

DAW  
Desarrollo de Aplicaciones Web  
2º Curso

DWES  
Desarrollo Web Entorno Servidor

UD 2. Desarrollo de Aplicaciones Web con PHP

2.2 PHP avanzado: includes, extensiones,  
formularios y debugger

IES BALMIS  
Dpto Informática  
Curso 2025-2026  
Versión 3 (10/2025)

## UD2 – Programación Web Dinámica en Servidor

### ÍNDICE

- 1. Lenguajes de script para servidores web**
- 2. Lenguaje PHP**
- 3. Herramientas gráficas para escribir código PHP**
- 4. Integración con los lenguajes de marcas**
- 5. Funciones integradas o predefinidas**
- 6. Arrays (Vectores – Matrices – Colecciones)**
7. Includes
8. Extensiones y código open source
9. Funciones definidas por el programador/usuario
10. Formularios HTML
11. Formularios PHP
12. Debugger

## 7. Includes

### 7.1 Agrupar código y reutilizarlo

Cuando vamos desarrollando funciones, es normal crearnos librerías o plantillas que contengan nuestro código. Para ello disponemos de dos instrucciones en PHP:

- **include**: que permite incluir el contenido de un archivo
- **require\_once**: que también permite incluir el contenido de un archivo, pero sólo lo incluye una vez aunque se llame varias veces.

```
<?php
    require_once "milibreria.php";

    include "include_cabecera.php";
    include "include_main.php";
    include "include_piedepagina.php";
?>
```

Las instrucciones **include** y **require\_once** funcionan con cualquier extensión, por lo que también se puede usar con archivos html.

#### Práctica p01 (p01inc.php)

Partiendo de una página "\_p01inc-original.html" con la estructura de css agrupado y con <header>, <main>, <section>, <footer>, crear un archivo en PHP para cada página que cargue la estructura de estilos, cabecera, cuerpo y pie de forma independiente.

La estructura inicial es:

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Página Web Sencilla</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            line-height: 1.6;
            color: #333;
            background-color: #f4f4f4;
        }
        .container {
            width: 80%;
            margin: 0 auto;
            overflow: hidden;
        }
        header {
            background: #333;
            color: #fff;
            padding: 20px 0;
            text-align: center;
        }
    </style>
</head>
<body>
    <div class="container">
        <div class="header">
            <h1>Página Web Sencilla</h1>
        </div>
        <div class="main">
            <h2>Contenido Principal</h2>
        </div>
        <div class="footer">
            <p>Página Web Sencilla</p>
        </div>
    </div>
</body>
</html>
```

```
        section {
            padding: 20px;
            background: #fff;
            margin: 20px 0;
            border-radius: 5px;
        }

        footer {
            background: #333;
            color: #fff;
            text-align: center;
            padding: 10px 0;
            margin-top: 20px;
        }

        h1, h2 {
            margin-top: 0;
        }

        p {
            margin-bottom: 15px;
        }
    </style>
</head>
<body>

    <header>
        <div class="container">
            <h1>Mi Página Web Sencilla</h1>
        </div>
    </header>

    <main>
        <section class="container">
            <h2>Sección 1</h2>
            <p>Este es el contenido de la primera sección de la página web.</p>
            <p>En una implementación estructurada, este contenido vendría del archivo
                "include_seccion1.php".</p>
        </section>

        <section class="container">
            <h2>Sección 2</h2>
            <p>Este es el contenido de la segunda sección de la página web.</p>
            <p>En una implementación estructurada, este contenido vendría del archivo
                "include_seccion2.php".</p>
        </section>
    </main>

    <footer>
        <div class="container">
            <p>&copy; 2025-2030 Mi Página Web Sencilla</p>
        </div>
    </footer>
</body>
</html>
```

No hay que confundir estas instrucciones con la **redirección de páginas**. Cuando redireccionamos, perdemos nuestra página y se carga la nueva.

Si deseamos redirigir a otra página, usaremos javascript en el cliente:

```
...  
<script>  
    location.href = "https://www.google.es";  
</script>  
...
```

o PHP en el servidor:

```
...  
<?php  
    header("Location: https://www.google.es");  
?>  
...
```

Otra opción de la redirección de páginas es utilizar gestionar el evento click para redirigir a otra página. En este caso, lo recomendable es usar Javascript/JQuery en el cliente.

#### **Práctica p02 (p02click.html)**

Crea una página en HTML con 4 botones que redirijan a diferentes páginas web utilizando funciones de javascript.

Recuerda aunque se puede usar el atributo **onclick** de javascript (nosotros usaremos el método **addEventListener** de Javascript) o el método **click** de JQuery.

## **7.2 Agrupar código y reutilizarlo**

Partiendo del diseño estático de una web, y aprovechando la programación en servidor con PHP junto con los `include/require_once`, podemos seguir una serie de pasos que nos ayuden.

#### **Práctica p03 (p03includes) (MUY IMPORTANTE)**

En este ejemplo se propone un proceso que nos permite:

- no tener código repetido o redundante
- aprovechar la parametrización para simplificar el código
- facilitar el mantenimiento del sitio web
- dar facilidad para escalar, es decir, aumentar opciones de menú con nuevo contenido

## 8. Extensiones y código open source

### 8.1 Extensiones

PHP dispone de muchas extensiones que pueden activarse en el **php.ini**.

Para activar una extensión deberemos añadir una entrada en el archivo php.ini indicando:

**extension=nombre**

donde nombre será una librería situada en la carpeta indicada por la variable `extension_dir` que por defecto en xampp es:

**extension\_dir = "\xampp\php\ext"**

Si activamos por ejemplo la extensión **curl** dispondremos de instrucciones para obtener información de una web.

Activa y desactiva la extensión para comprobar que el siguiente código PHP funciona o no dependiendo de la activación de la extensión:

#### Práctica p04 (p04curl.php)

```
<?php
// Crear un manejador de cURL
$ch = curl_init('http://www.iesdoctorbalmis.com/');
// Para curl_exec devuelva el contenido sin mostrarlo directamente
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

// Ejecutar
$html = curl_exec($ch);
// Comprobar si ocurrió un error
if (!curl_errno($ch)) {
    $info = curl_getinfo($ch);

    echo "<pre>";
    print_r($info);
    echo "</pre>";

    echo 'Se tardó ', $info['total_time'], ' segundos ';
    echo 'en enviar una petición a '.$info['url'];
    echo "<br><br>";
}

// Cerrar el manejador
curl_close($ch);
?>

<iframe srcdoc="<?php echo htmlspecialchars($html); ?>"
style="width:700px;height:400px;"></iframe>
```

#### Práctica p04 (p04info.php)

Accede con curl a <http://ip-api.com/json/80.24.59.78> y muestra el resultado obtenido en formato JSON.

## 8.2 Código Open Source

PHP dispone de muchas librerías para ampliar su funcionalidad:

- -Podemos consultar en el manual
  - <https://www.php.net/manual/es/> (Manual completo)
  - <https://www.php.net/manual/es/funcref.php> (Extensiones de PHP)

En internet existe mucho código libre que está a disposición de los programadores y que nos permite disponer de nuevas funcionalidades. Este código puede extender las funcionalidades de **HTML+CSS**, de **Javascript+jQuery** o de **PHP**.

Algunos ejemplos son:

- Envío de correos electrónicos a través de SMTP (Simple Mail Transfer Protocol) con PHPMailer
  - <https://github.com/PHPMailer/PHPMailer>
- Galerías de fotos con animación como NIVOSLIDER
  - <https://www.jqueryscript.net/demo/nivo-slider/>
- Creación de archivos PDF mediante código HTML con TCPDF
  - <https://tcpdf.org/>
  - <https://tcpdf.org/examples/>
- Creación de hojas de cálculo con PHPSpreadsheet
  - <https://phpspreadsheet.readthedocs.io/en/latest/>
- Uso de fuentes True Type
  - <https://fonts.google.com/>
- Utilidades jQuery UI que ofrece una combinación de interacción, efectos, widgets, utilidades y temas diseñados para funcionar bien juntos o por su cuenta
  - <https://jqueryui.com/demos/>
- Captcha para formularios en PHP como Simple-PHP-Captcha
  - <https://labs.abeautifulsite.net/simple-php-captcha/>

Antes de programar cualquier funcionalidad es recomendable buscar en internet si ya está programada y así reutilizar el código existente.

### **Práctica p05 (p05pdf\_create)**

Descarga el código del complemento tcpdf y crea un archivo PHP que muestre el resultado en un PDF.

### **Práctica p05 (p05pdfvisor)**

Este ejemplo usa el plugin instalado en el navegador cliente para visualizar archivos PDF.

Cuando instalamos un visor de PDF como Adobe Reader, se instala una extensión por defecto en el navegador que permite visualizar este tipo de archivos con la etiqueta **<object>**

## 9. Funciones definidas por el programador/usuario

En muchos casos tendremos que realizar una serie de instrucciones de forma reiterada y la forma más común de agruparlas para su posterior uso en cualquier lenguaje de programación son las funciones.

La sintaxis de las funciones en PHP es la siguiente:

```
// Definir la función
function nombreFuncion($param1, $param2, ... ) {
    [instrucciones];

    return $valor;
}

// Llamar a la función y almacenar el valor de retorno
$resultado = nombreFuncion($param1, $param2, ... );
```

Como ejemplo realizaremos una función para sumar dos números que recibimos como parámetros

```
function suma($num1, $num2) {

    // Devolvemos valor nulo si no son números
    if ( (!is_numeric($num1)) || (!is_numeric($num2)) ) {
        $valor=null;
    } else {
        $valor = $num1 + $num2;
    }

    return $valor;
}

// Llamar a la función y almacenar el valor de retorno

$num1=15;
$num2=34.26;
$resultado = suma($num1, $num2);
echo " La suma de $num1 + $num2 es $resultado ";
```

Puedes probar el código en el editor online:

<http://sandbox.onlinephpfunctions.com/>



## 9.1 Ejercicios de funciones

### Ejercicio 01 (b5ej01.php)

Crear una función denominada **parrafoAzul(\$cadena)** que reciba una cadena y muestre un párrafo en HTML de color azul que contenga dicha cadena.

### Ejercicio 02 (b5ej02.php)

Mostrar una tabla de 2 por 8 que muestre las primeras 8 potencias del número 2:

2 <sup>1</sup>	2
2 <sup>2</sup>	4
2 <sup>3</sup>	8

...

Hacer una función denominada **potencias()** que las calcule llamando la función **pow**. En PHP las funciones hay que definir las antes de invocarlas.

### Ejercicio 03 (b5ej03.php)

Crear una función llamada **numVocales(\$cadena)** que reciba una cadena y muestre el número de vocales que tiene.

- **Nota:** Debes comprobar que al trabajar con UTF8 las letras no ocupan 1 byte, por lo que utilizaremos las funciones de tipo **mb\_\*** como **mb\_substr**, **mb\_substr\_count** o **mb\_strlen**.
- **Nota:** También hay funciones interesantes para el trabajo con arrays como: **in\_array** y **array\_count\_values**.
  - Puedes consultarlas en [https://www.w3schools.com/php/php\\_arrays\\_functions.asp](https://www.w3schools.com/php/php_arrays_functions.asp)

### Ejercicio 04 (b5ej04.php)

Crear una función llamada **checkMatricula(\$matricula)** que reciba una cadena y devuelva true o false si cumple con el formato de longitud 7 o 8 siguiente:

9999-XXX o 9999-XX

donde X son letras mayúsculas y 9 son números.

**Nota:** consulta funciones de **ctype\_\*** para validar los números y las letras

### Ejercicio 05 (b5ej05.php)

Crear una función llamada **checkMail(\$email)** que reciba una cadena y devuelva true o false si cumple con el siguiente formato:

xxxxxxxx@xxxxxxxx.es

donde x son letras, números o los caracteres guión (-), guión bajo (\_) o punto (.)

**Nota:** Puedes usar la función **filter\_var** de PHP.

### Ejercicio 06 (b5ej06.php)

Crear una función llamada **sumArray(\$numeros)** que reciba un array de números y devuelva el resultado de sumar todos.

### Ejercicio 07 (b5ej07.php)

Partiendo de la práctica p07fotos.html, crea un archivo .php que contenga una función que recorra con **scandir** la carpeta **"/images"** y muestre todos los archivos que encuentre de forma dinámica.

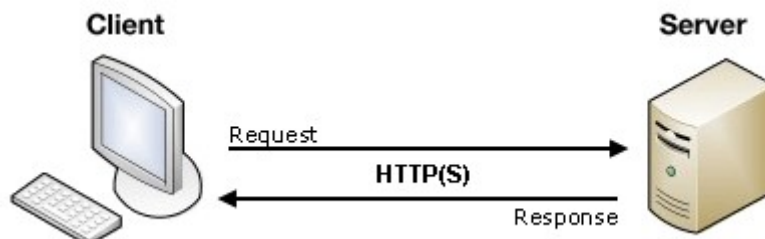
## 10. Formularios HTML

Los formularios HTML son uno de los puntos principales de interacción entre un usuario y un sitio web o aplicación. Ellos permiten a los usuarios enviar información a un sitio web. La mayor parte de las veces se envía información a un servidor web, pero la página web también puede interceptarla para usar los datos introducidos.

Un formulario HTML está hecho de uno o más campos de entrada. Estos pueden ser campos de texto (de una línea o multilínea), cajas de selección, botones, checkboxes, o botones de radio. La mayoría del tiempo, estos campos se sitúan junto a una etiqueta (label) que describe su propósito.

### 10.1 Estructura

Cuando usamos un formulario, y pulsamos en el botón de enviar los datos, se produce una comunicación con el servidor:



La sintaxis es:

```
<form
  action="[destino]"
  method="[tipo de envío]"
  enctype="[tipo MIME de datos]">
  [CONTENIDO DEL FORMULARIO]
</form>
```

Los atributos de la etiqueta **<form>** sirven para :

- **action:** indica el lugar donde se enviarán los datos recogidos en el formulario.
  - Por defecto, el valor del atributo es **action="#"**, y se enviará a la misma página que contiene el formulario.
- **method:** define cómo serán enviados los datos.
  - Las formas más comunes son GET y POST. El **method="GET"** envía los datos como si se añadieran a la URL, mientras que con **method="POST"** se envían dentro del body de la solicitud al servidor.
  - El valor por defecto es **method="POST"**
- **enctype:** especifica el valor del tipo de contenido.
  - El valor por defecto es **"application/x-www-form-urlencoded"** pero cuando añadimos entradas de ficheros (**<input type="file"/>**) deberemos indicar el valor **"multipart/form-data"** para que lo separe en varias partes.

## 10.2 Campos de un formulario

El cuerpo del formulario utilizará etiquetas para indicar cada campo de entrada. Las etiquetas campos también puede tener los atributos id, class y style como cualquier etiqueta.

### INPUT

La sintaxis para un campo que recoja caracteres alfanuméricos con sus atributos principales es:

```
<input
  type="[tipo de campo]"
  id="[identificador del campo]"
  name="[nombre del campo]"
  value="[valor del campo]"
  maxlength="[máximo número de caracteres]"
  title="[texto a mostrar al pasar el ratón por encima]"
  placeholder="[mensaje interior en el campo]"
  readonly
  required
/>
```

#### INPUT - Sintaxis

[https://www.w3schools.com/tags/tag\\_input.asp](https://www.w3schools.com/tags/tag_input.asp)

Los atributos de la etiqueta **<input>** sirven para :

- **type:** indica el tipo de campo y afectará a su comportamiento. Los posibles valores son "text", "number", "date", "email", "url", "password", "file", "checkbox", "radio", "button", ...

Para ver todos los tipos posibles consultar:

[https://www.w3schools.com/tags/att\\_input\\_type.asp](https://www.w3schools.com/tags/att_input_type.asp)

- **id:** este atributo sirve para identificar la etiqueta y utilizarlo para aplicarle CSS o javascript.
- **name:** este atributo servirá para luego recoger el dato de este campo.
- **value:** será el valor por defecto del contenido del campo, excepto para type="button" que es el texto que se muestra.
- **maxlength:** valor numérico que establece el máximo número de caracteres que puede contener el campo.
- **title:** texto a mostrar al pasar el ratón por encima
- **placeholder:** mensaje interior en el campo
- **readonly:** establece que el contenido del campo no es editable
- **required:** establece que el valor del campo no ser vacío

Algunos de los atributos sólo funcionarán en HTML5.

## **TEXTAREA**

La sintaxis para un campo que recoja caracteres alfanuméricos en un área de texto con sus atributos principales es:

```
<textarea
    rows="[número de filas]"
    id="[identificador del campo]"
    name="[nombre del campo]"
    maxlength="[máximo número de caracteres]"
    title="[texto a mostrar al pasar el ratón por encima]"
    placeholder="[mensaje interior en el campo]"
    readonly
    required
>[valor del campo]</textarea>
```

### **TEXTAREA - Sintaxis**

[https://www.w3schools.com/tags/tag\\_textarea.asp](https://www.w3schools.com/tags/tag_textarea.asp)

Los atributos que no aparecen en la etiqueta `<input>` y sí en la etiqueta `<textarea>` sirven para :

- **rows**: indica el número de filas que ocupará la caja del campo.

## **SELECT**

La sintaxis para un campo que recoja caracteres alfanuméricos es:

```
<select
    id="[identificador del campo]"
    name="[nombre del campo]"
    title="[texto a mostrar al pasar el ratón por encima]"
>
    <option value="[valor de la opción 1]" selected>
        [Texto de la opción 1]
    </option>
    <option value="[valor de la opción 2]">
        [Texto de la opción 2]
    </option>

</select>
```

### **INPUT - Sintaxis**

[https://www.w3schools.com/tags/tag\\_select.asp](https://www.w3schools.com/tags/tag_select.asp)

El atributo **selected** es para indicar la opción seleccionada.

## **RADIO**

Para indicar que una etiqueta input es un "Radio Button", es decir, un botón con selección de opción a elegir, deberemos asignar **type="radio"** y el mismo valor en **name** para todas las opciones. Por ejemplo:

```
<input type="radio" id="idradio1" name="btnRadio" checked>Opción 1  
<input type="radio" id="idradio2" name="btnRadio">Opción 2  
<input type="radio" id="idradio3" name="btnRadio">Opción 3
```

El atributo **checked** es para indicar la opción marcada por defecto.

## **CHECKBOX**

Para indicar que una etiqueta input es un "Check Box", es decir, un botón con selección de check, deberemos asignar **type="checkbox"**. Por ejemplo:

```
<input type="checkbox" id="idcheck1" name="btnCheck"> Opción
```

Esta etiqueta también puede llevar el atributo **checked** para indicar que aparezca marcada por defecto

## **FILE**

Para indicar que una etiqueta input es un "Check Box", es decir, un botón con selección de check, deberemos asignar **type="checkbox"**. Por ejemplo:

```
Elegir Fichero <input type="file" id="idFile" name="btnFile">
```

## **SUBMIT**

El botón de tipo submit es que realiza la función de enviar los datos:

```
<input type="submit"  
      id="idSubmit" name="btnSubmit"  
      value="Enviar">
```

En este botón, el atributo **value** contiene el texto que aparece en el botón.

## 10.3 Rótulos par los campos

Para poder disponer de un rótulo para los campos, se recomienda usar la etiqueta `<label>`.

```
<label for="[name del input]">[Texto del rótulo]</label>
```

El atributo `for` sirve para que al pulsar sobre el **label** con el ratón, el cursor se coloque en el campo.

Las etiquetas `<label>` también puede tener los atributos **id**, **class** y **style** como cualquier etiqueta.

### ***Práctica p06 (p06form → operaciones\_javascript.html)***

Se desea crear una página web que reciba un parámetro numérico entero "desde" y otro "hasta" y nos muestre:

- el rango indicado (desde-hasta)
- el sumatorio de todos los número comprendidos entre desde y hasta
- la multiplicación de desde por hasta
- la media aritmética de desde y hasta

Para ello, usaremos programación en cliente con Javascript

### ***Práctica p06 (p06form → operaciones\_jquery.html)***

Igual que el anterior, pero usaremos programación en cliente con JQuery

## 11. Formularios con PHP

Como se ha podido comprobar, podemos realizar programación en el cliente/navegador con Javascript/Jquery o también en servidor con PHP.

En la mayoría de ocasiones, dependiendo de lo que se desea realizar, se deberá desarrollar en el cliente o en el servidor teniendo en cuenta el origen de los datos para nuestro código.

Por ejemplo:

- Se desea mostrar un mensaje al coger el foco un input de HTML
  - es necesario realizarlo en Javascript/JQuery porque depende de una acción del usuario que se desarrolla en el cliente
- Se desean mostrar unos datos que están en una Base de Datos del servidor
  - debe realizarse en el servidor ya que es el que tiene acceso

Pero hay otros casos en los que se puede realizar la misma acción en el cliente o en el servidor. En estos casos, deberemos tener en cuenta la potencia del cliente para realizar procesos y la velocidad de la red para comunicar con el servidor, pues serán los dos parámetros que nos permitirán elegir.

Cuando preparamos un formulario deberemos indicar la siguiente línea:

```
<form
  id="idFormulario"
  name="nombreFormulario"
  action="pagina.php"
  method="POST"
  enctype="multipart/form-data"
>
```

Al recibir el archivo **pagina.php** el control del formulario, podremos procesar los datos recogidos con el array **\$\_POST** o **\$GET** dependiendo del método indicado en la etiqueta **<form>**.

### **Práctica p06 (p06form → operaciones\_php.php) (MUY IMPORTANTE)**

Se desea crear una página web que reciba un parámetro numérico entero "desde" y otro "hasta" y nos muestre:

- el rango indicado (desde-hasta)
- el sumatorio de todos los número comprendidos entre desde y hasta
- la multiplicación de desde por hasta
- la media aritmética de desde y hasta

Para ello, usaremos:

- programación en cliente con Javascript para gestionar los eventos y
- programación en servidor con PHP para el proceso de cálculo y mostrar el resultado

## 11.1 Ejercicios de Formularios con PHP

### Ejercicio 01 (b6ejer01.php) (MODELO EXAMEN)

#### HABITACIONES

Crear una página en PHP que nos indique el precio de una habitación de un hotel en base a las siguientes restricciones.

El formulario recogerá

- la temporada (ALTA, MEDIA o BAJA),
- el número de noches y
- el tipo de habitación (VISTAS o INTERIOR).

El precio base por noche de una habitación con vistas es de 80 € y el de la interior es de 60 €.

En temporada ALTA se aplica un incremento del 20% al precio base y en la BAJA un descuento del 10%.

Pasos:

- Controlar que existen todos los parámetros y mostrar un mensaje de error en caso contrario.
- Controlar que el valor de los parámetros sea correcto y mostrar mensajes de error en caso contrario.
- Que calcule y muestre **el precio** teniendo en cuenta los valores introducidos en el formulario o los recibidos por parámetro.

A tener en cuenta:

- El cálculo ya se ha realizado en el ejercicio **b2ejer03.php**
- El método del formulario puede ser **GET para recibirlo por parámetros**. Como en el ejercicio **b2ejer03.php**
- Para realizarlo, puedes utilizar como ejemplo el código PHP incluido en **operaciones\_php.php**

### Ejercicio 02 (b6ejer02.php) (MODELO EXAMEN)

#### PELIGROSIDAD

Crear una página en PHP que nos indique el grado de peligrosidad del mar en base a los siguientes valores de altura de olas, dirección del viento y temperatura:

Altura de las Olas	Dirección del viento	Temperatura	Peligrosidad
< 1 m	E, S	< 20	4
< 1 m	E, S	>= 20	3
< 1 m	O, N	< 20	2
< 1 m	O, N	>= 20	1
>= 1 m	E, S	< 20	5
>= 1 m	E, S	>= 20	4
>= 1 m	O, N	< 20	3
>= 1 m	O, N	>= 20	2



El formulario recogerá

- altura de las olas (valor entero),
- dirección del viento ('N', 'S', 'E' u 'O') y
- temperatura (valor entero).

Pasos:

- Controlar que existen todos los parámetros y mostrar un mensaje de error en caso contrario.
- Controlar que el valor de los parámetros sea correcto y mostrar mensajes de error en caso contrario.
- Que calcule y muestre el **grado de peligrosidad** teniendo en cuenta los valores introducidos en el formulario o los recibidos por parámetro.

A tener en cuenta:

- El cálculo ya se ha realizado en el ejercicio **b2ejer04.php**
- El método del formulario puede ser **GET para recibirlo por parámetros**. Como en el ejercicio **b2ejer04.php**
- Para realizarlo, puedes utilizar como ejemplo el código PHP incluido en **operaciones\_php.php**

### **Ejercicio 03 (b6ejer03.php) (MODELO EXAMEN)**

#### **MONEDAS**

Crear una página en PHP que nos informe de las monedas necesarias para cubrir una cantidad de euros intentando seleccionar las monedas de mayor importe posible.

La página debe mostrar un mensaje donde se indique la mínima cantidad de cada tipo de moneda (2€, 1€, 50c, 20c, 10c, 5c, 2c, 1c) para formar la cantidad indicada.

El formulario recogerá

- el importe en euros (valor entero que debe ser igual o inferior a 5€)

Pasos:


- Controlar que existen todos los parámetros y mostrar un mensaje de error en caso contrario.
- Controlar que el valor de los parámetros sea correcto y mostrar mensajes de error en caso contrario.
- Que calcule y muestre **las monedas necesarias** teniendo en cuenta los valores introducidos en el formulario o los recibidos por parámetro.

A tener en cuenta:

- El cálculo ya se ha realizado en el ejercicio **b2ejer05.php**
- El método del formulario puede ser **GET para recibirlo por parámetros**. Como en el ejercicio **b2ejer05.php**
- Para realizarlo, puedes utilizar como ejemplo el código PHP incluido en **operaciones\_php.php**

**Ejercicio 04 (b6ejer04.php) (MODELO EXAMEN)****GEOGRAFÍA**

Añadir al proyecto de geografía (montañas, lago, ...) una opción mas de valles, en la estructura dinámica con los siguientes datos:

- Opción de menú: "Valles"
- Rótulo del contenido: "  Valles"
- Texto del contenido: "

**Los valles son áreas geográficas que se caracterizan por ser terrenos bajos, generalmente situados entre montañas o colinas. Se forman a través de procesos geológicos como la erosión, el hundimiento de la tierra o la actividad glacial.**

**Los valles pueden tener diversas formas y tamaños, y se pueden clasificar en diferentes tipos, como valles en forma de U, que son típicos de la acción glacial, y valles en forma de V, que son el resultado de la erosión fluvial.**

"

Además de hacer la nueva página de "Valles", se debe modificar el menú en todas las páginas, lo que puede comprobarse en el esquema estático.

En el esquema dinámico, deberéis modificar **index.php**, **include\_nav.php** y crear **section\_valles.php**.

**Ejercicio 05 (b6ejer05.php) (MODELO EXAMEN)****COCHES**

Partiendo de un diseño estático, crear una estructura de programación en servidor con PHP para gestionarlo de forma dinámica, según el ejemplo de la práctica p03includes, es decir, crear una estructura dinámica con **index.php** y las carpetas **inc** y **content**.

Tened en cuenta que los enlaces entre páginas cambiarán:

- empresa → alquiler
- alquiler → contacto
- descuentos → alquiler y contacto

## 12. Debugger

Para que funcione, debemos haber instalado Xdebug en XAMPP.

En la instalación portable facilitada y disponible en SOFT ya se encuentra instalada esta extensión.

Podemos ejecutar desde un terminal CMD la información de PHP para comprobar que Xdebug está instalado y funcionando.

```
C:\SERVIDORES\xampp\php> php -i | find "xdebug"
```

Aparecerán múltiples líneas de configuración de xdebug.

Si no aparece nada es que no se ha instalado correctamente.

Ahora comprobaremos el funcionamiento desde el IDE (VSCode o NetBeans)

1. Abrir IDE
2. Añadir breakpoint en código PHP
3. Ejecutar con debug

Tenéis explicación en AULES del uso del debugger con PHP en **NetBeans** en el archivo:

**UD2-0 NetBeans - Instalación portable para PHP.pdf**

Tenéis explicación en AULES del uso del debugger con PHP en **VSCode** en el archivo:

**UD2-0 VSCode - Instalación portable para PHP.pdf**