

The background features a soft-focus, abstract composition. On the right side, there is a metallic, reflective sphere. Below it, a large, glowing orange shape, resembling a liquid drop or a flame, extends towards the bottom right corner. The overall color palette is light and airy, with a gradient from white to pale blue and yellow.

Creación formularios  
mediante PHP.

# ¿Qué es un formulario?

- Son elementos de las páginas web que **permiten recoger valores del usuario** y enviarlos a una URL (una página php) **para ser procesados**.
- HTML proporciona una gran **variedad de controles** para almacenar dicha información. (cuadro de texto, check, cuadro combinados..etc)
- Todos los controles deben ubicarse dentro de un elemento **Form**.
- Se pueden utilizar dos métodos para enviar dicha información, **GET y POST**.

```
<html>
<body>
<form action="formpost.php" method="post">
  Nombre: <input type="text" name="nombre"><br>
  Email: <input type="text" name="email"><br>
  <input type="submit" value="Enviar">
</form>
</body>
</html>
```



```
<html>
<body>
  Hola <?php echo $_POST["nombre"]; ?><br>
  Tu email es: <?php echo $_POST["email"]; ?>
</body>
</html>
```

# Métodos GET y POST

- **\$\_GET** y **\$\_POST** son variables superglobales que forman arrays de *keys* y *values*. El hecho de que sean **superglobals** hace que sean accesibles desde el script independientemente del ámbito.
- **\$\_GET** es un array de variables que se pasan al script a través de los **parámetros de URL**. La información que se envía es visible para todo el mundo y tiene limitada la cantidad de información que se puede enviar a 2000 caracteres.
- **\$\_POST** es un array de variables que se pasan al script a través del método **HTTP POST**. La información que se envía no es visible para los demás ya que los nombres y variables van embebidas en el **body** del **HTTP request**. No tiene límites en la cantidad de información a enviar. Soporta datos de entrada **multi-part binary** a la hora de subir archivos al servidor (lo que permite [subir múltiples archivos a la vez](#)).

# Tipos de controles más utilizados

Elemento	Descripción
input type="text"	Caja de texto
input type="password"	Caja de texto donde se muestran asteriscos en lugar de los caracteres escritos
input type="checkbox"	Cajas seleccionables que permite escoger múltiples opciones
input type="radio"	Cajas seleccionables en grupos que sólo permiten escoger una opción
input type="submit"	Botón para enviar el formulario
input type="file"	Cajas de texto y botón que permite subir archivos
input type="hidden"	Elemento escondido. Especialmente útil para tokens de seguridad
option	Una opción posible dentro de un elemento element
select	Lista de opciones de elementos option
textarea	Texto multilínea

Ejemplos de como utilizar los controles enumerados



# Validaciones de formularios

- ✓ La **validación de formularios** es algo fundamental ya que previene posibles ataques de intrusos y nos permite asegurar que los datos que se reciben son realmente del tipo deseado.
- ✓ Existen **dos formas de validación de formularios**: en el lado del cliente (JavaScript, más rápido) y en el lado del servidor (PHP, más costoso para el servidor).
- ✓ Podemos utilizar el atributo **maxlength** limita el número de caracteres para una caja de texto.
- ✓ Podemos validar los datos en el mismo script incluyendo en el action **<form action="<?php \$\_SERVER['PHP\_SELF']; ?>" method="post">**
- ✓ Esto lo que hace que los datos se envíen al mismo archivo, en lugar de llevarlos a otro archivo para tratarlos.

[Ejemplos de posibles validaciones.](#)

# Subir ficheros con formularios POST

- Debemos configurar las directivas `files_uploads`, `upload_tmp_dir`, `upload_max_filesize` del fichero `php.ini` vinculado al servidor apache.
- Es necesario que los permisos tanto del **directorio temporal** como del **directorio final** sean de escritura.
- Para que un formulario tenga la capacidad de aceptar archivos se añade el atributo **`enctype="multipart/form-data"`** al elemento `form`.
- Lo más laborioso a la hora de hacer un formulario que permita la subida de archivos es **manejarlos y organizarlos una vez están en el servidor**. Inicialmente los archivos se suben a un directorio temporal y luego se relocalizan.

# Subir ficheros con formularios POST

- La información sobre un archivo subido la proporciona el [array multidimensional](#) **\$\_FILES**
- **\$\_FILES["nombre\_input"]["name"]**. Guarda el nombre original del archivo del cliente.
- **\$\_FILES[" nombre\_input "]["type"]**. Guarda el MIME type del archivo.
- **\$\_FILES[" nombre\_input "]["size"]**. Guarda el tamaño del archivo en bytes.
- **\$\_FILES[" nombre\_input "]["tmp\_name"]**. Guarda el nombre del archivo temporal.
- **\$\_FILES[" nombre\_input "]["error"]**. Guarda cualquier código de error que pueda provocar la subida del archivo.
- La función *move\_uploaded\_file()* mueve un archivo subido del directorio temporal al directorio que se indique.

# Subir ficheros con formularios POST

- El array **`$_FILES['imagen']['error']`** especifica por qué no se ha podido subir el archivo, lo que permite especificar un mensaje de vuelta para cada tipo de error. Devuelve un ***integer*** con el número de error:

Error	Valor	Significado
UPLOAD_ERR_OK	0	No hay errores
UPLOAD_ERR_INI_SIZE	1	Supera el tamaño máximo indicado en php.ini
UPLOAD_ERR_FORM_SIZE	2	Supera el tamaño máximo indicado en MAX_FILE_SIZE de html
UPLOAD_ERR_PARTIAL	3	Sólo se ha subido el archivo parcialmente
UPLOAD_ERR_NO_FILE	4	No se ha subido ningún archivo
UPLOAD_ERR_NO_TMP_DIR	6	Falta la carpeta temporal
UPLOAD_ERR_CANT_WRITE	7	No se puede escribir en el directorio especificado
UPLOAD_ERR_EXTENSION	8	Una extensión de PHP ha detenido la subida



# Subir ficheros con formularios POST. Ejemplo.

```
<html>
<body>
<h2>Formulario subida de archivos</h2>
<form action="<?php $_SERVER["PHP_SELF"]; ?>" method="POST" enctype="multipart/form-data">
    <input type="file" name="ficherotxt" />
    <input type="submit" name="submit" />
</form>
<?php
$directorioSubida = "ficheros/";
if(isset($_POST["submit"]) && isset($_FILES["ficherotxt"])){
    /*
    echo "<pre>";
    print_r($_FILES);
    echo "</pre>";
    */
    $nombreArchivo = $_FILES['ficherotxt']['name'];
    $filesize = $_FILES['ficherotxt']['size'];
    $directorioTemp = $_FILES['ficherotxt']['tmp_name'];
    $tipoArchivo = $_FILES['ficherotxt']['type'];

    $arrayArchivo = pathinfo($nombreArchivo);
    //print_r($arrayArchivo);
    $extension = $arrayArchivo['extension'];

    $nombreArchivo = $arrayArchivo['filename'];
    $nombreCompleto = $directorioSubida.$nombreArchivo.".".$extension;
    //echo $nombreCompleto;
    move_uploaded_file($directorioTemp, $nombreCompleto);
    print "El archivo se ha subido correctamente";
}
?>
</body>
</html>
```

# Redirecciones con PHP

- Los direccionamientos son encabezados HTTP que deben ser enviados antes que cualquier otro tipo de contenido en una página web.
- Pueden ser temporales o definitivos.
- La forma de realizar **redirecciones en PHP** es mediante la función `header()`.
- *En las redirecciones se pueden añadir pares claves/valor que serían recogidos mediante el método GET.*

```
1. $nuevaURL = 'https://www.srorigofuente.es';  
2. header('Location: ' . $nuevaURL);
```

```
1. <?php  
2. header("HTTP/1.1 301 Moved Permanently");  
3. $nuevaURL = 'https://www.srorigofuente.es';  
4. header('Location: ' . $nuevaURL);  
5. ?>
```

# El tipo fecha en PHP.

## **string date ( format [timestamp] )**

Con esta función podemos obtener la **fecha local del sistema** formateada para obtener el día, mes, semana, año y hora actual.

- La función nos devuelve un string.
- **format (Obligatorio)**: Es un string donde le pasaremos el formato de datos que queremos obtener.
- **timestamp (Opcional)**: Este valor es un número entero que representa una fecha concreta, si se omite el valor por defecto es la de la fecha del sistema ([time\(\)](#)).

## **string setlocale (constant, location)**

Se puede establecer también el idioma local para el formato de la fecha, pero tan solo lo cambia en el script que ese está ejecutando.

- **constant (Obligatorio)**: Es la información local que queremos especificar.
- **location (Obligatorio)**: Valor de la información local especificada, este valor también puede ser un [array](#).

[Constantes disponibles\(link\)](#)

# El tipo fecha en PHP.

Definimos el idioma por defecto y probamos varios formatos con la función date().

```
Setlocale(LC_ALL,"es_ES");  
// Día del mes con 2 dígitos, y con ceros iniciales, de 01 a 31  
date("d");  
// Día del mes, sin ceros iniciales, de 1 a 31  
date("j");  
// Día de la semana en inglés, con 3 letras, de Mon a Sun  
date("D");  
// Día de la semana en inglés, de Sunday a Saturday  
date("l");  
// del día de la semana, desde 1 (lunes) hasta 7 (domingo)  
date("N");  
// Sufijo del día del mes con 2 caracteres --> st, nd, rd o th  
date("S");  
// Número entero que representa el día de la semana, de 0 (dom) a 6  
// (sab)  
date("w");
```

```
Setlocale(LC_ALL,"es_ES");  
// Día del año, de 0 a 365  
date("Z");  
// Mes actual en inglés, de January a December  
date("F");  
// Mes actual en 2 dígitos y con 0 en caso del 1 al 9, de 1 a 12  
date("m");  
// Mes actual en texto en 3 dígitos en inglés, de Jan a Dec  
date("M");  
// Mes actual en dígitos sin 0 inicial, de 1 a 12  
date("n");  
// Número de días del mes actual, de 28 a 31  
date("t");  
// Número de días del mes actual, de 28 a 31  
date("T");
```



# El tipo fecha en PHP.

Como obtener la hora actual.

```
// Antes del mediodía, despues del mediodía, am o pm (minúsculas)
date("a");
// Antes del mediodía, despues del mediodía, AM o PM (mayúsculas)
date("A");
// Horario de 12 horas sin ceros, de 1 a 12
date("g");
// Horario de 12 horas con ceros, de 01 a 12
date("h");
// Horario de 24 horas sin ceros, de 0 a 23
date("G");
// Horario de 24 horas con ceros, de 01 a 23
date("H");
// minutos con ceros iniciales
date("i");
// segundos con ceros iniciales
date("s");
```

Otros formatos predefinidos.

```
//formato para RSS
date(DATE_RSS);
//formato W3C
date(DATE_W3C);
//formato para COOKIES
date(DATE_COOKIE);
//formato para ATOM
date(DATE_ATOM);
```

# El tipo fecha en PHP.

La función [strftime\(\)](#) permite realizar formateos con información local del servidor.

```
1 //Establecer la información local en castellano de España
2 setlocale(LC_TIME,"es_ES");
3 echo strftime("Hoy es %A y son las %H:%M");
4 echo strftime("El año es %Y y el mes es %B");
5
```

[Mas formatos con strftime.](#)

## **Comparar, sumar y restar fechas**

- [Cómo comparar fechas en PHP de manera fácil con date\(\) y strtotime\(\)](#)
- [Cómo sumar y restar fechas en PHP con strtotime\(\) y date\(\)](#)