value."<br />";
}
?>
```

En el nostre cas tenim una llista de emails dins l'array `$email` el `foreach` agafa element a element i ho posa dins la variable `$value` . Així la primera vegada tendrà el primer correu, la segona el segon, etc.

9.4.2. Arrays associatius

El `foreach` ens permet recorrer l'array agafant tant la clau com el valor, que es posaran a cada iteració dins les variables que facem servir.

```
foreach (array as key => value)
{
   code to be executed;
}
```

Exemple:

```
<?php
$person = array('name' => 'Andrew', 'age' => 21, 'address' => '77, Lincoln st.');
foreach ($person as $key => $value) {
   echo $key." is ".$value."<br />";
}
?>
```

9.5. Switch

El `switch` ens permet executar codi en funció del valor de la variable que facem servir.

```php
// Compare same variable with multiple values
switch ($color) {
    case 'red':
        echo 'The color is red';
         break;
    case 'yellow':
        echo 'The color is yellow';
        break;
    case 'blue':
        echo 'The color is blue';
        break;
    default:
        echo 'The color is unknown';
}
```

9.5.1. Exemples

```php
//for loop
for ($i = 0; $i < 20; $i++) {
    echo "i value = " . i;
}

//while loop
$number = 1;
while ($number < 10) {
    echo 'value : ' . $number ;
    $number += 1;
}

//do while
$number = 1;
do {
    echo 'value : ' . $number ;
    $number += 1;
} while ($number < 10);

// foreach with break / continue exemple
$values = ['one', 'two', 'three'];
foreach ($values as $value) {
    if ($value === 'two') {
        break; // exit loop
    } elseif ($value === 'three') {
        continue; // next loop iteration
    }
}
```

9.6. Testimonis, comptadors i acumuladors

Testimonis, comptadors i acumuladors són patrons de programació que ens trobam sovint en el dia a dia.

9.6.1. Testimonis

Quan utilitzam blues sovint ens trobam amb la necessitat de sortir del bucle (`break`) o continuar amb la següent iteració (`continue`) en funció d'una condició que s'ha de comprovar cada vegada per decidir el que s'ha de fer. La variable o condició que es fa servir se'n diu testimoni.

Suposem que tiram un dau 30 vegades, volem saber si ha sortit un cinc en alguna d'aquestes tirades. Per això necessitam un bucle que es vagi repetint les vegades que volem (30) i un testimoni que ens digui si ha sorti o no el cinc.

Inicialment podem dir que no ha sortit cap cinc, ja que de fet encara no hem començat.

```
<?php
print "<p>Inici</p>\n";
$hayCinco = false;
for ($i = 0; $i < 30; $i++) {
    $dado = rand(1, 6);
    print "<p>Tirada de dado: $dado</p>\n";
    if ($dado == 5) {
        $hayCinco = true;
    }
}
if ($hayCinco) {
    print "<p>Ha salido al menos un 5.</p>\n";
} else {
    print "<p>No ha salido ningún 5.</p>\n";
}
print "<p>Final</p>\n";
?>
```

En aquest exemple `$hayCinco` es la variable que conté el valor que ens indica si ha sortit el cinc dins del bucle.

Fixe'm-nos que si no fos per aquesta variable no podríem saber si ha sortit o no. La variable fa de testimoni.

Hi pot haver tants testimonis com necessitem.

9.6.2. Comptadors

Un comptador és una variable (o més d'una) que ens serveix per anar acumulant ocurrències d'un fet.

En el nostre exemple, suposem que volem saber les vegades que ha sortit el nombre cinc.

Definim una variable `$cuenta` que inicialitzam a zero (valor inicial o de partida), cada vegada que surt un cinc incrementam el contador, és a dir, sumam un. Es pot llegir com "els que hi havia i un més".

```
<?php
print "<p>Comienzo</p>\n";
$cuenta = 0;
for ($i = 0; $i < 30; $i++) {
    $dado = rand(1, 6);
    print "<p>Tirada de dado: $dado</p>\n";
    if ($dado == 5) {
        $cuenta++;
    }
}
print "<p>Han salido $cuenta cinco(s).</p>\n";
print "<p>Final</p>\n";
?>
```

9.6.3. Acumuladors

Un acumulador és semplant a un comptador, però ara el que ens insteressa no és comptar coses, sinó anar afegit quantitats.

Suposem que volem sumar totes les tirades. Per això podem crear una variable `$total` que inicialitzarem a zero (encara no hem tirat) i on anirem afegint el valor de cada tirada.

```
<?php
print "<p>Comienzo</p>\n";
$total = 0;
for ($i = 0; $i < 3; $i++) {
    $dado = rand(1, 6);
    print "<p>Tirada de dado: $dado</p>\n";
    $total = $total + $dado;
}
print "<p>Ha obtenido $total puntos.</p>\n";
print "<p>Final</p>\n";
?>
```

Podem tenir tans acumuladors com necessitem. Sovint es fan servir fins i tot arrays per anar gestionant les sumes o també pels comptadors.

10. Formularis

Una aplicació que no pot agafar dades de l'exterior o de l'usuari ens limita molt. El que volem sovint és poder dir a l'aplicació que faci alguna cosa a partir d'informació que li passarem nosaltres. Aquesta informació passa del navegador a la nostra aplicació web normalment mitjançant un formulari.

Els formularis ens permeten obitenir dades mitjançant el navegador i enviar-los al servidor, bé mitjançant parametres de la url o bé dins el cos de la petició.
Ens serveixen per cercar, editar, modificar, crear o borrar dades de la nostra aplicació.

L'HTML ens defineix tant la sintaxi del formulari i la petició com els controls que podem trobar, en PHP farem el tractament dels valors i de la informació.

Els formularis més bàsics tenen la següent estructura:

```
<form action="url_on_enviar_les_dades" method="metode_d'enviament">
    // aquí posam els diferents controls d'entrada (inputs)
    Nom*: <input type="text" name="nom" value="<?php echo $nom; ?>" required><br>

    // i posam el botó d'enviar
     <input type="submit" value="Saludar">
</form>
```

10.1. Enviament de la informació

Els formularis bàsicament poden enviar les peticions al servidor fent serivr dos mètodes: `GET` i `POST`.
És important saber quan fer-ne servir un o altre i les seves implicacions.

10.1.1. GET

La informació s'envia de forma visible i codificada dins la URL en el que es coneix com un `Query String` amb un format de clau valor separat per `&`, per exemple:

```
http://www.aulabalear.org?alumne=toni&edat=10
```

Aquesta *url* pasa dos paràmetres per *get*: alumne i edat. El primer, alume té el valor `toni` i l'edat té el valor 10.
Hem de tenir en compte que podem passar paràmetres sense valor

```
http://www.aulabalear.org?alumne=toni&edat=10&religio=
```

En aquest cas podríem suposar que la persona no ha contestat o no ha establer el valor de la variable `religio`

10.1.2. Usos i limitacions

- Màxim 2000 caràcters
- La informació és visible
- No es poden enviar dades binàries
- El seu us és ideal per peticions que no modifiquen dades

10.1.3. Exemple en PHP

```
<!DOCTYPE html>
<html>
    <!-- capçalera-->

    <head>
        <meta charset="UTF-8">
        <title>Ejemplo get</title>
    </head>
    <!-- cos-->

    <body>

        <?php
    // Agafam les variables del get del navegador
    // normalment posarem com a nom de la variable el paràmetre que
    // ve del get, però ho podeu canviar.
    // En el codi agafam els valors amb `$GET`, si hi són els utilitzam
    // i si no posam un valor per defecte per tenir sempre la variable
    // inicialitzada, és a dir, amb un valor inicial, i que PHP no es queixi
    // de que la variable no està definida.


     $nom = isset($_GET["nom"]) ? $_GET["nom"] : "";
     $llinatge = isset($_GET["llinatge"]) ? $_GET["llinatge"] : "";
     $edat = isset($_GET["edat"]) ? $_GET["edat"] : 18;
     $cars = isset($_GET["cars"]) ? $_GET["cars"] : "";


     // així si no s'ens passa l'edat el valor que agafam per defecte
     // és 18.

    // podem definir una variable que ens indiqui si tenim o no totes les
    // dades que necessitam, comprovant els continguts de les variables.
    // nosaltres ho fem amb $tengo_datos

    if (($nom !="" ) and ($llinatge != "")) {
        $tengo_datos = true;
     } else {
        $tengo_datos = false;
     }


    // definim el formulari. En el nostre cas enviam la informacio
    // a la mateixa plana, que hem anomenat formget.php
    // fixau-vos com establir el valor dels inputs per a que agafin
    // el valor de les variables que hem definit un poc més adalt.

    ?>

        <form action="formget.php" method="get">
```

```
            Nom*: <input type="text" name="nom"
                value="<?php echo $nom; ?>" required><br>
            Llinatge*: <input type="text" name="llinatge" required
                value="<?php echo $llinatge; ?>"><br>
            Edat*: <input type="number" name="edat" required
                value="<?php echo $edat; ?>"><br>
            <label for="cars">Choose a car:</label>
            <select id="cars" name="cars">
                <option value="volvo">Volvo</option>
                <option value="saab">Saab</option>
                <option value="fiat">Fiat</option>
                <option value="audi">Audi</option>
            </select>

            <input type="submit" value="Enviar">
        </form>

        <hr>

        <?php
    // podem mirar si ens han posat nom o dades en general, si és així
    // imprimim la salutació, en cas contrari demanam que es faci.

    if($tengo_datos==true) {

        echo "Hola $nom $llinatge <br>";
        echo "Tens $edat anys <br>";
        echo "I t'agrada $cars <br>";
          } else {
            echo "Introdueix nom i llinatges<br>";

          }
     ?>
        <hr>
    </body>

</html>
```

10.2. POST

La informació es codifica i s'envia en el cos *body* de la petició del HTTP Request i no és visible a la url.
Una de les coses més importants que hem de recordar és el de fer una redirecció una vegada hem obtingut les dades, ja que en cas contrari si referescam la plana es tornarà a enviar tota la ifnormaició una altra vegada.

És el patró **redirect after post** que ve a dir que després d'un post sempre hem de fer un redirect. El redirect pot nar a la mateixa plana o anar a una plana distinta. Sovint això darrer sol ser el més senzill.

Ho veurem quan tractem les bases de dades.

10.2.1. Usos i limitacions

- No está limitat en volum a enviar
- La informació no és visible. Junt amb https fa que la comunicació sigui segura.
- Es pot enviar text normal i binaris
- És ideal per peticions que modifiquen les dades.

10.2.2. Exemple en PHP

```
<html>
<body>
<form action="." method="post">
    Nom: <input type="text" name="nom"><br>
    Email: <input type="text" name="email"><br>
    <input type="submit" value="Enviar">
</form>
Hola <?php isset($_POST["nom"]) ? print $_POST["nom"] : ""; ?><br>
Email: <?php isset($_POST["email"]) ? print $_POST["email"] : ""; ?>
</body>
</html>
```

10.2.3. Enviant arxius

Quan volem enviar arxius li hem de dir al formlari, per indicar que volem que les dades es xapin en múltiples parts: una per a cada arxiu i una per les dades del body del formulari.

Exemple

```
<!DOCTYPE html>
<html>
    <!-- capçalera-->

    <head>
        <meta charset="UTF-8">
        <title>Ejemplo POST</title>
    </head>
    <!-- cos-->

    <body>

        <?php
    // Agafam les variables del get del navegador
    // normalment posarem com a nom de la variable el paràmetre que
    // ve del get, però ho podeu canviar.
    // En el codi agafam els valors amb `$POST`, si hi són els utilitzam
    // i si no posam un valor per defecte per tenir sempre la variable
    // inicialitzada, és a dir, amb un valor inicial, i que PHP no es queixi
    // de que la variable no està definida.


     $nom = isset($_POST["nom"]) ? $_POST["nom"] : "";
     $llinatge = isset($_POST["llinatge"]) ? $_POST["llinatge"] : "";
     $edat = isset($_POST["edat"]) ? $_POST["edat"] : 18;
     $cars = isset($_POST["cars"]) ? $_POST["cars"] : "";


     // així si no s'ens passa l'edat el valor que agafam per defecte
     // és 18.

    // podem definir una variable que ens indiqui si tenim o no totes les
    // dades que necessitam, comprovant els continguts de les variables.
    // nosaltres ho fem amb $tengo_datos

    if (($nom !="" ) and ($llinatge != "")) {
        $tengo_datos = true;
     } else {
        $tengo_datos = false;
     }


    // definim el formulari. En el nostre cas enviam la informacio
    // a la mateixa plana, que hem anomenat formget.php
    // fixau-vos com establir el valor dels inputs per a que agafin
    // el valor de les variables que hem definit un poc més adalt.

    ?>

        <form action="formpost.php" method="post">
```

```
            Nom*: <input type="text" name="nom"
                value="<?php echo $nom; ?>" required><br>
            Llinatge*: <input type="text" name="llinatge" required
                value="<?php echo $llinatge; ?>"><br>
            Edat*: <input type="number" name="edat" required
                value="<?php echo $edat; ?>"><br>
            <label for="cars">Tria un cotxe:</label>
            <select id="cars" name="cars">
                <option value="volvo">Volvo</option>
                <option value="saab">Saab</option>
                <option value="fiat">Fiat</option>
                <option value="audi">Audi</option>
            </select>

            <input type="submit" value="Enviar">
        </form>

        <hr>

        <?php
    // podem mirar si ens han posat nom o dades en general, si és així
    // imprimim la salutació, en cas contrari demanam que es faci.

    if($tengo_datos==true) {

        echo "Hola $nom $llinatge <br>";
        echo "Tens $edat anys <br>";
        echo "I t'agrada $cars <br>";
          } else {
            echo "Introdueix nom i llinatges<br>";

          }
     ?>
        <hr>
    </body>

</html>
```

Com podem veure, al codi hi ha poques diferències respecte al `GET`, però si ens fixam com la url no canvia podem veure millor les diferències entre `GET` i `POST`.

10.2.4. Testejant si és un GET o un POST

Algunes vegades ens interessarà saber si el mètode amb el que estam arribant a la plana és un `GET` o és un `POST` o un altre tipus, i fer una cosa o altra. Això ho podem fer accedint a l'array global `$_SERVER` amb la clau `REQUEST_METHOD`.

```php
if($_SERVER["REQUEST_METHOD"] == "POST"){
  // obtenim la info
}
```

11. Elements dels formularis

Els elements més comuns d'un formulari ès l' `input` que agafa diversos tipus segons el que volguem obtenir.
En HTML5 a més ens permet fer autocompletat, validacions i mostrar diversos controls segons la dada a obtenir.

important Recordau que per a que per a poder recollir la informació tal com estam fent aquí, els elements han de tenir nom, és a dir, a l'atribut `name` li hem de posar un valor. Si no és així no podrem recollir la informació, ni per `get` ni per `post` .

Aquí veurem alguns dels elements més comuns i com s'utilitzen, a les referències us poso on trobar més informació:

12. Com fer les proves

Per no repetir tant codi, el que farem serà agafar un codi php de base i canviar sols les seccions que toqui per provar el nostre element d'entrada de formulari. Per a facilitar les coses i que es vegi millor ho farem amb un `GET`

Tractarem:

- Com agafar la informació i establir un valor per defecte
- Com posar la informació dins l'element
- Com imprimir la informació

El codi base que farem servir és:

```
<!DOCTYPE html>
<html>
    <!-- capçalera-->

    <head>
        <meta charset="UTF-8">
        <title>Ejemplo get</title>
    </head>
    <!-- cos-->

    <body>

        <?php
    // Agafam les variables del get del navegador

    /// ----------------------------------------


    // comprovarem si tenim dades o no ----------

    $tengo_datos = false; // això ho canviarem

    // ----------------------------------------

    ?>

        <form action="formget.php" method="get">
            <!-- aqui posarem el que volem provar -->




            <!- ------------------------------------>

                <input type="submit" value="Enviar">
        </form>

        <hr>

        <?php
    // podem mirar si ens han posat nom o dades en general, si és així
    // imprimim la salutació, en cas contrari demanam que es faci.

    if($tengo_datos==true) {

        // imprimim els valors

      } else {

        echo "Introdueix les dades<br>";
```

```
            }
        ?>
            <hr>
    </body>

</html>
```

12.1. input

És un dels elements més utilitzat i també un dels més complexos per la quantiat d'opcions que podem fer servir.

12.1.1. tipus text

Ens permet introduïr un text simple, sense salts de línea. Els atributs més importants són el `value` per indicar el valor per defecte, `placeholder` per indica run text d'exemple i `required` per indicar si el valor és obligatori o no.

```
    <label for="fname">Nom:</label><br>
    <input type="text" id="fname" name="fname">
```

El `label` ens permet posar un texte lligat al camp `input` el `for` ha de tenir el mateix valor que l'atribut `id` del camp.

Un dels atributs força emprats en el texte és el `maxlength` per indicar la longitud màxima del texte que hi podem posar.

```
<input id="texte" type="text" name="texte" maxlength="10" required
               value="<?php echo $texte?>">
```

Exemple

```php
<!DOCTYPE html>
<html>
    <!-- capçalera-->

    <head>
        <meta charset="UTF-8">
        <title>Ejemplo get</title>
    </head>
    <!-- cos-->

    <body>

        <?php
    // Agafam les variables del get del navegador
    $texte = isset($_GET["texte"]) ? $_GET["texte"] : "";
    /// ----------------------------------------
    // comprovarem si tenim dades o no ----------
    if ($texte==""){
        $tengo_datos = false; // això ho canviarem
    } else {
        $tengo_datos = true;
    }
    ?>

        <form action="input.php" method="get">
            <label for="texte">Texte*</label>
            <input id="texte" type="text" name="texte"
                placeholder="escriu un texte" required
                value="<?php echo $texte?>">

            <input type="submit" value="Enviar">

        </form>
        <hr>

        <?php
    if($tengo_datos==true) {
        echo "Has escrit $texte";
          } else {
            echo "Introdueix les dades<br>";
          }
     ?>
        <hr>
    </body>

</html>
```

Adaptau el codi per fer servir els tipus: `number` , `date` , `month` , `email` , `datetime` , `color` , `password` , `tel` , `time` , `url` , `week` .

12.1.2. radio

Los radio buttons son controles que nos permiten seleccionar solamente una de las opciones de entre las que presentamos. Su `input type` es `radio` y en el formulario pasan el nombre `name` y el valor `value` del radio button seleccionado. Por tanto cada radio button debe tener todos el mismo nombre `name`.

Para seleccionar uno de los radio buttons incluiremos el atributo `checked`. Para poder establecer el valor de checked a patir de los parámetros del formulario, debemos comparar el valor que tenemos con el valor que espera cada radio button.

```
<?php echo ($bebida=="refresco"? "checked": ""); ?>
```

que debemos incluir como código en nuestro radio button al igual que hacíamos en el caso anterior con el texto.

```
<!DOCTYPE html>
<html>
    <!-- capçalera-->

    <head>
        <meta charset="UTF-8">
        <title>Ejemplo de radio button</title>
    </head>
    <!-- cos-->

    <body>

        <?php
    // Agafam les variables del get del navegador
    $bebida = isset($_GET["bebida"]) ? $_GET["bebida"] : "";
    /// ----------------------------------------
    // comprovarem si tenim dades o no ----------
    if ($bebida==""){
        $tengo_datos = false; // això ho canviarem
    } else {
        $tengo_datos = true;
    }
    ?>

        <form action="radio.php" method="get">
            <p>Seleccione qué va a beber: </p>
            <p>
                <label for="refresco">Refresco</label>
                <input type="radio" name="bebida" value="refresco" id="refresco"
                    <?php echo ($bebida=="refresco"? "checked": ""); ?>>

            </p>
            <p>
                <label for="cocacola">Agua</label>
                <input type="radio" name="bebida" value="agua" id="agua"
                    <?php echo ($bebida=="agua"? "checked": ""); ?>>
            </p>
            <p>
                <label for="vino">Vino</label>
                <input type="radio" name="bebida" value="vino" id="vino"
                    <?php echo ($bebida=="vino"? "checked": ""); ?>>
            </p>
            <p> <label for="gaseosa">Gaseosa</label>
                <input type="radio" name="bebida" value="gaseosa" id="gaseosa"
                    <?php echo ($bebida=="gaseosa"? "checked": ""); ?>>
            </p>
            <p><input type="submit" value="Enviar"></p>


        </form>
```

```
        <hr>

        <?php
    if($tengo_datos==true) {
        echo "Has elegido <strong>$bebida</strong>";
          } else {
            echo "Selecciona tu bebida<br>";
          }
     ?>
        <hr>
    </body>

</html>
```

12.1.3. check

Un check button ens permet marcar més d'una opció. Realment funcionen com a elements independents, amb el seu propi nom y valor, amb l'atribut `checked` per indicar si l'element ha estat triat o no.

Hem de tenir en compte que els elements que no triem no passaran com a paràmetres del formulari, per tant som els responsables de saber quins elements hi ha i què implica que passi el seu valor o no.

```
<!DOCTYPE html>
<html>
    <!-- capçalera-->

    <head>
        <meta charset="UTF-8">
        <title>Ejemplo de radio button</title>
    </head>
    <!-- cos-->

    <body>

        <?php
    // Agafam les variables del get del navegador
    $refresco = isset($_GET["refresco"]) ? $_GET["refresco"] : "";
    $agua = isset($_GET["agua"]) ? $_GET["agua"] : "";
    $vino = isset($_GET["vino"]) ? $_GET["vino"] : "";
    $gaseosa = isset($_GET["gaseosa"]) ? $_GET["gaseosa"] : "";
    /// ----------------------------------------
    // comprovarem si tenim dades o no ----------
    if (($refresco=="") and ($agua=="") and ($vino=="") and ($gaseosa=="")){
        $tengo_datos = false; // això ho canviarem
    } else {
        $tengo_datos = true;
    }
    ?>

        <form action="check.php" method="get">
            <p>Seleccione qué va a beber: </p>
            <p>
                <label for="refresco">Refresco</label>
                <input type="checkbox" name="refresco" value="refresco" id="refresco"
                    <?php echo ($refresco=="refresco"? "checked": ""); ?>>

            </p>
            <p>
                <label for="agua">Agua</label>
                <input type="checkbox" name="agua" value="agua" id="agua"
                    <?php echo ($agua=="agua"? "checked": ""); ?>>
            </p>
            <p>
                <label for="vino">Vino</label>
                <input type="checkbox" name="vino" value="vino" id="vino"
                    <?php echo ($vino=="vino"? "checked": ""); ?>>
            </p>
            <p> <label for="gaseosa">Gaseosa</label>
                <input type="checkbox" name="gaseosa" value="gaseosa" id="gaseosa"
                    <?php echo ($gaseosa=="gaseosa"? "checked": ""); ?>>
            </p>
            <p><input type="submit" value="Enviar"></p>
```

```
        </form>
        <hr>

        <?php
    if($tengo_datos==true) {
        echo "Has elegido: <br>";
        echo "<ul>";
        if ($refresco=="refresco") {
            echo "<li>$refresco</li>";
        }
        if ($agua=="agua") {
            echo "<li>$agua</li>";
        }
        if ($vino=="vino") {
            echo "<li>$vino</li>";
        }
        if ($gaseosa=="gaseosa") {
            echo "<li>$gaseosa</li>";
        }
        echo "</ul>";
          } else {
            echo "Selecciona tus bebidas<br>";
          }
     ?>
        <hr>
    </body>

</html>
```

Fixau-vos com estam agafant els valors i com els estam pintant, un per un. Els donam sentit a l'hora de pintar-los però són independents.
Observau també la quatitat de codi que estam posant i repetint, codi que és pràcticament sembre el mateix.

Podem fer que ens quedi menys codi fent servir els arrays i fent servir els bucles per a que sigui el propi PHP qui escrigui el codi HTML que necessitam.

12.1.4. select

El `select` ens permet mostrar una llista desplegable i que l'usuari trii una de les opcions o més d'una si posam l'atribut `multiple`. Podem marcar l'opció seleccionada amb l'atribut `selected` però per marcar-ho, tal com feiem en el cas del radio button, haurem de comparar el que tenim amb el valor de cada opció del `select`.

```php
<!DOCTYPE html>
<html>
    <!-- capçalera-->

    <head>
        <meta charset="UTF-8">
        <title>Ejemplo de radio button</title>
    </head>
    <!-- cos-->

    <body>

        <?php
    // Agafam les variables del get del navegador
    $bebida = isset($_GET["bebida"]) ? $_GET["bebida"] : "";
    /// ----------------------------------------
    // comprovarem si tenim dades o no ----------
    if ($bebida==""){
        $tengo_datos = false;
    } else {
        $tengo_datos = true;
    }
    ?>

        <form action="select.php" method="get">
            <label for="bebida">¿Qué vas a beber?:</label>

            <select name="bebida" id="bebida">
                <option value="">--selecciona tu bebida--</option>
                <option value="vino"
                    <?php echo ($bebida=="vino"? "selected": ""); ?>>
                    Vino</option>
                <option value="refresco"
                    <?php echo ($bebida=="refresco"? "selected": ""); ?>>
                    Refresco</option>
                <option value="gaseosa"
                    <?php echo ($bebida=="gaseosa"? "selected": ""); ?>>
                    Gaseosa</option>
                <option value="agua"
                    <?php echo ($bebida=="agua"? "selected": ""); ?>>
                    Agua</option>
            </select>

            <p><input type="submit" value="Enviar"></p>


        </form>
        <hr>

        <?php
```

```
    if($tengo_datos==true) {
        echo "Has elegido: $bebida<br>";

          } else {
            echo "Selecciona tus bebidas<br>";
          }
     ?>
        <hr>
    </body>

</html>
```

12.1.5. textarea

Un textarea ens permete demanar informació textual que conté més d'una línea. En passar per GET hem de tenir en compte que tot va a la URL, per la qual cosa el límit que podem passar és menor que si ho fem per POST. Normalment passarem sempre aquests tipus de components per POST.

Un textarea agafa com a atributs propis l'amplada en columnes `cols` i l'alçada en línees `rows` .

A diferència d'altres controls, els valor no es troba dins el value, sino entre els tags que defineixen el component.

Podem establir el nombre màxim i mínim de caràcters amb `minlength` i `maxlength` i com altres components indicar si el contingut és obligatori `required` , si hi ha un `placeholder` o bé si està deshabilitat `disabled` o bé `readonly` .

12.1.6. Referències

- https://developer.mozilla.org/es/docs/Web/HTML/Elemento
- https://www.tutorialrepublic.com/html-tutorial/html-forms.php
- https://www.w3schools.com/html/html_forms.asp

És important validar les dades i verificar que el format és correcte! i mai us heu de fiar de les dades que venen de l'usuari.

13. Preguntes freqüents

13.1. No surt per pantalla?

- Hem agafat els valors i els hem posat a una variable?

- Recordem que no surtirà res si no fem echo o print
- Important! Si feis servir un nom de paràmetre que no existeix *petarà* hem de comprovar que el nostre `php.ini` té activada la opció de depuració per a poder veure els errors.