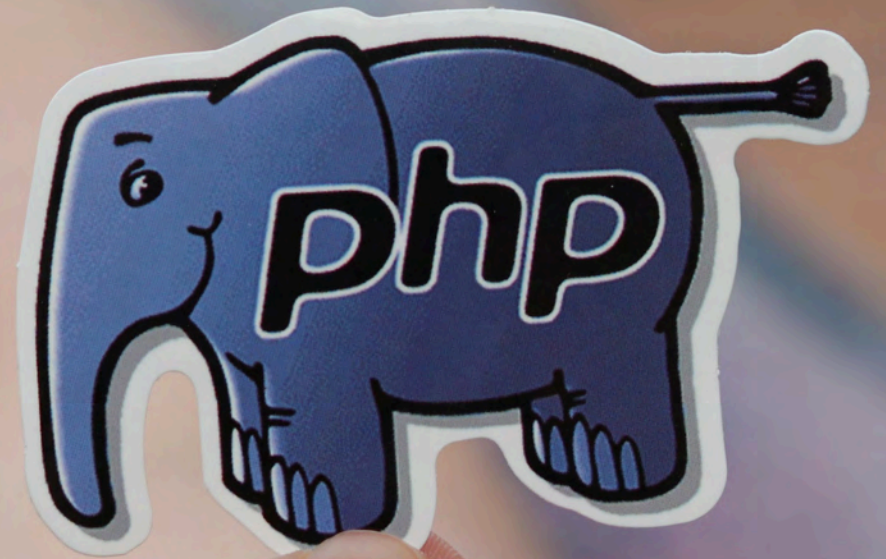


Bloque B

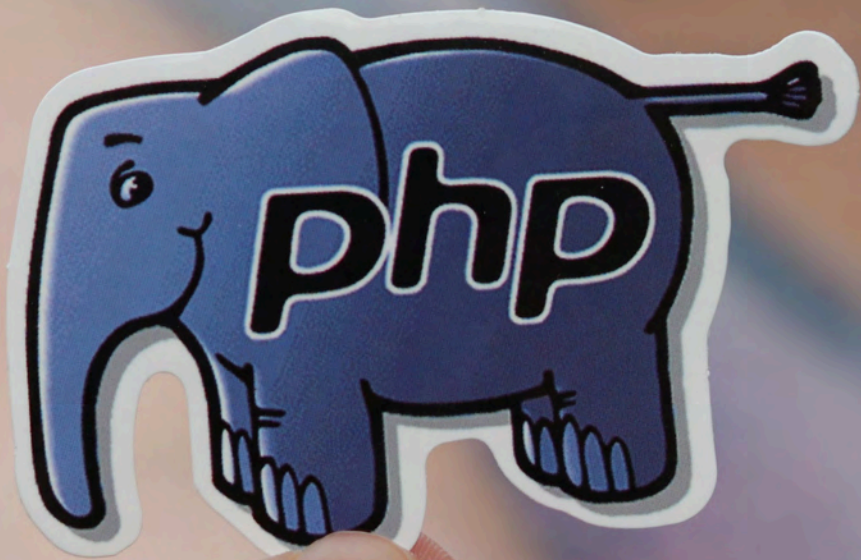
Páginas Web Dinámicas

Unidad 7. Imágenes & Archivos



Contenidos

1. Introducción
2. Subiendo ficheros desde un navegador
3. Recibiendo archivos en el servidor
4. Moviendo un archivo a su destino
5. Limpiando un nombre de archivo y archivos duplicados
6. Validando el tamaño y tipo de un fichero
7. Redimensionar imágenes
8. Recortar imágenes
9. Editando imágenes mediante extensiones



1. Introducción

Introducción

Esta unidad muestra cómo permitir a los visitantes **subir imágenes al servidor y cómo mostrarlas de forma segura en tus páginas PHP**. Estas técnicas **también funcionan para otros tipos de archivos**.

Primero, aprenderás **cómo los usuarios suben imágenes y cómo el servidor las recibe**. Veremos cómo:

- El control de subida de archivos HTML se utiliza en un formulario HTML para que los usuarios puedan subir archivos.
- El intérprete de PHP agrega datos sobre el archivo a un array superglobal llamado `$_FILES`.
- El archivo se coloca en una carpeta temporal en el servidor.
- El archivo debe ser movido a una carpeta donde se almacenarán los archivos subidos.

Introducción

A continuación, aprenderemos a validar los archivos que se han subido y comprobar que:

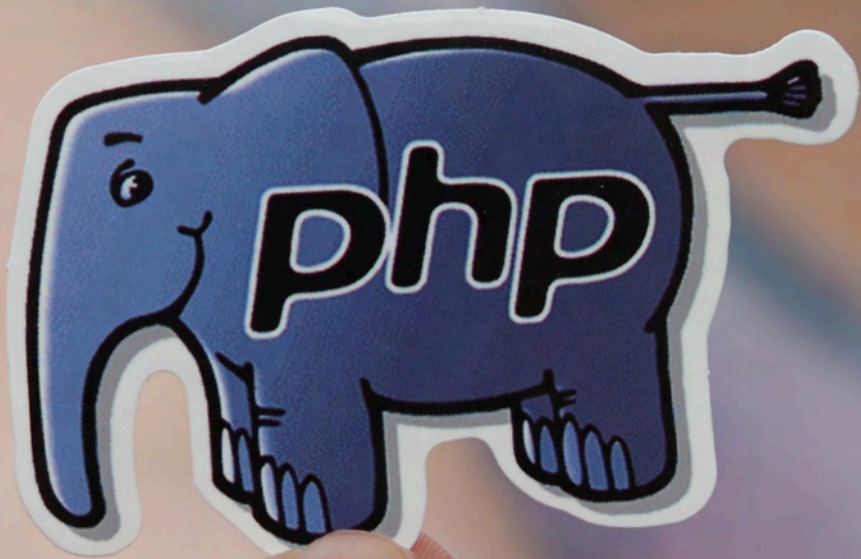
- El nombre del fichero sólo contiene caracteres permitidos.
- No existe ya un archivo con ese nombre.
- Es un tipo de medio y extensión de archivo permitidos.
- El tamaño del archivo no es demasiado grande.

Introducción

Por último, descubrirás cómo manipular las imágenes para crear:

- Miniaturas de la imagen.
- Versiones recortadas de la imagen.

A lo largo de la unidad, conocerás más funciones integradas que te ayudarán con estas tareas. Aunque la unidad demuestra estas técnicas utilizando imágenes, también pueden utilizarse para permitir a los visitantes cargar archivos de audio, vídeo, PDF y de otros tipos.



2. Subiendo ficheros desde un navegador

Subiendo ficheros desde un navegador

Los formularios HTML pueden contener un **control de carga de archivos**, que los visitantes pueden utilizar para cargar archivos en el servidor.

Al crear un formulario HTML que permita a los visitantes cargar archivos, la etiqueta `<form>` de apertura debe tener los tres atributos siguientes:

- `method` con un valor de *POST* para especificar que **el formulario debe enviarse mediante HTTP POST** (porque los archivos no deben enviarse mediante HTTP GET)
- `enctype` con valor *multipart/form-data* para especificar el **tipo de codificación** que debe utilizar el navegador para enviar los datos
- `action` cuyo valor es **el archivo PHP al que deben enviarse los datos del formulario**

Subiendo ficheros desde un navegador

El control de carga de archivos se crea utilizando el elemento HTML `<input>`. Su atributo `type` debe tener el valor *file*. En el navegador, esto crea un botón que abre una nueva ventana que permite al usuario seleccionar el archivo que desea subir:

```
<input type="file" name="image">
```

Al igual que los otros controles de formulario, el control de entrada del archivo envía un par nombre/valor al servidor:

- El **nombre** es el valor del atributo `name` para ese control de archivo (en el ejemplo anterior se llama *image*).
- El **valor** es el archivo que el usuario está enviando.

Subiendo ficheros desde un navegador

Para ayudar a restringir el tipo de archivo que un usuario puede cargar, el control de entrada de archivos tiene un atributo `accept` .

Su valor debe ser una lista separada por comas de los tipos de medios que el sitio acepta. (Los tipos de medios suelen denominarse tipos MIME, para más información consultar el *Anexo I de la Unidad 5*).

```
<input type="file" name="image" accept="image/jpeg, image/png">
```

Subiendo ficheros desde un navegador

Si se utiliza el atributo `accept`, cuando el visitante haga clic en el botón para subir un archivo, los navegadores modernos desactivarán los archivos que no estén en la lista de tipos aceptados para que no puedan ser seleccionados.

Esto es útil para la usabilidad, pero **no se puede confiar en el atributo `accept` para restringir el tipo de archivos que suben los visitantes** porque pueden anular la configuración y los navegadores más antiguos no lo soportan. (Chrome 10, Internet Explorer 10, Firefox 10 y Safari 6 fueron las primeras versiones de los principales navegadores en admitir esta función). Por lo tanto, también deberías intentar validar el tipo de medio en el servidor utilizando PHP (lo veremos en esta unidad).

Subiendo ficheros desde un navegador

Para permitir todos los subtipos de un tipo de medio, se puede añadir un asterisco en lugar de un subtipo.

Lo siguiente permite todos los formatos de imagen (incluyendo BMP, GIF, JPEG, PNG, TIFF y WebP):

```
<input type="file" accept="image/*">
```

Ejemplo: Subiendo ficheros desde un navegador

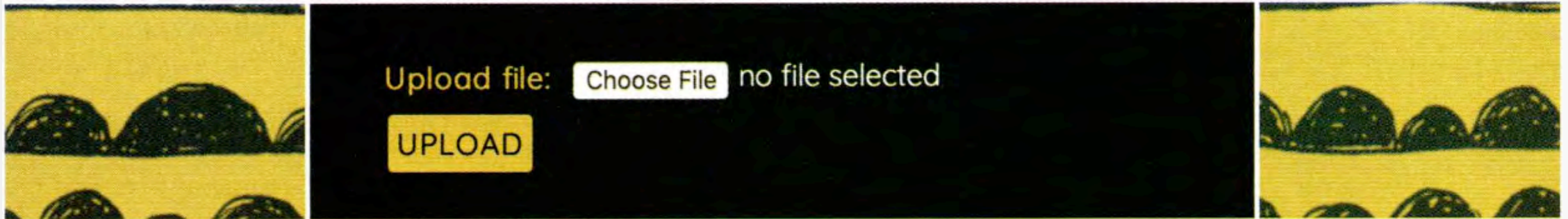
1. El siguiente formulario permite a los visitantes cargar imágenes. Se utiliza en todos los ejemplos de este capítulo. La etiqueta `<form>` de apertura necesita:
 - atributo `method` con el valor *POST*.
 - atributo `enctype` con el valor *multipart/form-data*.
 - atributo `action` especificando el archivo al que enviar los datos del formulario (este valor cambia en cada ejemplo).
2. Para crear un control de carga de archivos, el elemento `<input>` lleva un atributo `type` cuyo valor es *file*. Como los ejemplos de este capítulo muestran cómo permitir a los visitantes cargar una imagen, el valor del atributo `name` es *image*.
3. El botón *submit* se utiliza para enviar el formulario.

Ejemplo: Subiendo ficheros desde un navegador

```
① <form method="post" action="filename.php" enctype="multipart/form-data">  
    <label for="image"><b>Upload file:</b></label>  
② <input type="file" name="image" accept="image/*" id="image"><br>  
③ <input type="submit" value="Upload">  
</form>
```

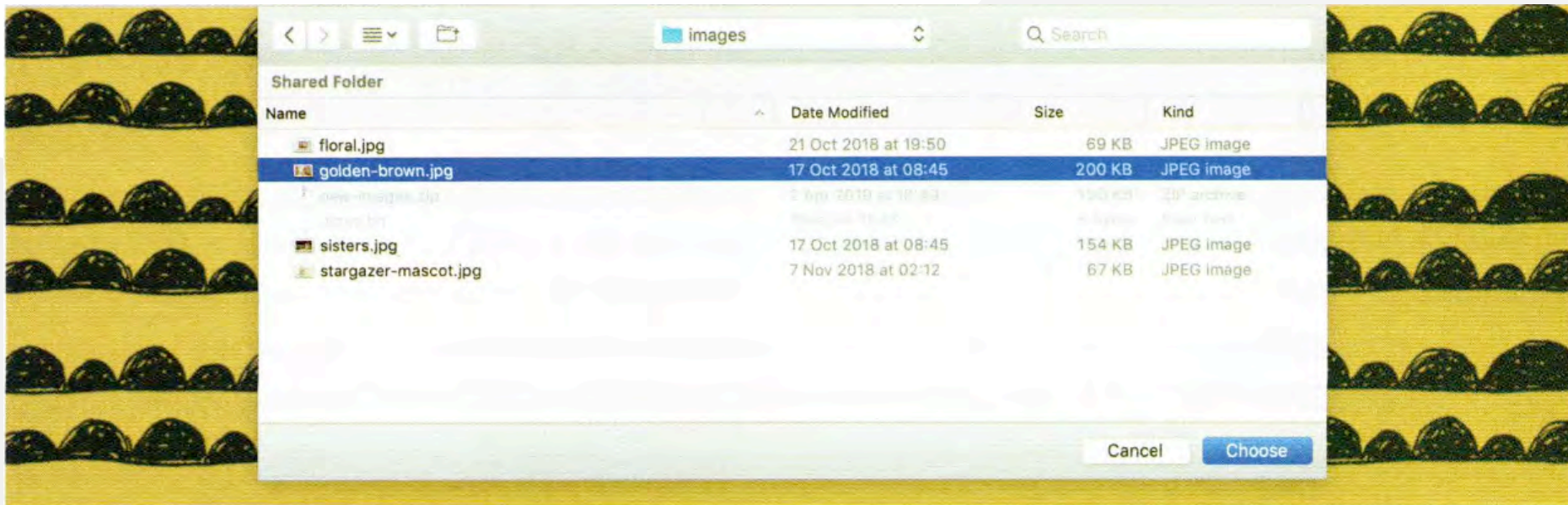

Ejemplo: Subiendo ficheros desde un navegador

La imagen siguiente muestra el formulario que crea el control de carga de archivos. Cuando se selecciona una imagen, el texto junto al botón se sustituye por el nombre del archivo.

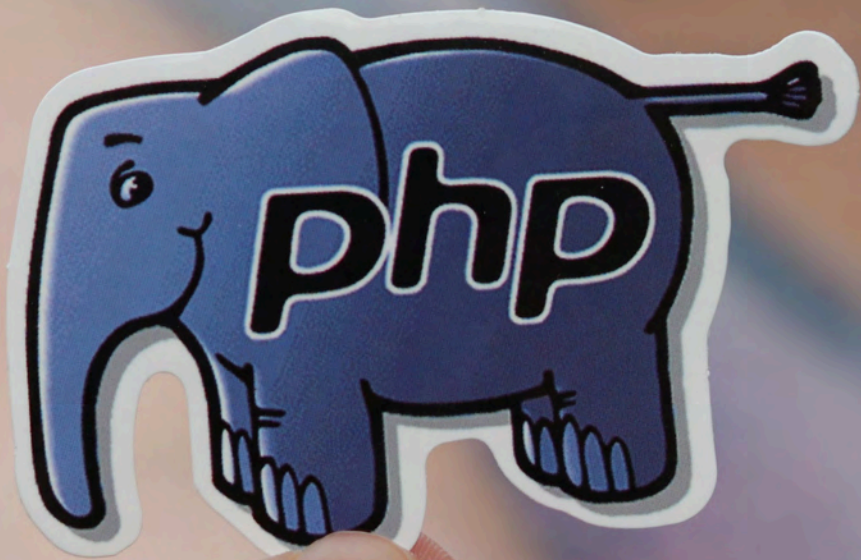


Ejemplo: Subiendo ficheros desde un navegador

Aquí se puede ver la ventana que se abre cuando el usuario hace clic en *Choose File*. Un archivo de texto y un archivo zip están **desactivados** porque no son imágenes.



El aspecto visual de la ventana que aparece para seleccionar archivos varía según el navegador y el sistema operativo (no es posible controlar su aspecto mediante CSS).



3. Recibiendo archivos en el servidor

Recibiendo archivos en el servidor

Cuando se carga un archivo a través de una página web, **el servidor web lo guarda en una carpeta temporal**, y el intérprete de PHP **almacena los detalles sobre el archivo en un array superglobal llamado `$_FILES`**.

Recibiendo archivos en el servidor

Un formulario puede tener múltiples controles de subida de archivos, por lo que el intérprete de PHP creará un **elemento en el array superglobal** `$_FILES` **por cada control de subida de archivos** que envíe el formulario.

El **nombre del elemento** coincide con el nombre del control de subida de archivos y su **valor** es un array de datos sobre el archivo que se subió a través de ese control de formulario.

Recibiendo archivos en el servidor

La siguiente tabla muestra la información que el array superglobal `$_FILES` almacena para cada archivo que se ha subido.

Las imágenes de esta unidad se suben utilizando un control de subida de archivos cuyo nombre es *image*, por lo que el array `$_FILES` tendrá un elemento llamado *image*, y su valor será un array que contendrá información sobre esa imagen.

KEY	VALUE	HOW TO ACCESS VALUE
<code>name</code>	File name	<code>\$_FILES['image']['name']</code>
<code>tmp_name</code>	Temporary location of the file (set by the PHP interpreter)	<code>\$_FILES['image']['tmp_name']</code>
<code>size</code>	Size in bytes	<code>\$_FILES['image']['size']</code>
<code>type</code>	Media type (according to the browser)	<code>\$_FILES['image']['type']</code>
<code>error</code>	0 if file uploaded successfully, an error code if there was a problem	<code>\$_FILES['image']['error']</code>

Recibiendo archivos en el servidor

Una vez que se ha subido un archivo, el código PHP debe comprobar que el intérprete PHP no ha encontrado ningún error con la subida.

Si la clave de error en el array que fue creado para este archivo tiene un valor de 0 entonces significa que el intérprete PHP no encontró ningún error.

```
if ($_FILES['image']['errors'] === 0) {  
    // Process image  
} else {  
    // Show error message  
}
```

Ejemplo: Comprobar que se ha cargado un archivo

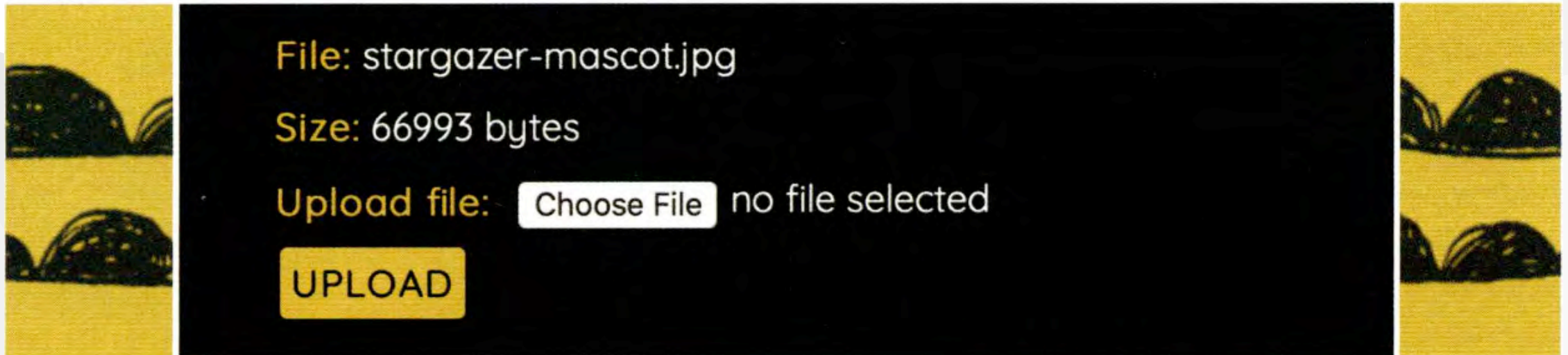
1. La variable `$message` se inicializa con una cadena en blanco. Almacena un mensaje si se envía el formulario.
2. Si el formulario se ha enviado utilizando *HTTP POST*...
3. Una sentencia *if* comprueba que no hay errores.
4. Si no hay errores, el nombre y el tamaño del archivo se almacenan en `$message` .
5. En caso contrario, `$message` almacena un mensaje de error.
6. Se muestra el valor de la variable `$message` .

Ejemplo: Comprobar que se ha cargado un archivo

```
<?php
① $message = ''; // Initialize
② if ($_SERVER['REQUEST_METHOD'] == 'POST') { // If form sent
③     if ($_FILES['image']['error'] === 0) { // If no errors
④         $message = '<b>File:</b> ' . $_FILES['image']['name'] . '<br>'; // File name
           $message .= '<b>Size:</b> ' . $_FILES['image']['size'] . ' bytes'; // File size
           } else { // Otherwise
⑤         $message = 'The file could not be uploaded.'; // Error message
           }
       }
    ?> ...

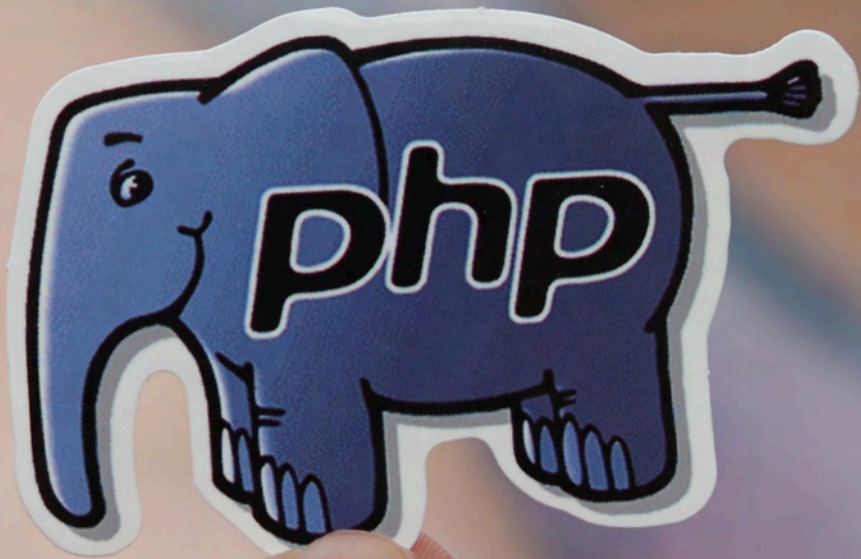
⑥ <?= $message ?>
   <form method="POST" action="upload-file.php" enctype="multipart/form-data">
       <label for="image"><b>Upload file:</b></label>
       <input type="file" name="image" accept="image/*" id="image"><br>
       <input type="submit" value="Upload">
   </form>
```

Ejemplo: Comprobar que se ha cargado un archivo



Actividad: Comprobar que se ha cargado un archivo

Un cliente te ha solicitado que crees una funcionalidad para un portal donde los usuarios puedan subir archivos. Como desarrollador, deberás asegurarte en tu implementación de que el fichero se ha subido correctamente al servidor. Si la subida ha sido exitosa, deberás mostrar un mensaje de confirmación.



4. Moviendo un archivo a su destino

Moviendo un archivo a su destino

La función `move_uploaded_file()` de PHP **mueve un archivo** de su ubicación temporal a donde debe ser almacenado en el servidor.

Moviendo un archivo a su destino

Cuando se carga un archivo en el servidor, se le asigna un nombre de archivo temporal y se coloca en una carpeta temporal. (El nombre de archivo temporal es creado por el intérprete PHP).

El intérprete de PHP borrará el archivo temporal de esta carpeta cuando el script termine de ejecutarse. Por lo tanto, para almacenar un archivo subido en el servidor, se debe llamar a la función `move_uploaded_file()` para moverlo a otra carpeta. Tiene dos parámetros:

- La **ubicación temporal** del archivo
- El **destino** donde debe guardarse el archivo

Devuelve *true* si pudo mover el archivo a la nueva ubicación y *false* en caso contrario.

Moviendo un archivo a su destino

El destino (la ubicación donde debe guardarse el archivo) está formado por:

- Ruta a la carpeta que almacenará el archivo subido (esta carpeta debe haber sido creada antes de intentar mover un archivo a ella).
- Nombre del fichero (su nombre original o un nombre nuevo).

Si se desea utilizar el nombre original del fichero que se ha subido, se puede acceder a él a través del array que el intérprete PHP ha creado para ese fichero. Su clave es name.

Moviendo un archivo a su destino

En el siguiente ejemplo, la ruta del archivo de destino se almacena en una variable llamada `$destination`. Se crea especificando la carpeta *uploads*, seguida del nombre original del archivo utilizado al subir la imagen.

```
$destination = '../uploads/' . $_FILES['image']['name'];  
move_uploaded_file($_FILES['image']['tmp_name'], $destination);
```

Diagram illustrating the components of the code:

- NEW FOLDER**: `../uploads/`
- FILENAME**: `$_FILES['image']['name']`
- TEMPORARY LOCATION**: `$_FILES['image']['tmp_name']`
- DESTINATION FILEPATH**: `$destination`

Moviendo un archivo a su destino

Permisos

Los permisos del directorio de destino deben:

- Permitir al servidor web leer/escribir archivos - esto le permite guardar y mostrar imágenes y otros tipos de ficheros subidos al servidor.
- Deshabilitar los permisos de ejecución: esto impide que se ejecuten secuencias de comandos maliciosas.

Moviendo un archivo a su destino

Verificar que un archivo fue subido

La función `move_uploaded_file()` de PHP verifica que un archivo fue subido vía HTTP POST antes de moverlo. Si necesitas utilizar un archivo antes de moverlo, hay que utilizar `is_uploaded_file()` de PHP para realizar esta comprobación.

Ejemplo: moviendo un archivo subido

1. Una variable llamada `$moved` se inicializa con un valor de false. Esto cambiará a true si la imagen se mueve con éxito.
2. Si el formulario fue enviado y no hubo errores...
3. `$temp` contiene la ubicación donde el intérprete PHP almacenó temporalmente el archivo.
4. `$path` almacena la ruta donde se guardará el archivo. (El archivo mantendrá el mismo nombre de archivo que tenía cuando fue subido).

Ejemplo: moviendo un archivo subido

5. `move_uploaded_file()` intenta mover el archivo desde su ubicación temporal (en `$temp`) a la nueva ubicación (en `$path`). La función devuelve `true` si funcionó o `false` si falló. Este valor reemplaza el valor almacenado en la variable `$moved` en el Paso 1.
6. Una sentencia condicional comprueba si `$moved` tiene el valor `true`, indicando que el movimiento funcionó.
- 7.
8. Si es así, `$message` almacena una etiqueta HTML `` que muestra la imagen subida.
9. En caso contrario, `$message` almacena un mensaje de error.
10. El valor almacenado en `$message` se muestra al usuario.

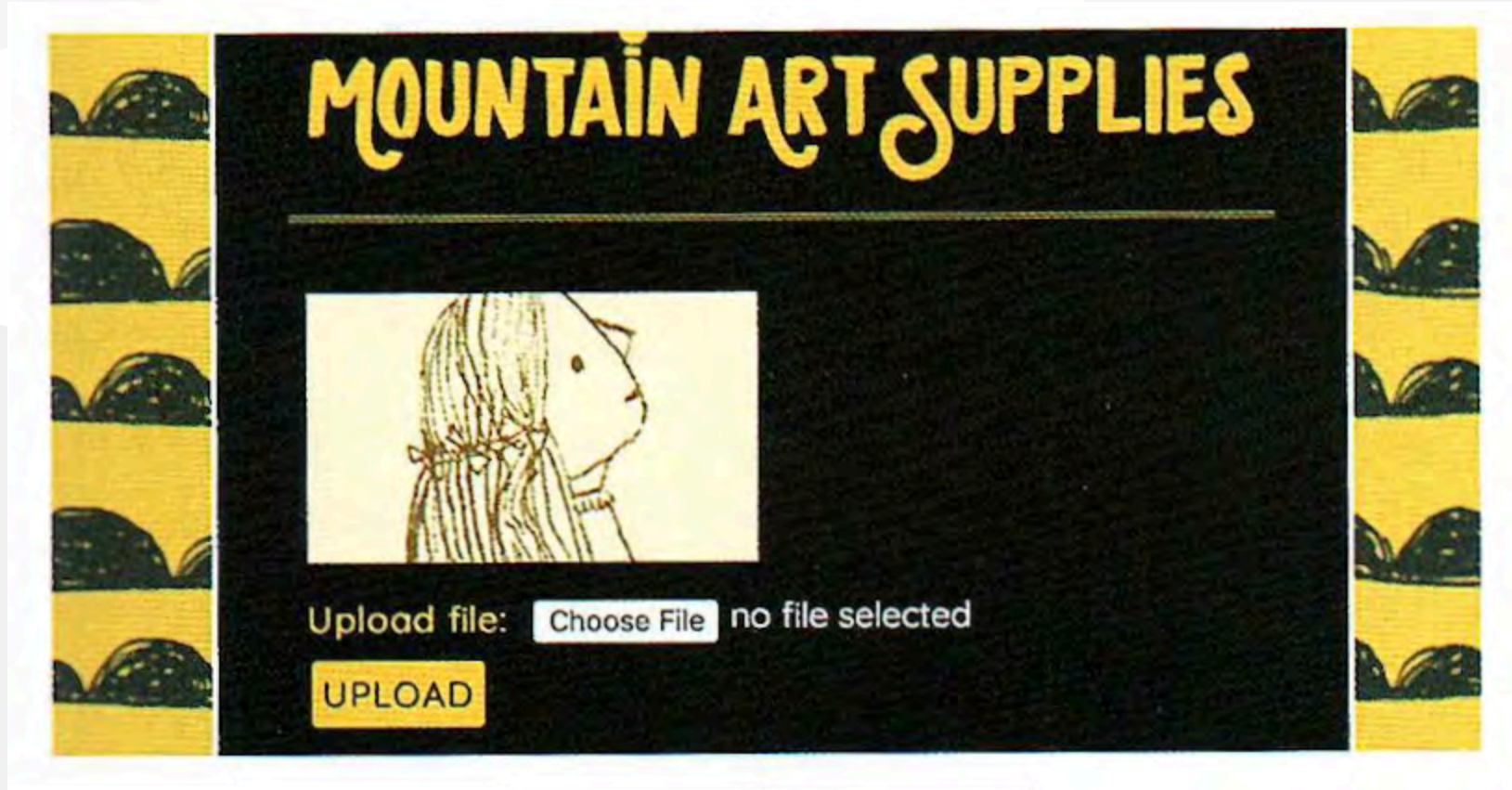
Ejemplo: moviendo un archivo subido

```
<?php
    $message = ''; // Initialize
    ① $moved = false; // Initialize

    ② [if ($_SERVER['REQUEST_METHOD'] == 'POST') { // If sent +
        if ($_FILES['image']['error'] === 0) { // No errors
            // Store temporary path and new destination
            ③ $temp = $_FILES['image']['tmp_name'];
            ④ $path = 'uploads/' . $_FILES['image']['name'];
            // Move the file and store result in $moved
            ⑤ $moved = move_uploaded_file($temp, $path);
        }

        ⑥ if ($moved === true) { // If move worked, show image
            ⑦ $message = '';
            ⑧ [ else { // Else store error message
                $message = 'The file could not be saved.';
            }
        }
    }
    ?> ...
    ⑨ <?= $message ?>
```

Ejemplo: moviendo un archivo subido



Actividad: Moviendo un archivo subido

Modifica la actividad anterior (*Actividad: Comprobar que se ha cargado un archivo*) para almacenar la imagen de perfil que el usuario suba en un directorio denominado *images* ubicado en */var/www/*. Comprueba que se ha realizado correctamente la operación y en caso afirmativo muestra la imagen almacenada en la página PHP (en caso contrario, muestra un mensaje de error).



5. Limpiando un nombre de archivo y archivos duplicados

Limpiando un nombre de archivo y archivos duplicados

Antes de mover un archivo de su ubicación temporal, debes:

1. **Eliminar los caracteres** del nombre del fichero **que puedan causar problemas.**
2. **Asegurarte de que no se sobrescribirá otro archivo** con el mismo nombre.

Limpiando un nombre de archivo y archivos duplicados

Los caracteres como los ampersands, los dos puntos, los puntos y los espacios deben eliminarse de los nombres de archivo, ya que pueden causar problemas. Para ello, se pueden sustituir los caracteres que no sean A-Z, a-z y 0-9, por un guión.

Limpiando un nombre de archivo y archivos duplicados

1. Utiliza la función `pathinfo()` de PHP, para obtener el nombre base y la extensión del archivo.
2. Utiliza la función `preg_replace()` de PHP para reemplazar cualquier carácter en el nombre base que no sea A-L, a-z, y 0-9, por un guión.
3. Crea la ruta del archivo de destino uniendo el directorio de subida, el nombre base, un punto y la extensión del archivo. Este valor debe guardarse en una variable.

```
① { $basename = pathinfo($filename, PATHINFO_FILENAME);  
    $extension = pathinfo($filename, PATHINFO_EXTENSION);  
② $basename = preg_replace('/[^A-z0-9]/', '-', $basename);  
③ $filepath = 'uploads/' . $basename . '.' . $extension;
```

Limpiando un nombre de archivo y archivos duplicados

Si se llama a `move_uploaded_file()` y existe un archivo con el mismo nombre, el archivo antiguo se sustituye por el nuevo. Para evitar esto, cada archivo necesita un nombre único:

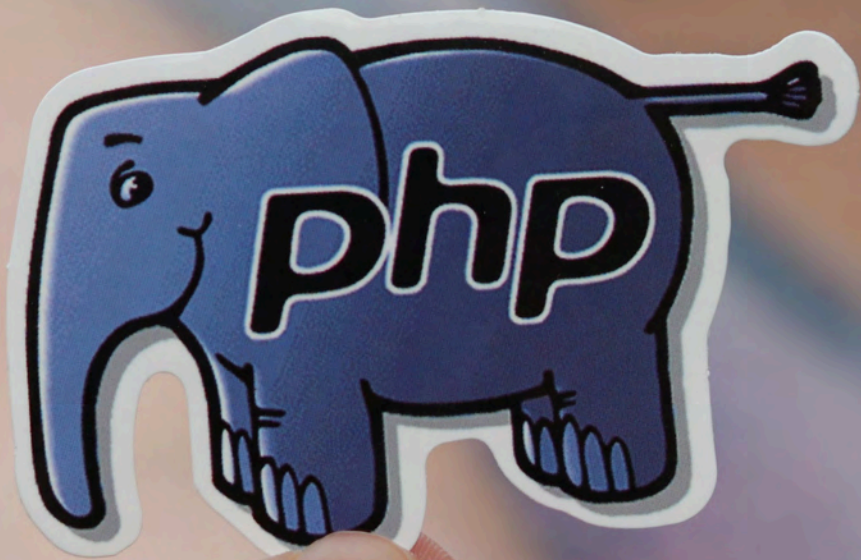
4. Pon un contador a **0** y guárdalo en una variable llamada `i`.
5. En la condición de un bucle while, utiliza la función `file_exists()` de PHP para comprobar si ya existe un archivo con el mismo nombre.
6. Si lo hace, suma 1 al valor almacenado en el contador.

Limpiando un nombre de archivo y archivos duplicados

7. Actualiza el nombre del fichero, añadiendo el valor en el contador después del nombre base, antes de la extensión. Por ejemplo, si *upload.jpg* existe, llama al archivo *upload1.jpg*.

La condición del bucle se ejecuta de nuevo para comprobar si existe el nuevo nombre de fichero. El bucle repite los pasos 5-7 hasta que tenga un único nombre de fichero.

```
④ $i = 1;
⑤ while (file_exists('uploads/' . $filename)) {
⑥     $i = $i + 1;
⑦     $filename = $basename . $i . '.' . $extension;
}
```



6. Validando el tamaño y tipo de un fichero

Validando el tamaño y tipo de un fichero

Para asegurarse de que un sitio puede trabajar con un archivo cargado, antes de moverlo, hay que comprobar:

1. El **archivo no es demasiado grande** (los archivos grandes tardan más en descargarse o procesarse).
2. El **sitio puede trabajar con el tipo de medio y la extensión** del archivo.

Se puede establecer un tamaño máximo de subida de archivos en *php.ini* o *.htaccess*, o crear código de validación para restringir los tamaños en las páginas que aceptan subidas.

Validando el tamaño y tipo de un fichero

Para ver si un archivo es mayor que el **tamaño máximo de subida** establecido en php.ini o .htaccess mira en el array `$_FILES` . Si lo es, la clave de error para ese archivo tendrá un valor de `1` .

También se puede **comprobar el tamaño de un archivo** en el array `$_FILES` ; la clave de tamaño del archivo contiene el tamaño en bytes. Los dos operadores ternarios siguientes se utilizan para realizar ambas comprobaciones. La condición del:

- Primer operador ternario comprueba si el código de error es 1.
- Segundo, comprueba si el tamaño es superior a 5mb.

```
$error = ($_FILES['image']['error'] === 1) ? 'Too large' : '';  
$error = ($_FILES['image']['size'] >= 5242880) ? '' : 'Too large';
```

Validando el tamaño y tipo de un fichero

Validar el **tipo de medio** y la **extensión** de un archivo ayuda a garantizar que un sitio pueda manejarlo con seguridad. En el siguiente ejemplo:

1. `$allowed_types` es un array de tipos de medios permitidos.
2. La función `mime_content_type()` de PHP intenta detectar el tipo de medio del archivo y lo almacena en `$type`.
3. iii. La función `in_array()` de PHP comprueba si el tipo de medio de este archivo está en el array de tipos de medios permitidos.

Validando el tamaño y tipo de un fichero

4. `$allowed_exts` almacena un array de extensiones permitidas.
5. El nombre del archivo se convierte a minúsculas y se almacena en `$filename`.
6. La extensión del archivo se recoge y se almacena en `$ext`.
7. La función `in_array()` de PHP comprueba si esta extensión de archivo está en el array de extensiones permitidas.

```
① $allowed_types = ['image/jpeg', 'image/png', 'image/gif'];  
② $type = mime_content_type($_FILES['image']['tmp_name']);  
③ $error = in_array($type, $allowed_types) ? '' : 'Wrong file type';  
④ $allowed_exts = ['jpeg', 'jpg', 'png', 'gif'];  
⑤ $filename = strtolower($_FILES['image']['name']);  
⑥ $ext = pathinfo($filename, PATHINFO_EXTENSION);  
⑦ $error .= in_array($ext, $allowed_exts) ? '' : 'Wrong extension';
```

Ejemplo: Validando subidas de ficheros

Este ejemplo reúne el código para cargar, validar y guardar un archivo.

1. Se crean seis variables para contener el:

- Resultado de si se ha subido o no el archivo
- Mensaje de éxito / fracaso que ve el usuario
- Errores si hay problemas con la imagen
- Ruta a la carpeta que almacena los archivos subidos
- Tamaño máximo del archivo en bytes
- Tipos de medios permitidos
- Extensiones de archivo permitidas

Ejemplo: Validando subidas de ficheros

2. Se define una función llamada `create_filename()` .

Utiliza el código que hemos estudiado en esta unidad previamente para limpiar el nombre del fichero y asegurarse de que es único, y devuelve el nuevo nombre. Sus dos parámetros son:

- Nombre del fichero
- Ruta relativa a la carpeta donde se almacenará

3. Una sentencia if comprueba si el formulario ha sido enviado.

4. Se utiliza un operador ternario para comprobar si hubo un error al subir esta imagen porque era mayor que el límite de tamaño establecido en php.ini o .htaccess. Si es así, se almacena un mensaje de error en `$error` .

Ejemplo: Validando subidas de ficheros

5. Otra sentencia if comprueba si el archivo se ha subido sin errores.
6. Se valida el tamaño del archivo. Si es menor o igual que el tamaño máximo almacenado en `$max_size` en el Paso 1, `$error` almacena una cadena en blanco; si es mayor que el tamaño máximo permitido, `$error` contiene el mensaje ' too big ' (demasiado grande).
7. La función incorporada de PHP `mime_content_type()` obtiene el tipo de medio del archivo y lo almacena en `$type`.
8. La función `in_array()` de PHP comprueba si el tipo de medio almacenado en `$type` está en el array `$allowed_types`. Si lo está, se añade una cadena en blanco a la variable `$error`. Si no lo está, se añade un mensaje de error a `$error`.

Ejemplo: Validando subidas de ficheros

9. La función `pathinfo()` de PHP obtiene la extensión del archivo de la imagen cargada. Esta función es llamada dentro de la función `strtolower()` de PHP para asegurar que la extensión está en minúsculas. Luego se almacena en `$ext`.
10. La función `in_array()` de PHP se utiliza para comprobar si la extensión del archivo está permitida. Si lo está, se añade una cadena en blanco a la variable `$error`. Si no lo es, se añade un mensaje para indicar que se trata de una extensión incorrecta.
11. La condición de una sentencia `if` comprueba si `$error` contiene un valor que no se trate como verdadero. Una cadena en blanco se trata como falso (no hay errores).

Ejemplo: Validando subidas de ficheros

12. Si no hay errores, se llama a la función `create_filename()` (del Paso 2) para asegurar que el nombre del archivo es limpio y único.
13. `$destination` contiene la ruta para guardar el nuevo archivo.
14. Se llama a la función de PHP `move_uploaded_file()` para mover el archivo desde su ubicación temporal a la carpeta uploads. Devuelve true si funciona; false si no. El resultado se almacena en una variable llamada `$moved`.

Ejemplo: Validando subidas de ficheros

- 15. Si la variable `$moved` tiene un valor de true, la imagen se ha subido, ha pasado las comprobaciones y se ha guardado, por lo que la variable `$message` almacena una etiqueta HTML `` que mostrará la imagen.
- 16. En caso contrario, se almacena un mensaje de error en `$message`.
- 17. El valor almacenado en la variable `$message` se muestra antes del formulario de subida.

Ejemplo: Validando subidas de ficheros

```
<?php
$moved      = false;           // Initialize
$message    = '';             // Initialize
$error      = '';             // Initialize
① $upload_path = 'uploads/';   // Upload path
$max_size   = 5242880;         // Max file size (in bytes)
$allowed_types = ['image/jpeg', 'image/png', 'image/gif',]; // Allowed file types
$allowed_exts = ['jpeg', 'jpg', 'png', 'gif',]; // Allowed file extensions

function create_filename($filename, $upload_path) // Function to make filename
{
    $basename = pathinfo($filename, PATHINFO_FILENAME); // Get basename
    $extension = pathinfo($filename, PATHINFO_EXTENSION); // Get extension
    $basename = preg_replace('/^[^A-z0-9]/', '-', $basename); // Clean basename
    $i        = 0; // Counter
    ② while (file_exists($upload_path . $filename)) { // If file exists
        $i = $i + 1; // Update counter
        $filename = $basename . $i . '.' . $extension; // New filepath
    }
    return $filename; // Return filename
}
```

Ejemplo: Validando subidas de ficheros

```
③ if ($_SERVER['REQUEST_METHOD'] == 'POST') { // If form submitted
④     $error = ($_FILES['image']['error'] === 1) ? 'too big ' : ''; // Check size error

⑤     if ($_FILES['image']['error'] == 0) { // If no upload errors
⑥         $error .= ($_FILES['image']['size'] <= $max_size) ? '' : 'too big '; // Check size
        // Check the media type is in the $allowed_types array
⑦         $type = mime_content_type($_FILES['image']['tmp_name']);
⑧         $error .= in_array($type, $allowed_types) ? '' : 'wrong type ';
        // Check the file extension is in the $allowed_exts array
⑨         $ext = strtolower(pathinfo($_FILES['image']['name'], PATHINFO_EXTENSION));
⑩         $error .= in_array($ext, $allowed_exts) ? '' : 'wrong file extension ';
        // If there are no errors, create the new filepath and try to move the file
⑪         if (!$error) {
⑫             $filename = create_filename($_FILES['image']['name'], $upload_path);
⑬             $destination = $upload_path . $filename;
⑭             $moved = move_uploaded_file($_FILES['image']['tmp_name'], $destination);
        }
    }
}
```


Ejemplo: Validando subidas de ficheros

```
⑮ { if ($moved === true) { // If it moved
    $message = 'Uploaded:<br>'; // Show image
  } else { // Otherwise
    $message = '<b>Could not upload file:</b> ' . $error; // Show errors
  }
}
⑰ ?> ... <?= $message ?> <!-- Show form -->
```


Actividad: Validando subidas de ficheros

Modifica la actividad anterior (*Actividad: Moviendo un archivo subido*) para implementar las técnicas de:

- Limpieza de nombres de ficheros
- Evitar sobrescribir ficheros ya existentes
- Validación de tipos de medios
- Validación del tamaño de ficheros

Implementa las comprobaciones necesarias para que el fichero que el usuario sube como imagen de perfil no contenga caracteres raros en su nombre, no sobrescriba otra imagen de perfil ya subida previamente, sea un fichero en formato .png o .jpeg y contenga la extensión apropiada.



7. Redimensionar imágenes

Redimensionar imágenes

Cuando los usuarios suben imágenes, los sitios suelen redimensionarlas para que todas tengan un tamaño similar; esto hace que la página parezca más ordenada y se cargue más rápido. Para redimensionar una imagen, necesitas su **proporción**: su anchura dividida por su altura.

Redimensionar imágenes

Las imágenes cargadas se redimensionan a menudo por dos razones:

- Cuando un conjunto de imágenes tienen tamaños similares, se ven mejor que cuando tienen tamaños muy diferentes.
- Cuando un archivo cargado es mayor que el tamaño al que se muestra, se ralentiza la carga de la página.

Redimensionar imágenes

Cuando redimensiones imágenes, **debes mantener la misma proporción** (la anchura dividida por la altura), de lo contrario las imágenes redimensionadas se verán **distorsionadas** (ver ejemplo a continuación).

Si se desea que todas las imágenes tengan exactamente el mismo tamaño, se puede **recortar** la imagen (seleccionar parte de ella) y luego **redimensionar** esa selección, conservando su proporción (vemos cómo hacer esto en los siguientes apartados).

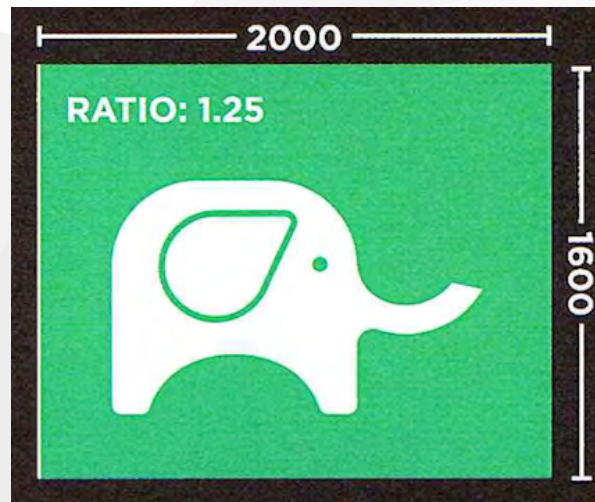
Redimensionar imágenes

Apaisado (Landscape)

En las imágenes apaisadas, la anchura es mayor que la altura, por lo que la proporción es mayor que 1.

En el ejemplo siguiente, si la anchura es 2000 y la altura 1600, la proporción será

$$2000 \div 1600 = 1.25$$



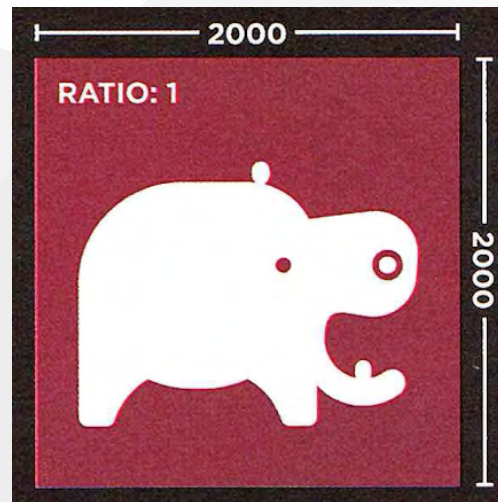
Redimensionar imágenes

Cuadrado

En las imágenes cuadradas, la altura y la anchura son iguales, por lo que la proporción es exactamente 1.

En el ejemplo siguiente, si la anchura es 2000 y la altura 2000, la proporción será

$$2000 \div 2000 = 1$$



Redimensionar imágenes

Retrato (Portrait)

En las imágenes en formato retrato (o vertical), la anchura es siempre menor que la altura, por lo que la proporción es menor que 1.

En el ejemplo siguiente, si la anchura es de 1600 y la altura de 2000, la proporción será:

$$1600 \div 2000 = 0.8$$



Redimensionar imágenes

A continuación puedes ver cómo calcular la nueva anchura y altura que tendrá una imagen al redimensionarla. Al mantener la misma proporción que la imagen original, la imagen redimensionada no se verá distorsionada.

Para que un conjunto de imágenes redimensionadas tenga un aspecto más coherente, se redimensionan para que quepan dentro de una **caja de contención** cuadrada (o **caja delimitadora**) que establece la anchura y altura máximas que puede tener la imagen.

Redimensionar imágenes

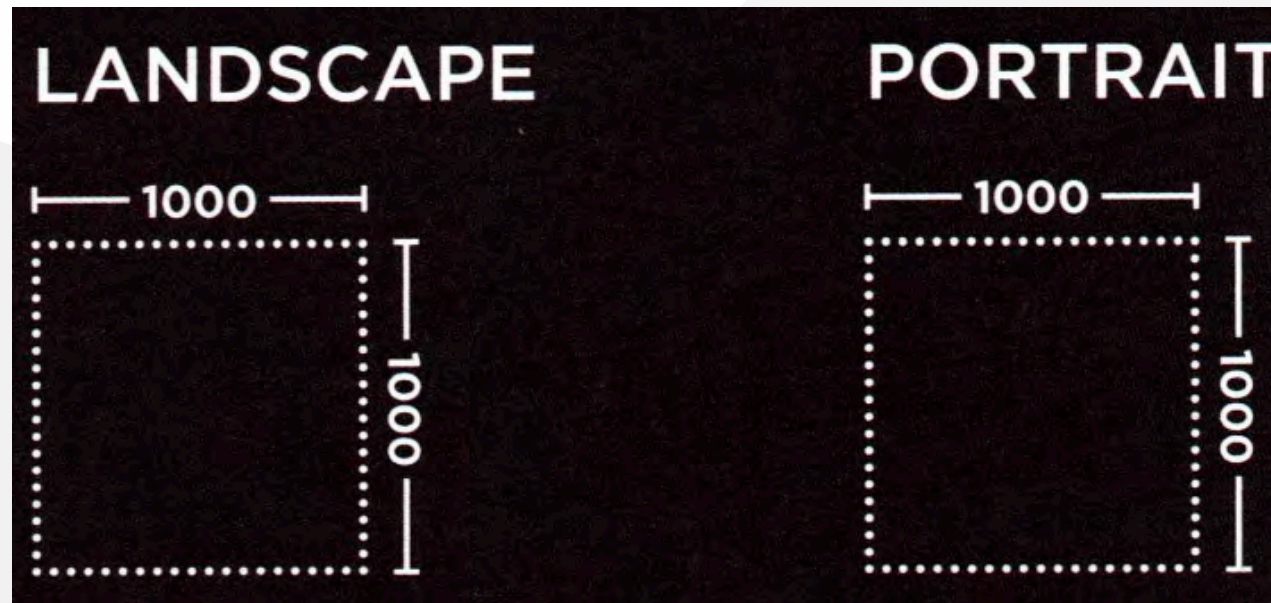
Cuando se cambia el tamaño de la imagen, el lado más largo de la imagen (la anchura o la altura) llenará el contenedor, y el lado más corto se calculará utilizando la proporción de la imagen.

Redimensionar imágenes

Hay que definir la anchura y la altura del contenedor.

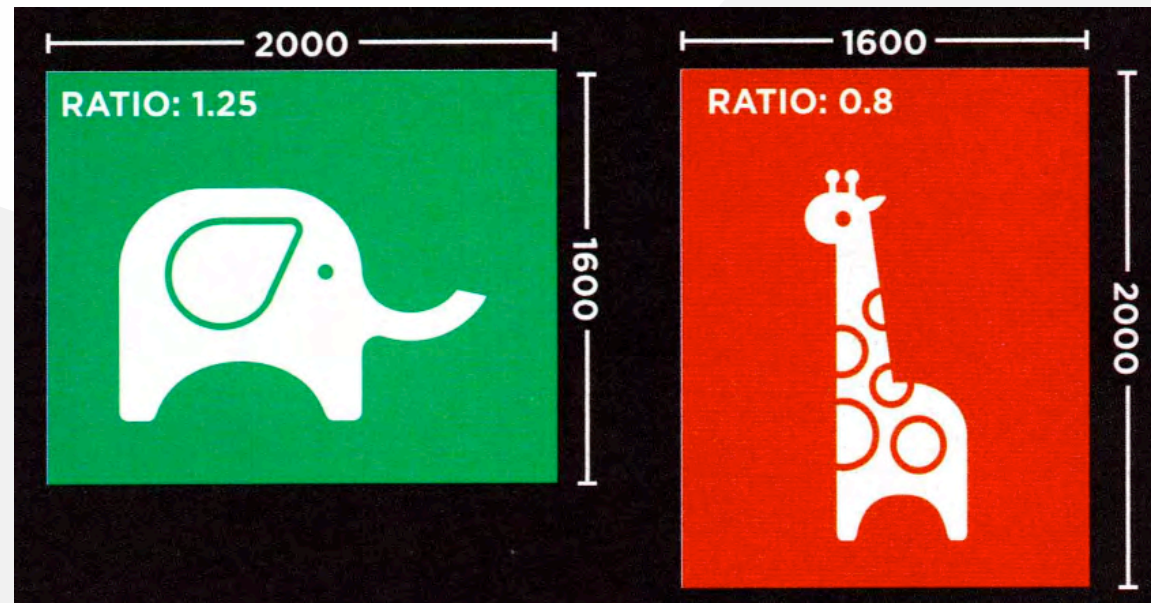
Se trata de la anchura y altura máximas que puede tener la imagen redimensionada.

En este ejemplo, la anchura y la altura máximas se fijan en 1000.



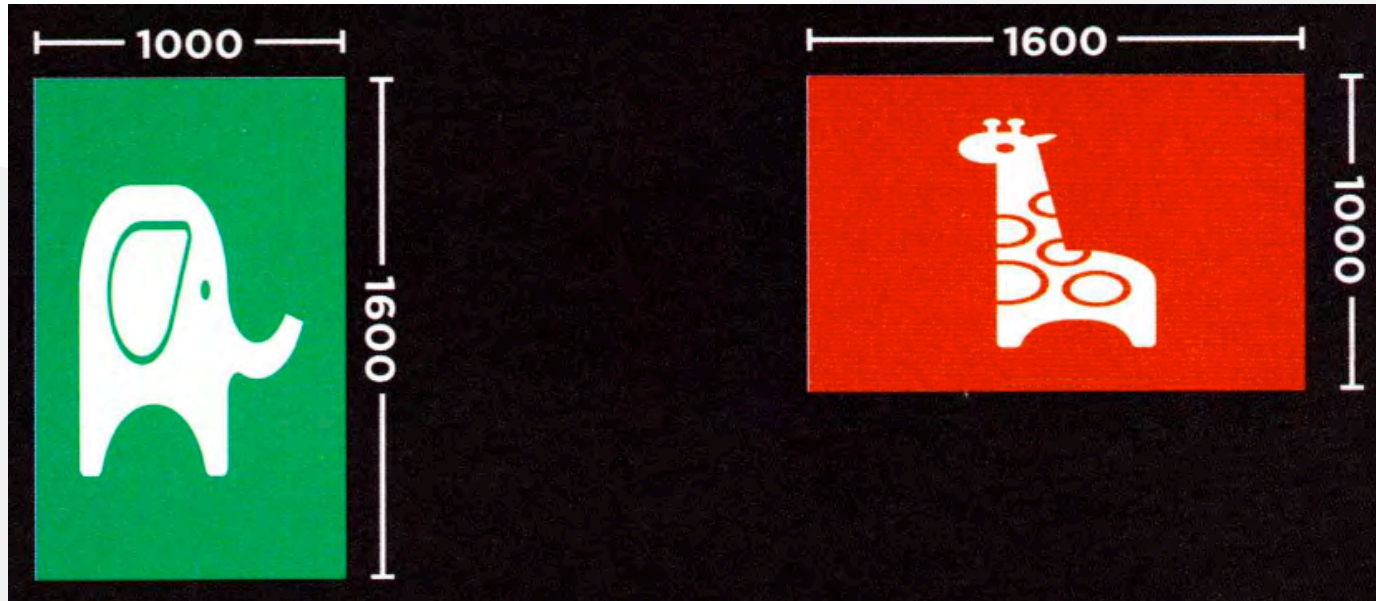
Redimensionar imágenes

1. Obtener la anchura y la altura de la imagen original cargada.
Utilízalos para calcular la proporción de la imagen ($\text{anchura} \div \text{altura}$).



Redimensionar imágenes

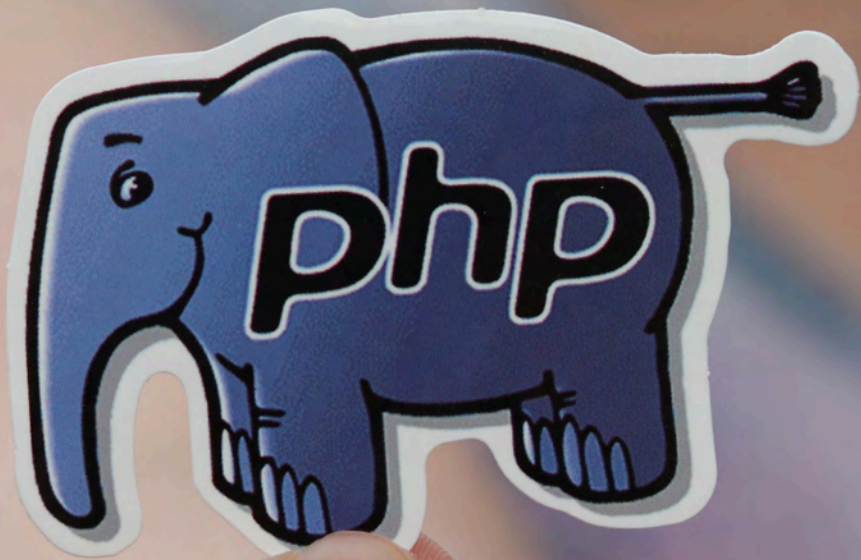
2. Si la anchura es mayor que la altura, la imagen es apaisada. En caso contrario, será vertical. Ajusta el lado más largo de la imagen al tamaño del contenedor.



Redimensionar imágenes

3. Calcula la longitud del lado más corto de la imagen redimensionada.
- **Landscape:** Divide la altura del contenedor por la proporción.
 - **Portrait** (Retrato): Multiplica la anchura del contenedor por la proporción.





8. Recortar imágenes

Recortar imágenes

Recortar imágenes permite crear un **conjunto de imágenes que tienen todas exactamente el mismo tamaño**, y permite que las nuevas imágenes llenen la caja que las contiene.

Cuando se recortan imágenes, **se elimina parte de la imagen original**.

Recortar imágenes

Para recortar una imagen, debes seleccionar la parte de la imagen original que desees conservar.

Para que un conjunto de imágenes tenga una forma coherente, la sección recortada de cada imagen debe tener la misma proporción.

Una vez seleccionada el área que desea recortar, puedes cambiar su tamaño para asegurarse de que todas las imágenes tengan el mismo tamaño.

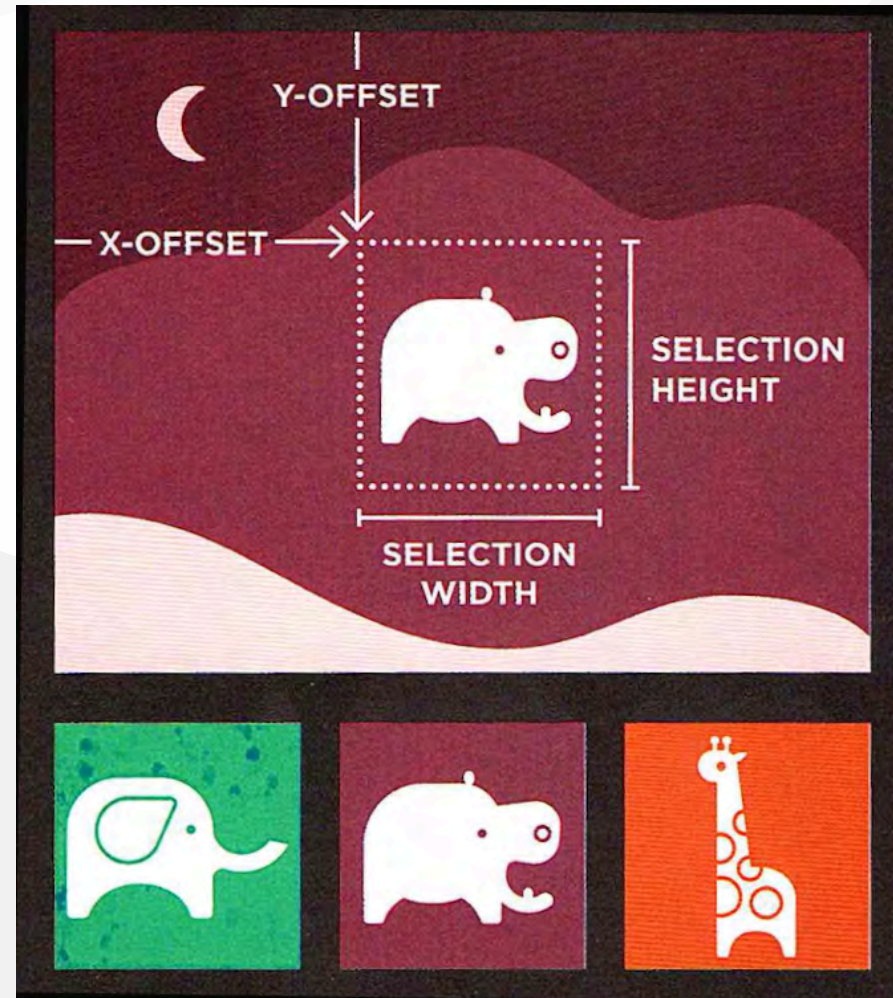
Recortar imágenes

Para seleccionar el área de la imagen que desea recortar, necesita cuatro datos:

- **Anchura** de la selección: La anchura del área de la imagen que desea conservar a partir del desplazamiento x.
- **Altura** de selección: La altura del área de la imagen que desea mantener desde el desplazamiento y.
- **Desplazamiento X**: La distancia desde el lado izquierdo de la imagen hasta el inicio de la selección.
- **Desplazamiento Y**: La distancia desde la parte superior de la imagen hasta el inicio de la selección.

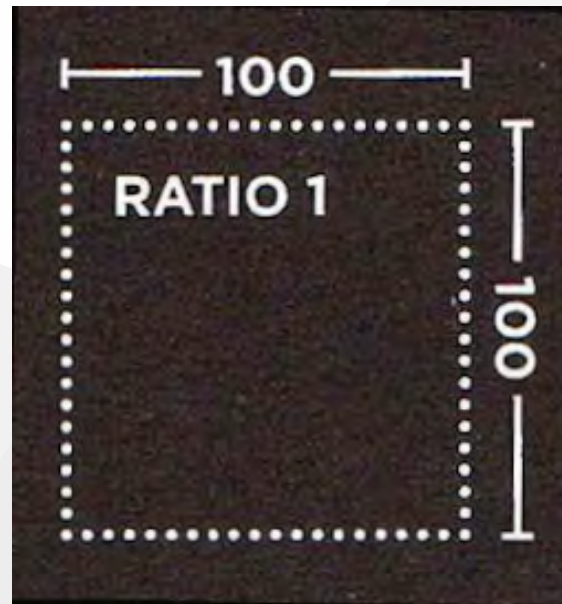
Para garantizar que todas las imágenes cargadas tengan el mismo tamaño, es necesario especificar la anchura y la altura que queremos que tengan. Esos valores se utilizarán para calcular la proporción de la nueva imagen ($\text{anchura} \div \text{altura}$).

Recortar imágenes



Recortar imágenes

Para garantizar que todas las imágenes cargadas tengan el mismo tamaño, es necesario especificar la anchura y la altura que queremos que tengan. Esos valores se utilizarán para calcular la proporción de la nueva imagen ($\text{anchura} \div \text{altura}$).



Recortar imágenes

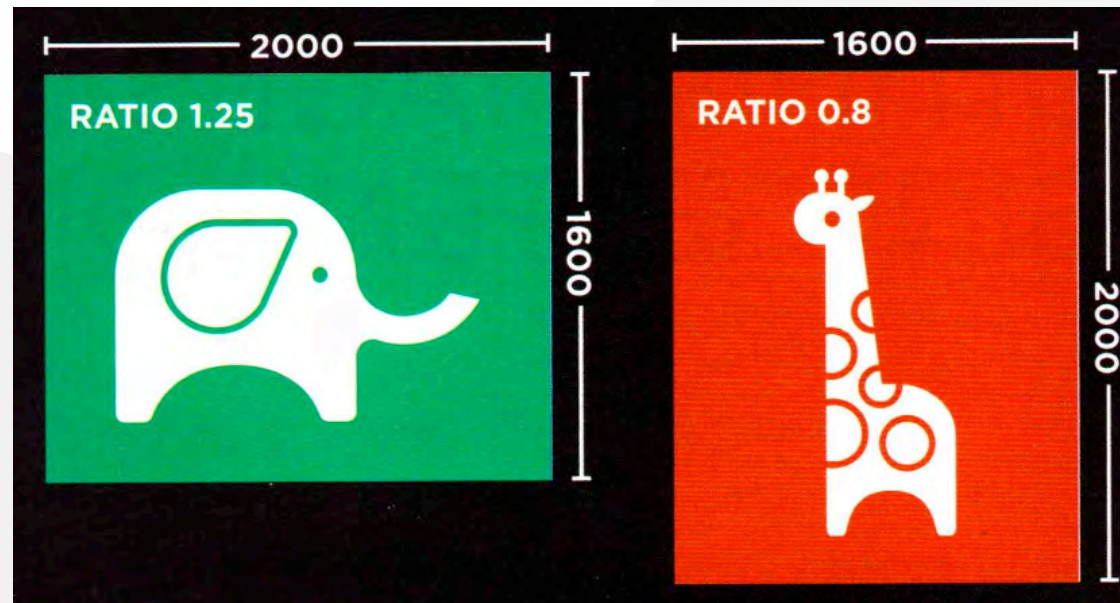
Existen herramientas JavaScript que permiten a los usuarios **recortar imágenes en el navegador antes de cargar la imagen**. Algunas de las opciones disponibles son:

- [Croppie](#)
- [Cropper.js](#)
- [Croppr.js](#)
- [Vanilla Image Cropper](#)
- [JS-Cropper](#)

Recortar imágenes

Los pasos que tendremos que realizar para recortar una imagen son:

1. Obtener la anchura y la altura de la imagen cargada y calcular la proporción de la imagen cargada (anchura / altura).



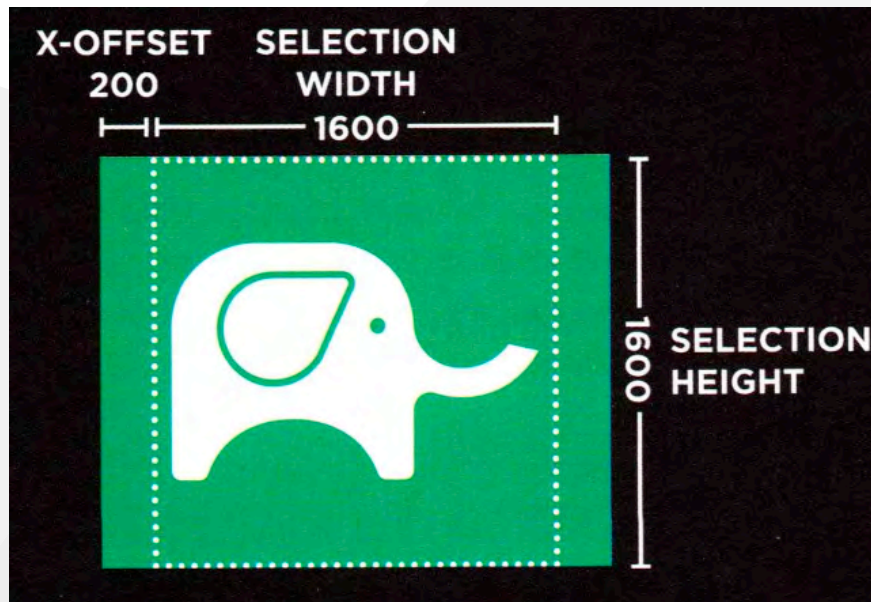
Recortar imágenes

2. Selecciona la parte de la imagen que se desee conservar. Esta selección debe tener la misma proporción que la nueva imagen. A continuación se muestran los cálculos para las selecciones y el desplazamiento.

Recortar imágenes

Si la nueva relación de imagen es menor que la relación de imagen cargada:

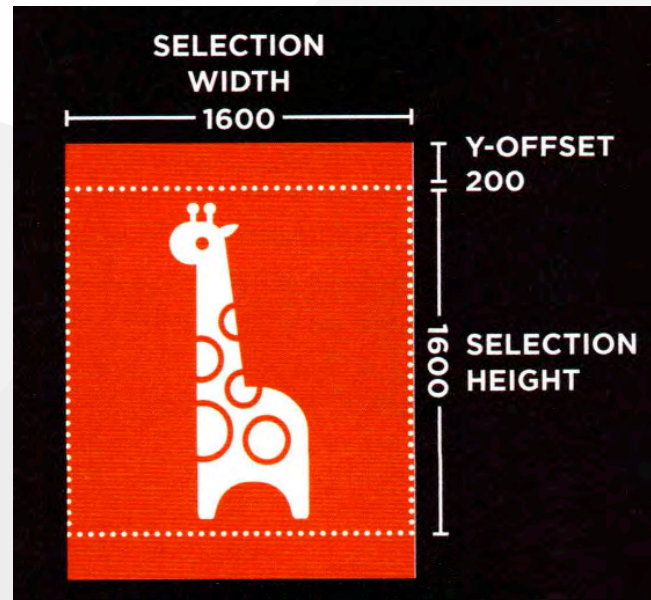
- **Anchura** de la selección = altura original x nueva proporción.
- **Altura** de selección = altura original.
- **Desplazamiento X** = (anchura original - anchura de la selección) / 2.
- **Desplazamiento Y** = 0.



Recortar imágenes

En caso contrario, utiliza los siguientes cálculos:

- **Anchura de selección** = anchura original
- **Altura de selección** = anchura original x nueva proporción
- **Desplazamiento X** = 0
- **Desplazamiento Y** = (altura original - altura de la selección) / 2



Recortar imágenes

3. El área recortada se redimensiona para que tenga el tamaño de la nueva imagen.





9. Editando imágenes mediante extensiones

Editando imágenes mediante extensiones

Las extensiones añaden funcionalidad al intérprete PHP, permitiéndole realizar tareas adicionales. **GD** e **Imagick** son dos extensiones populares que permiten al intérprete PHP **redimensionar y recortar imágenes**.

Editando imágenes mediante extensiones

Cuando se instalan extensiones en un servidor web, **suelen proporcionarle funciones o clases adicionales** que nuestras páginas PHP pueden utilizar (del mismo modo que nuestro código utiliza las funciones y clases incorporadas de PHP).

Las extensiones GD e Imagick realizan tareas similares a las funciones básicas de Photoshop (o cualquier otro software de edición de imágenes) pero, en lugar de manipular imágenes utilizando una interfaz gráfica de usuario, nos permiten editar la imagen utilizando código PHP.

Editando imágenes mediante extensiones

El resto de esta unidad explica cómo redimensionar y recortar imágenes primero utilizando GD, y luego utilizando Imagick.

GD es más complicado de utilizar que Imagick, pero ha sido instalado por defecto con el intérprete PHP desde PHP 4.3, mientras que Imagick debe ser instalado en el servidor web antes de poder ser utilizado.

Editando imágenes mediante extensiones (GD)

Usando GD

- Si utilizas MAMP en un Mac, GD debería estar habilitado por defecto.
- Si utilizas XAMPP en un PC, probablemente necesites habilitar la extensión GD antes de poder utilizarla (ver *Anexo I - Habilitando GD en XAMPP*).

Editando imágenes mediante extensiones (GD)

Para redimensionar y recortar una imagen utilizando GD, debes usar las funciones de GD que se muestran a continuación.

GD dispone de funciones establecidas para abrir distintos tipos de medios (GIF, JPEG, PNG, WEBP, etc.), y de funciones correspondientes para guardarlos (el tipo de medio, indicado por *mediatype* en la siguiente tabla, se sustituye por el tipo de archivo como veremos a continuación).

FUNCTION	DESCRIPTION
<code>getimagesize()</code>	Get the dimensions and media type of an image
<code>imagecreatefrommediatype()</code>	Open the image (replace <i>mediatype</i> with the image's media type)
<code>imagecreatetruecolor()</code>	Create a new blank image using the dimensions of the resized or cropped image
<code>imagecopyresampled()</code>	Take the selected part of the original image, resize it and paste it into the new image created in the previous step
<code>imagemediatype()</code>	Save the image (replace <i>mediatype</i> with the image's media type)

Editando imágenes mediante extensiones (GD)

Determinando el tipo de medio

Para seleccionar la función correcta para abrir o guardar una imagen, es necesario conocer el tipo de medio de la imagen.

La función `getimagesize()` de GD requiere una ruta a una imagen como argumento, y devuelve un array que contiene datos sobre la imagen, incluido su tipo de medio.

Editando imágenes mediante extensiones (GD)

Determinando el tipo de medio

La tabla mostrada a continuación muestra los datos contenidos en ese array (las claves son una mezcla de números y palabras).

KEY	DESCRIPTION
0	Width of the image (in pixels)
1	Height of the image (in pixels)
2	Constant describing the image type
3	String with dimensions for use in an tag: height="yyy" width="xxx"
mime	Media type of the image
channels	3 if image is RGB, 4 if it is CMYK
bits	Number of bits used for each color

Editando imágenes mediante extensiones (GD)

Abrir y guardar imágenes

El tipo de medio de la imagen puede utilizarse en una sentencia *switch* (como se muestra en el siguiente ejemplo) para que se llame a la función adecuada para abrir o guardar la imagen.

La tabla mostrada a continuación muestra algunas de las funciones que ofrece GD para abrir y guardar formatos de imagen.

FORMAT	OPEN	SAVE
GIF	<code>imagecreatefromgif()</code>	<code>imagegif()</code>
JPEG	<code>imagecreatefromjpeg()</code>	<code>imagejpeg()</code>
PNG	<code>imagecreatefrompng()</code>	<code>imagepng()</code>
WEBP	<code>imagecreatefromwebp()</code>	<code>imagewebp()</code>

Editando imágenes mediante extensiones (GD)

Redimensionar y recortar imágenes

La función `imagecopyresampled()` copia parte de una imagen (o toda) en una nueva imagen en blanco.

Para ello, la función tiene 10 parámetros, pero puede ser más fácil pensar en ellos como 5 pares de parámetros:

```
imagecopyresampled($new, $orig, $new_x, $new_y, $orig_x, $orig_y,  
                   $new_width, $new_height, $orig_width, $orig_height);
```

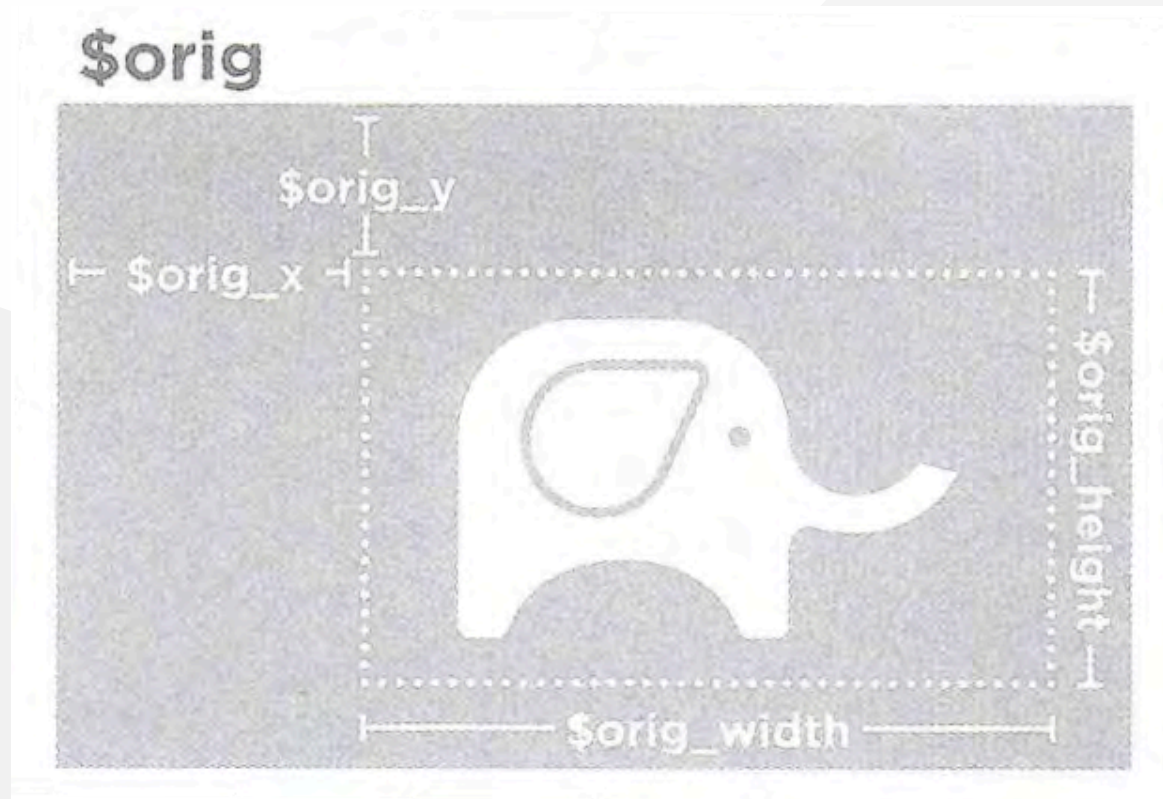

Editando imágenes mediante extensiones (GD)

Redimensionar y recortar imágenes

- `$new` , `$orig` : Imágenes nueva y original (almacenadas en variables antes de llamar a la función, lo vemos a continuación).
- `$new_x` , `$new_y` : X-offset y Y-offset donde debe posicionar el área copiada en la nueva imagen.
- `$orig_x` , `$orig_y` : X-offset y Y-offset donde debe tomar la selección de la imagen original.
- `$new_width` , `$new_height` : Anchura y altura de la selección en la nueva imagen.
- `$orig_width` , `$orig_height` : Ancho y alto de la selección en la imagen original.

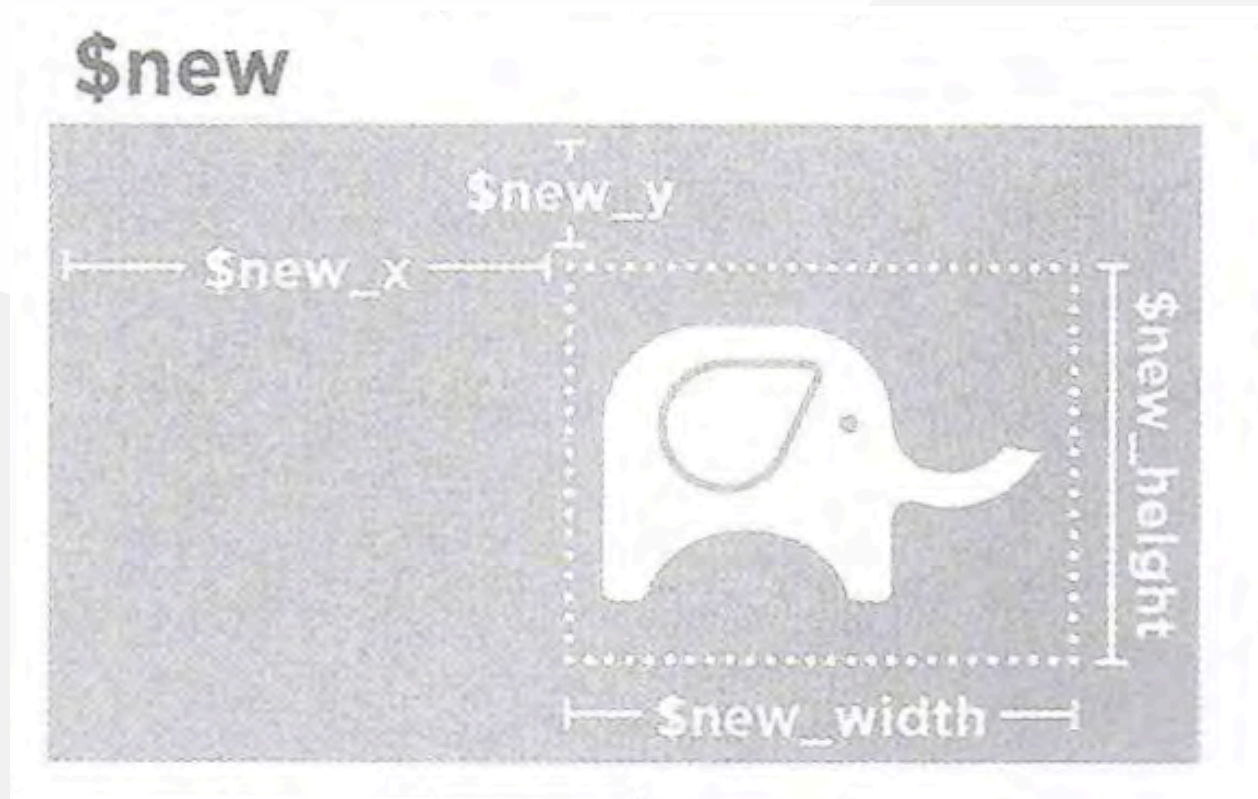
Editando imágenes mediante extensiones (GD)

Redimensionar y recortar imágenes



Editando imágenes mediante extensiones (GD)

Redimensionar y recortar imágenes



Ejemplo: Redimensionando imágenes usando GD

La función que se muestra en este ejemplo **utiliza GD para crear una miniatura**. La miniatura mantendrá la misma relación de aspecto que el original. El nuevo tamaño se basa en una anchura y altura máximas dadas como argumentos.

Este ejemplo parte del ejemplo anterior (*Ejemplo: Validando subidas de ficheros*). Las únicas diferencias son que crea una ruta para la imagen en miniatura redimensionada, y luego llama a esta función para crear una miniatura una vez que la imagen cargada ha sido movida (Pasos 14-15).

Ejemplo: Redimensionando imágenes usando GD

1. `resize_image_gd()` tiene cuatro parámetros:
 - Ruta a la imagen subida
 - Ruta para guardar la imagen redimensionada
 - Ancho máximo de la nueva imagen
 - Altura máxima de la nueva imagen
2. La función `getimagesize()` de GD devuelve un array que contiene datos sobre la imagen, incluyendo sus dimensiones y tipo de medio.
3. El ancho, alto y tipo de medio de la imagen se toman del array y se almacenan en variables.
4. Las variables `$new_width` y `$new_height` se inicializan con la anchura y altura máximas que puede tener la miniatura.

Ejemplo: Redimensionando imágenes usando GD

5. `$orig_ratio` almacena el ratio de la imagen subida.
6. Si la anchura es mayor que la altura de la imagen, entonces la imagen es apaisada.
7. Para una imagen apaisada, la anchura de la imagen redimensionada será la anchura máxima. Este valor se estableció cuando se inicializó la variable en el paso 4. Pero hay que calcular la nueva altura de la imagen; para ello, divide la anchura de la imagen por su proporción.
8. En caso contrario, la imagen es vertical o cuadrada. Su altura sigue siendo la altura máxima fijada en el paso 4. La nueva anchura se calcula multiplicando la nueva altura de la imagen por su ratio.

Ejemplo: Redimensionando imágenes usando GD

9. Se utiliza una sentencia switch para seleccionar la función correcta para abrir la imagen. (como se ha mostrado previamente, GD utiliza funciones separadas para abrir imágenes que son diferentes tipos de medios). El tipo de medio de la imagen (almacenado en `$media_type` en el Paso 3) se utiliza como la condición de la sentencia switch. La imagen abierta se almacena en `$orig`.
10. La función `imagecreatetruecolor()` de GD crea una imagen en blanco, que se almacena en `$new`. Los dos argumentos proporcionados son la anchura y la altura que debe tener la nueva imagen.
11. La función `imagecopyresampled()` de GD copia la imagen original, la redimensiona y la pega en la nueva imagen creada en el paso 10. Necesita que se le den valores para los 10 parámetros que se describieron en la página anterior.

Ejemplo: Redimensionando imágenes usando GD

12. Se utiliza otra sentencia switch para seleccionar la función correcta para guardar la imagen redimensionada. Esta vez, se utiliza una forma abreviada de la sentencia switch (para que el ejemplo quepa en la página). Las funciones que guardan imágenes devuelven verdadero si la imagen se guarda, y falso si no; este valor se almacena en `$result`.
13. Se devuelve el valor almacenado en `$result`.
14. Una vez que la imagen ha sido subida y movida, la ruta donde se guardará la nueva imagen en miniatura se crea uniendo la ruta a la carpeta *uploads*, el texto *thumb*, y el nombre del archivo.
15. Se llama a `resize_image_gd()`.

Ejemplo: Redimensionando imágenes usando GD

```
<?php
① function resize_image_gd($orig_path, $new_path, $max_width, $max_height)
{
  ②   $image_data = getimagesize($orig_path);           // Get image data
  ③   $orig_width = $image_data[0];                     // Image width
  ④   $orig_height = $image_data[1];                   // Image height
  ⑤   $media_type = $image_data['mime'];                // Media type
  ⑥   $new_width = $max_width;                          // Maximum new width
  ⑦   $new_height = $max_height;                       // Maximum new height
  ⑧   $orig_ratio = $orig_width / $orig_height;        // Original image ratio

  // Calculate new size
  ⑨   if ($orig_width > $orig_height) {                // If landscape
  ⑩     $new_height = $new_width / $orig_ratio;        // Set height using ratio
  ⑪   } else {                                          // Otherwise
  ⑫     $new_width = $new_height * $orig_ratio;        // Set width using ratio
  ⑬   }
}
```

Ejemplo: Redimensionando imágenes usando GD

```
switch($media_type) {  
    case 'image/gif' :  
        $orig = imagecreatefromgif($orig_path);  
        break;  
    case 'image/jpeg' :  
        $orig = imagecreatefromjpeg($orig_path);  
        break;  
    case 'image/png' :  
        $orig = imagecreatefrompng($orig_path);  
        break;  
}
```

⑨

```
// Check the media type  
// If it is a GIF  
// This function opens image  
// End switch statement  
// If it is a JPG  
// This function opens image  
// End switch statement  
// If it is a PNG  
// This function opens image  
// End switch statement
```


Ejemplo: Redimensionando imágenes usando GD

```
⑩ $new = imagecreatetruecolor($new_width, $new_height); // Create a blank image

⑪ [ imagecopyresampled($new, $orig, 0, 0, 0, 0, $new_width, $new_height,
    $orig_width, $orig_height); // Copy orig to new image

// Save image - The thumbs folder must have been created + have the right permissions
switch($media_type) {
    case 'image/gif' : $result = imagegif($new, $new_path); break;
    case 'image/jpeg': $result = imagejpeg($new, $new_path); break;
    case 'image/png' : $result = imagepng($new, $new_path); break;
}

⑫ [
⑬ return $result;
} ... // Code to upload and validate the image is the same as on p296-7
$moved = move_uploaded_file($_FILES['image']['tmp_name'], $destination); // Move file

⑭ $thumbpath = $upload_path . 'thumb_' . $filename; // Create thumbnail path
⑮ $resized = resize_image_gd($destination, $thumbpath, 200, 200); // Create thumbnail
```


Actividad: Redimensionando imágenes usando GD

La actividad se basa en un caso real en el que un desarrollador web necesita implementar un sistema de carga y redimensión de imágenes para un catálogo de productos de una tienda online.

En este caso, la tienda online necesita que cada vez que se suba una imagen de producto, se cree una versión en miniatura (thumbnail) para su visualización en ciertas páginas de la tienda. Las imágenes originales deben conservarse, pero las miniaturas deben generarse automáticamente y almacenarse en una carpeta especial para imágenes redimensionadas.

Actividad: Redimensionando imágenes usando GD

Requerimientos de la actividad:

- Los usuarios pueden subir imágenes de productos con formato .jpeg, .jpg, .png, o .gif.
- El tamaño máximo de las imágenes subidas es de 5MB.
- Se debe crear una versión miniatura de la imagen con dimensiones máximas de 200x200 píxeles.
- Las imágenes originales se almacenarán en la carpeta `uploads/`, mientras que las miniaturas se guardarán en `uploads/thumbs/`.

Redimensionar y recortar con Imagick

La extensión **Imagick** PHP nos permite controlar un software de edición de imágenes de código abierto llamado *ImageMagick* utilizando código PHP. La extensión Imagick:

- Requiere mucho menos código que GD
- Calcula proporciones y dimensiones para redimensionar imágenes (no necesita calcularlas en su código)
- Utiliza los mismos métodos para todos los formatos de imágenes
- Soporta más formatos de imagen que GD

Redimensionar y recortar con Imagick

Pero requiere que tanto la extensión Imagick como el software ImageMagick estén instalados en el servidor web; no están instalados por defecto. Así que debemos realizar alguna de estas comprobaciones o configuraciones dependiendo del entorno de trabajo:

- Habilitar Imagick para MAMP en un Mac.
- Instalar Imagick + ImageMagick para XAMPP en PC (ver Anexo II - Cómo instalar Imagick e ImageMagick para XAMPP en un PC).
- Comprobar si tu proveedor de hosting lo soporta.

Redimensionar y recortar con Imagick

Para utilizar Imagick, se crea un objeto que representa la imagen utilizando la clase Imagick, y se pasa al constructor la ruta a la imagen que va a representar.

VARIBLE TO STORE OBJECT	CLASS NAME	PATH TO IMAGE
<code>\$image</code>	<code>new Imagick</code>	<code>(\$filepath)</code>

```
$image = new Imagick($filepath);
```

El objeto Imagick que crea tiene un conjunto de métodos que manipulan y guardan la imagen.

METHOD	DESCRIPTION
<code>thumbnailImage()</code>	Resize image
<code>cropThumbnailImage()</code>	Crop and resize image
<code>writeImage()</code>	Save image

Redimensionar y recortar con Imagick

En un PC, la ruta que utiliza Imagick para guardar un archivo debe ser una ruta absoluta (no relativa), y las rutas absolutas son diferentes en un PC en comparación con Mac y Unix.

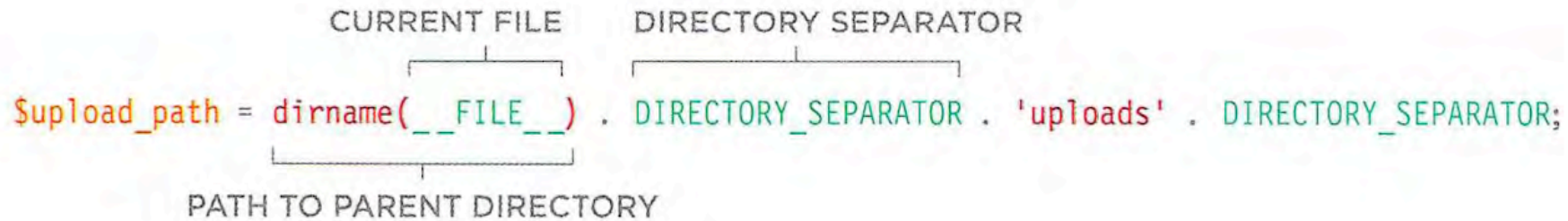
- En un PC, comienzan con una letra de unidad, por ejemplo, `C:\`.
- En Mac y Unix, empiezan por una barra `/`

El separador de directorios también es diferente:

- En un PC es una barra invertida (`\`),
- En Mac y Unix es una barra inclinada (`/`).

Redimensionar y recortar con Imagick

Para crear la ruta correcta al directorio de subida (y almacenarla en una variable), se utiliza el siguiente código.



```
$upload_path = dirname(__FILE__) . DIRECTORY_SEPARATOR . 'uploads' . DIRECTORY_SEPARATOR;
```

The diagram illustrates the components of the PHP code for constructing the upload path. Annotations include:

- CURRENT FILE**: Points to the `__FILE__` constant within the `dirname()` function.
- DIRECTORY SEPARATOR**: Points to the `DIRECTORY_SEPARATOR` constant used twice in the path construction.
- PATH TO PARENT DIRECTORY**: Points to the `dirname()` function, indicating its purpose is to find the parent directory.

La declaración anterior utiliza:

- La función `dirname()` de PHP para devolver la ruta al directorio que contiene el archivo especificado como argumento.
- La constante `__FILE__`, que almacena la ruta al archivo que se está ejecutando actualmente.
- La constante `DIRECTORY_SEPARATOR`, que almacena el separador de directorio correcto para el sistema operativo que se utiliza para ejecutar el archivo PHP.

Ejemplo: Redimensionar y recortar con Imagick

Los dos ejemplos siguientes muestran dos funciones definidas por el usuario que utilizan Imagick para redimensionar y recortar imágenes.

Las sentencias que llaman a estas funciones aparecen justo después del código que mueve el archivo cargado a su destino, igual que en el ejemplo anterior (*Ejemplo: Redimensionando imágenes usando GD*).

El código para subir, validar y mover la imagen es el mismo que el de *Ejemplo: Validando subidas de ficheros*.

Ejemplo: Redimensionar y recortar con Imagick

1. La función `create_thumbnail()` crea una miniatura de una imagen utilizando Imagick. Sus dos parámetros son
 - Ruta a la imagen que se acaba de cargar
 - Ruta de la nueva miniatura que creará Imagick
2. Se crea un nuevo objeto utilizando la clase `Imagick`. Necesita la ruta de la imagen que se ha cargado.
3. El método `thumbnailImage()` del objeto Imagick cambia el tamaño de la imagen. Para ello utiliza tres argumentos:
 - La nueva anchura de la imagen
 - La nueva altura de la imagen
 - Un valor booleano de true para indicar a Imagick que la anchura y la altura son valores máximos y que la miniatura debe tener **la misma proporción que la original**

Ejemplo: Redimensionar y recortar con Imagick

4. El método `writeImage()` de Imagick guarda la imagen en la ubicación almacenada en el parámetro `$destination`.
5. La función devuelve `true` para mostrar que ha funcionado.
6. Una vez que el archivo se ha movido, la variable `$thumbpath` almacena una ruta donde se guardará la nueva miniatura.
7. Se llama a `create_thumbnail()`, y se le da la ruta de la imagen subida y la ruta de la miniatura.

Ejemplo: Redimensionar y recortar con Imagick

```
① function create_thumbnail($temporary, $destination)
{
②     $image = new Imagick($temporary);           // Object to represent image
③     $image->thumbnailImage(200, 200, true);      // Create thumbnail
④     $image->writeImage($destination);           // Save file
⑤     return true;                               // Return true to show success
} ... // Once file has been validated and moved, create the thumbnail path, then thumbnail
$moved      = move_uploaded_file($_FILES['image']['tmp_name'], $destination); // Move
⑥ $thumbpath = $upload_path . 'thumb_' . $filename; // Path to thumbnail
⑦ $thumb     = create_thumbnail($destination, $thumbpath); // Create thumbnail
```

Ejemplo: Redimensionar y recortar con Imagick

8. `create_cropped_thumbnail()` crea un recorte cuadrado de la imagen cargada. Esto asegura que todas las miniaturas tengan el mismo tamaño.
9. La única diferencia con el ejemplo anterior es que utiliza el método `cropThumbnailImage()` del objeto Imagick para crear la miniatura recortada.

```
⑧ function create_cropped_thumbnail($temporary, $destination)
{
    $image = new Imagick($temporary);           // Object to represent image
    ⑨ $image->cropThumbnailImage(200, 200, true); // Create thumbnail
    $image->writeImage($destination);           // Save file
    return true;                                // Return true to show success
}
```


Actividad: Redimensionar y recortar con Imagick

Desarrolla la actividad anterior (*Actividad: Redimensionando imágenes usando GD*) utilizando la extensión Imagick.

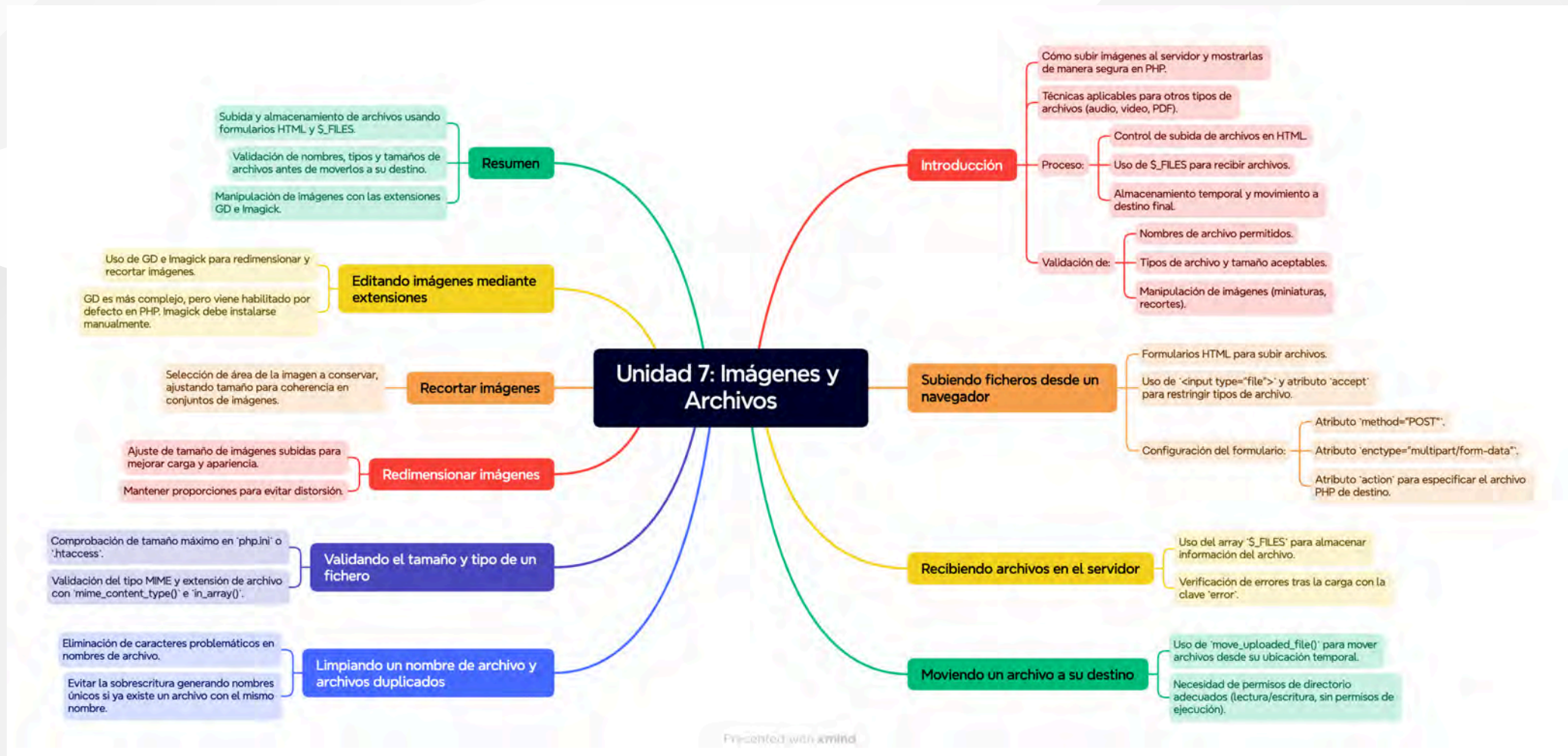
Resumen

- Los formularios HTML utilizan un control de subida de archivos.
- Cuando se sube un fichero, el array superglobal `$_FILES` almacena datos sobre él.
- Cuando se suben archivos, se colocan en una ubicación temporal. Para guardarlos, hay que moverlos a otra carpeta.
- Antes de intentar trabajar con los ficheros, compruebe que se han subido a través de HTTP y que no se han producido errores.

Resumen

- Debemos asegurarnos de que el nombre del archivo sólo utiliza caracteres permitidos.
- Se debe validar el tamaño y el tipo de medio de los archivos cargados antes de guardarlos.
- Cuando se cambie el tamaño de una imagen, es preciso mantener la misma proporción; de lo contrario, se verá estirada y distorsionada.
- GD e Imagick son extensiones que permiten redimensionar y recortar imágenes en el servidor utilizando PHP.

Resumen



Presented with xmind

Referencias

- [Subida de ficheros PHP.net](#)
- [W3Schools - Subida de Archivos en PHP](#)
- [Procesamiento de imágenes y GD](#)
- [Imagick PHP Manual](#)

Anexo I: Habilitar GD en XAMPP

Cuando instalas XAMPP en un PC, se instalará la extensión GD. Pero no está habilitada por defecto.

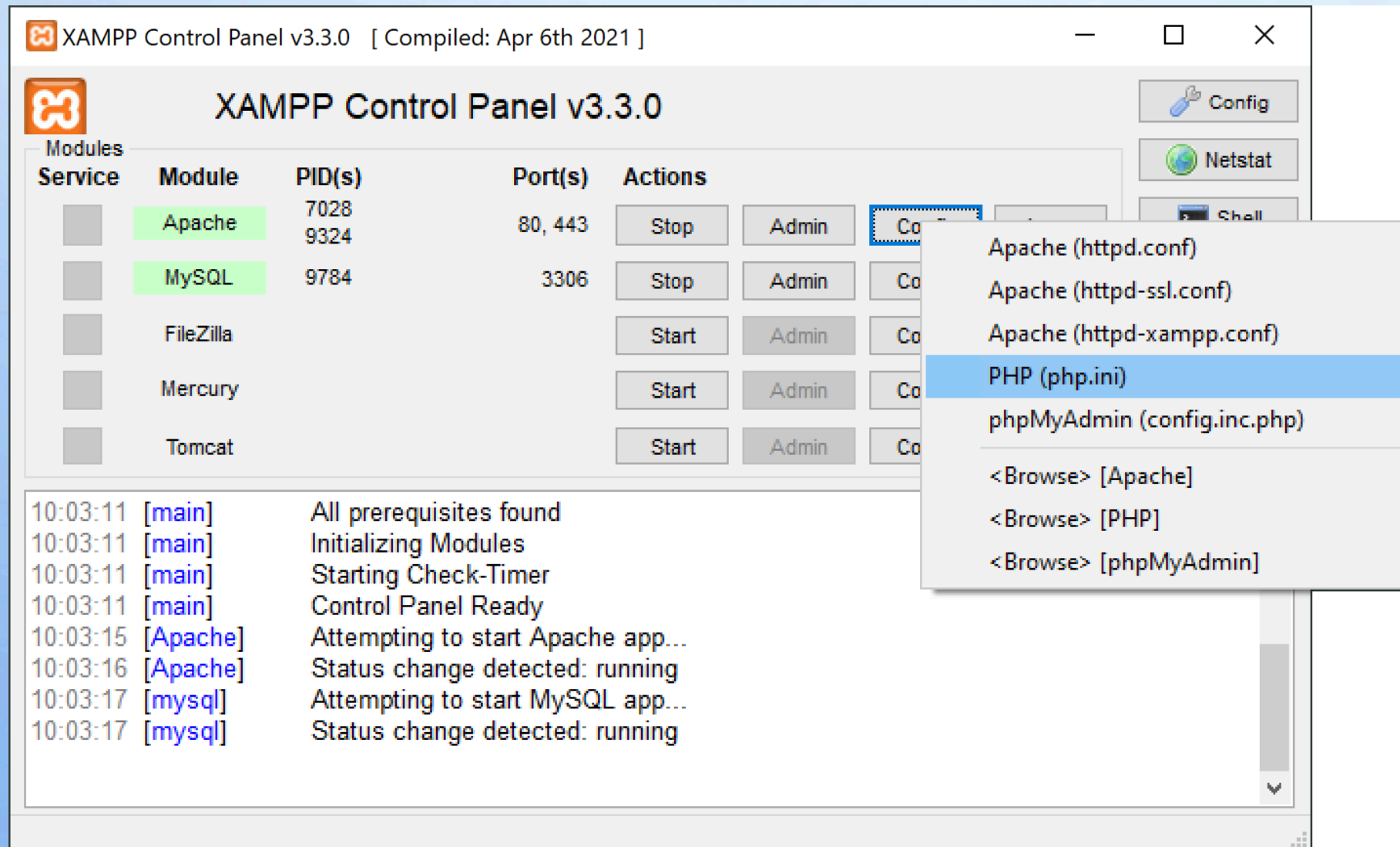
Necesitarás editar el fichero php.ini para que GD pueda ser utilizada, y luego reiniciar XAMPP para que el cambio sea reconocido.

Anexo I: Habilitar GD en XAMPP

Para editar el archivo php.ini, abre el Panel de Control XAMPP, y haz clic en el botón **Config** para el servidor web Apache. A continuación, selecciona el archivo php.ini del menú.

El archivo php.ini debería abrirse en tu editor de texto o código predeterminado.

Anexo I: Habilitar GD en XAMPP

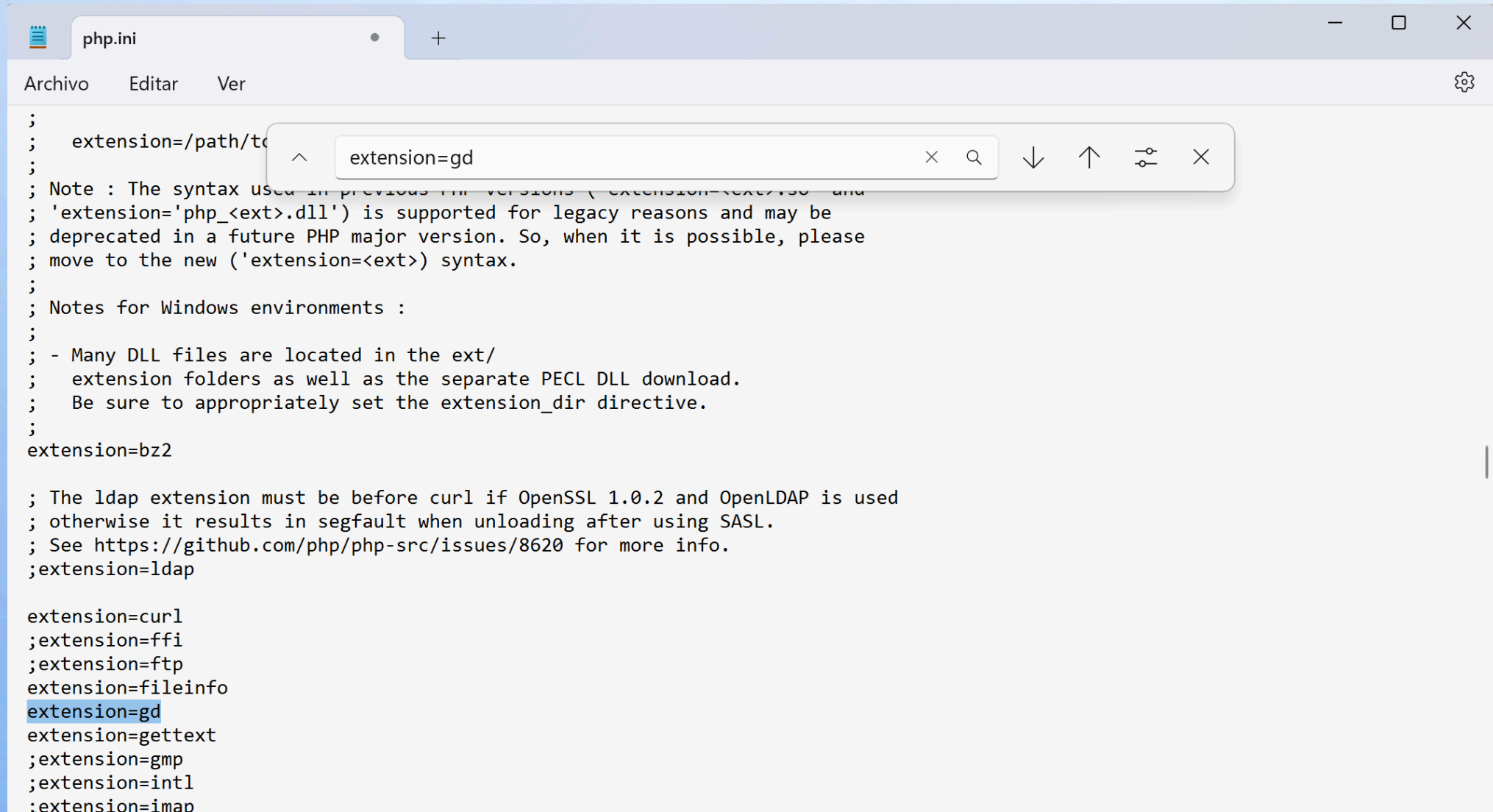


Anexo I: Habilitar GD en XAMPP

En el archivo php.ini, busca el texto `extension=gd` .

Si la línea comienza con un punto y coma, elimínalo (porque el punto y coma indica que el resto de la línea es un comentario). A continuación, guarda el archivo.

Anexo I: Habilitar GD en XAMPP



The image shows a text editor window titled 'php.ini'. The editor has a menu bar with 'Archivo', 'Editar', and 'Ver'. A search bar is open, displaying 'extension=gd'. The text in the editor includes comments about the 'extension' directive syntax and a list of extensions. The 'extension=gd' line is highlighted in blue.

```
;
; extension=/path/to/extension
;
; Note : The syntax used in previous PHP versions ('extension=exts_dir/extension.dll') is supported for legacy reasons and may be
; deprecated in a future PHP major version. So, when it is possible, please
; move to the new ('extension=<ext>') syntax.
;
; Notes for Windows environments :
;
; - Many DLL files are located in the ext/
;   extension folders as well as the separate PECL DLL download.
;   Be sure to appropriately set the extension_dir directive.
;
extension=bz2

; The ldap extension must be before curl if OpenSSL 1.0.2 and OpenLDAP is used
; otherwise it results in segfault when unloading after using SASL.
; See https://github.com/php/php-src/issues/8620 for more info.
;extension=ldap

extension=curl
;extension=ffi
;extension=ftp
extension=fileinfo
extension=gd
extension=gettext
;extension=gmp
;extension=intl
;extension=imap
```

Anexo I: Habilitar GD en XAMPP

Una vez modificado el archivo `php.ini`, deberás reiniciar el servidor web para que los cambios surtan efecto.

Para comprobar que se está ejecutando, puede utilizar la función `phpinfo()` y buscar el encabezado «gd». La primera línea debería indicar que «GD Support» está activado.

Anexo I: Habilitar GD en XAMPP

FTP support	enabled
FTPS support	enabled



GD Support	enabled
GD Version	bundled (2.1.0 compatible)
FreeType Support	enabled
FreeType Linkage	with freetype
FreeType Version	2.11.1
GIF Read Support	enabled
GIF Create Support	enabled
JPEG Support	enabled
libJPEG Version	8
PNG Support	enabled
libPNG Version	1.6.34
WBMP Support	enabled
XPM Support	enabled
libXpm Version	30512
XBM Support	enabled
WebP Support	enabled
BMP Support	enabled
AVIF Support	enabled
TGA Read Support	enabled

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Si quieres que tu código PHP pueda utilizar ImageMagick para editar imágenes en un PC, debes descargar dos programas:

- **ImageMagick**: un software de edición de imágenes
- **Imagick**: una extensión PHP que permite al código PHP controlar el programa ImageMagick

La extensión Imagick añade clases y métodos (que actúan como clases y métodos incorporados) que puedes utilizar para trabajar con imágenes.

Permiten que tu código PHP controle el software ImageMagick, abriendo, editando y guardando imágenes.

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando ImageMagick

1. Obtener la información necesaria



Para instalar ImageMagick, necesitas saber si tu versión de Windows es de 32 o 64 bits.

Si no estás seguro, ve al menú Inicio de Windows y selecciona Configuración > Sistema > Acerca de para abrir la configuración de Acerca de.

Bajo el título Especificaciones del dispositivo, el Tipo de sistema indicará si la versión es de 32 bits o de 64 bits.

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando ImageMagick

 Especificaciones del dispositivo Copiar 

Nombre del dispositivo	2DAW6
Procesador	12th Gen Intel(R) Core(TM) i5-1245U 2.50 GHz
RAM instalada	16,0 GB (15,8 GB usable)
Identificador de dispositivo	4D10184B-7C3A-494A-8A81-F29625445C1E
Id. del producto	00330-66913-82472-AAOEM
Tipo de sistema	Sistema operativo de 64 bits, procesador basado en x64
Lápiz y entrada táctil	Compatibilidad del lápiz y la función táctil con 10 puntos táctiles

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando ImageMagick

2. Descargar el software

Para descargar ImageMagick, dirígete a la página de descargas de <http://imagemagick.org>.

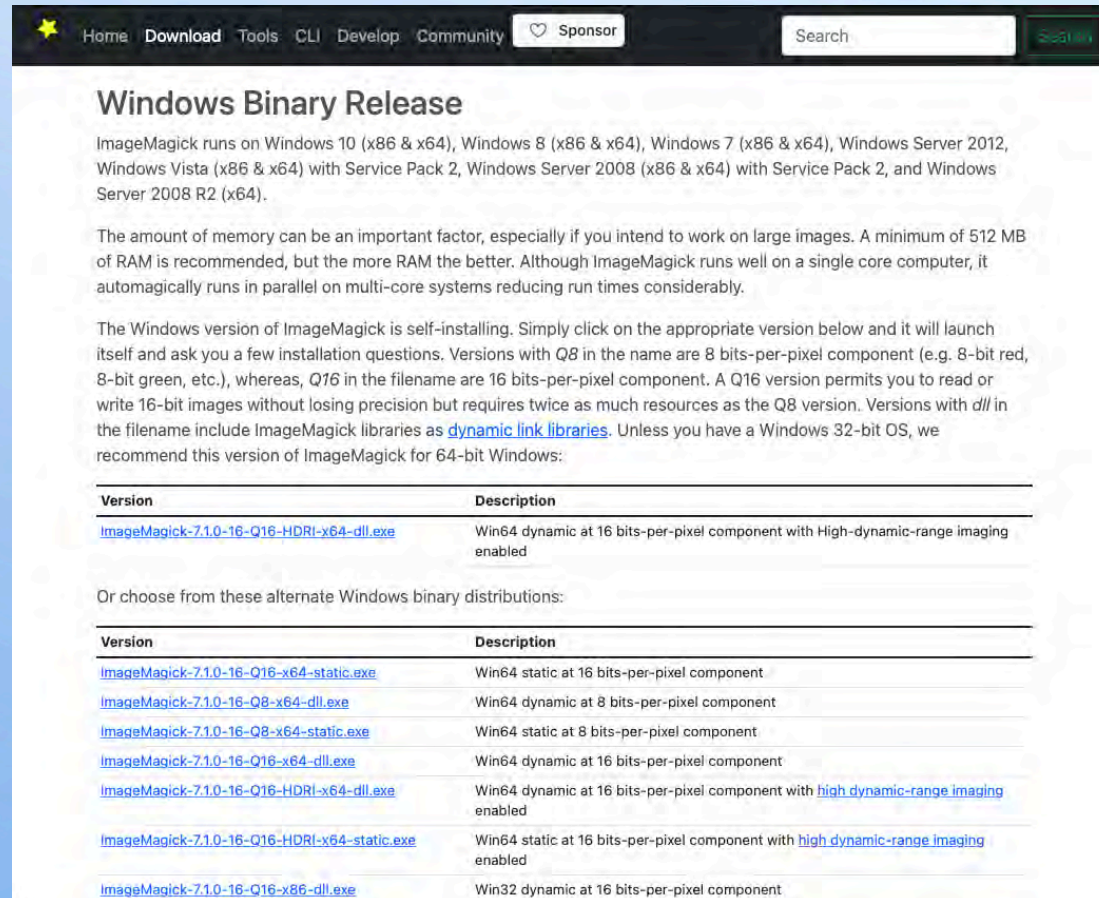
Si tu versión de Windows es

- 64 bits utiliza la primera versión.
- 32-bit utiliza la versión que tiene la descripción «Win32 dynamic at 16-bits-per-pixel component».

Esto descargará un archivo .exe.

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando ImageMagick



The screenshot shows the ImageMagick website's 'Windows Binary Release' page. The page has a dark navigation bar with links: Home, Download, Tools, CLI, Develop, Community, a Sponsor button, and a Search bar. The main content area is white and contains the following text:

Windows Binary Release

ImageMagick runs on Windows 10 (x86 & x64), Windows 8 (x86 & x64), Windows 7 (x86 & x64), Windows Server 2012, Windows Vista (x86 & x64) with Service Pack 2, Windows Server 2008 (x86 & x64) with Service Pack 2, and Windows Server 2008 R2 (x64).

The amount of memory can be an important factor, especially if you intend to work on large images. A minimum of 512 MB of RAM is recommended, but the more RAM the better. Although ImageMagick runs well on a single core computer, it automatically runs in parallel on multi-core systems reducing run times considerably.

The Windows version of ImageMagick is self-installing. Simply click on the appropriate version below and it will launch itself and ask you a few installation questions. Versions with Q8 in the name are 8 bits-per-pixel component (e.g. 8-bit red, 8-bit green, etc.), whereas, Q16 in the filename are 16 bits-per-pixel component. A Q16 version permits you to read or write 16-bit images without losing precision but requires twice as much resources as the Q8 version. Versions with dll in the filename include ImageMagick libraries as [dynamic link libraries](#). Unless you have a Windows 32-bit OS, we recommend this version of ImageMagick for 64-bit Windows:

Version	Description
ImageMagick-7.1.0-16-Q16-HDRI-x64-dll.exe	Win64 dynamic at 16 bits-per-pixel component with High-dynamic-range imaging enabled

Or choose from these alternate Windows binary distributions:

Version	Description
ImageMagick-7.1.0-16-Q16-x64-static.exe	Win64 static at 16 bits-per-pixel component
ImageMagick-7.1.0-16-Q8-x64-dll.exe	Win64 dynamic at 8 bits-per-pixel component
ImageMagick-7.1.0-16-Q8-x64-static.exe	Win64 static at 8 bits-per-pixel component
ImageMagick-7.1.0-16-Q16-x64-dll.exe	Win64 dynamic at 16 bits-per-pixel component
ImageMagick-7.1.0-16-Q16-HDRI-x64-dll.exe	Win64 dynamic at 16 bits-per-pixel component with high dynamic-range imaging enabled
ImageMagick-7.1.0-16-Q16-HDRI-x64-static.exe	Win64 static at 16 bits-per-pixel component with high dynamic-range imaging enabled
ImageMagick-7.1.0-16-Q16-x86-dll.exe	Win32 dynamic at 16 bits-per-pixel component

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando ImageMagick

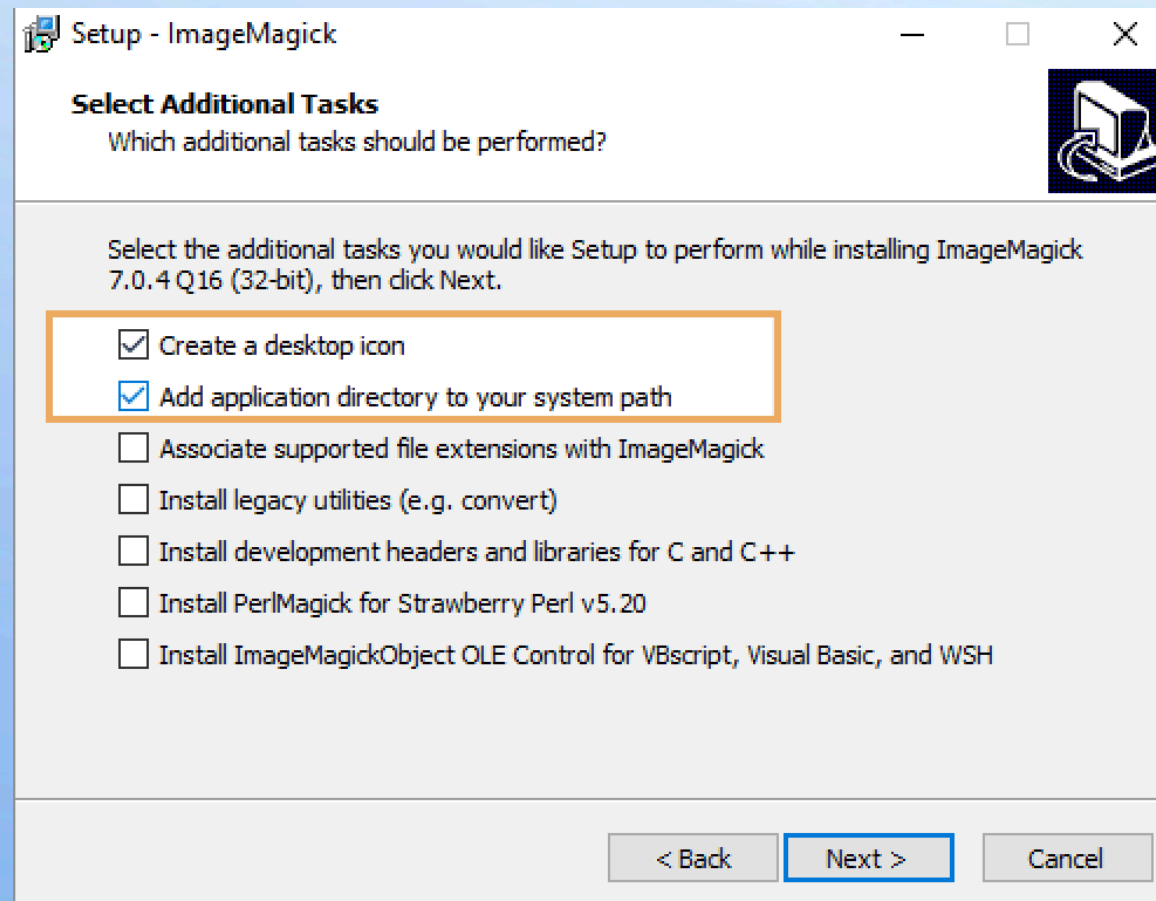
3. Instalar el software

Haz doble clic en el archivo .exe y sigue las instrucciones del instalador.

Durante la instalación, marca la casilla para añadir el directorio de la aplicación a la ruta del sistema.

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando ImageMagick



Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick

1. Obtener la información necesaria

Para instalar imagick, es necesario saber tres cosas, que puede averiguar en la primera tabla de datos que genera la función `phpinfo()` :

- Versión de PHP
- Arquitectura del sistema operativo (x64 o x86)
- Si «Thread Safety» está activado

Instalando Imagick

DWES - U7. Imágenes & Archivos

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick

2. Descargar el software


Haz clic [aquí](#) para descargar la extensión Imagick.

Busca la versión estable (no beta) más reciente.

Haz clic en el enlace DLL situado junto al logotipo de Windows para esa versión.

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick



[Home](#)
[News](#)

Documentation:
[Support](#)




Downloads:
[Browse Packages](#)
[Search Packages](#)
[Download Statistics](#)

Top Level :: Images :: imagick

imagick

Package Information	
Summary	Provides a wrapper to the ImageMagick library.
Maintainers	Dan Ackroyd (lead) [wishlist] [details]
License	PHP License
Description	Imagick is a native php extension to create and modify images using the ImageMagick API. This extension requires ImageMagick version 6.5.3-10+ and PHP 5.4.0+.
Homepage	https://phpimagick.com

[\[Latest Tarball \]](#)[\[Changelog \]](#)[\[View Statistics \]](#)
[\[Browse Source \]](#)[\[Package Bugs \]](#)[\[View Documentation \]](#)

Available Releases				
Version	State	Release Date	Downloads	
3.7.0	stable	2022-01-11	imagick-3.7.0.tgz (351.7kB)  DLL 	[Changelog]
3.7.0RC1	beta	2022-01-04	imagick-3.7.0RC1.tgz (352.2kB)	[Changelog]
3.6.0	stable	2021-11-17	imagick-3.6.0.tgz (343.1kB)  DLL	[Changelog]
3.6.0RC2	beta	2021-11-11	imagick-3.6.0RC2.tgz (343.2kB)	[Changelog]

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick

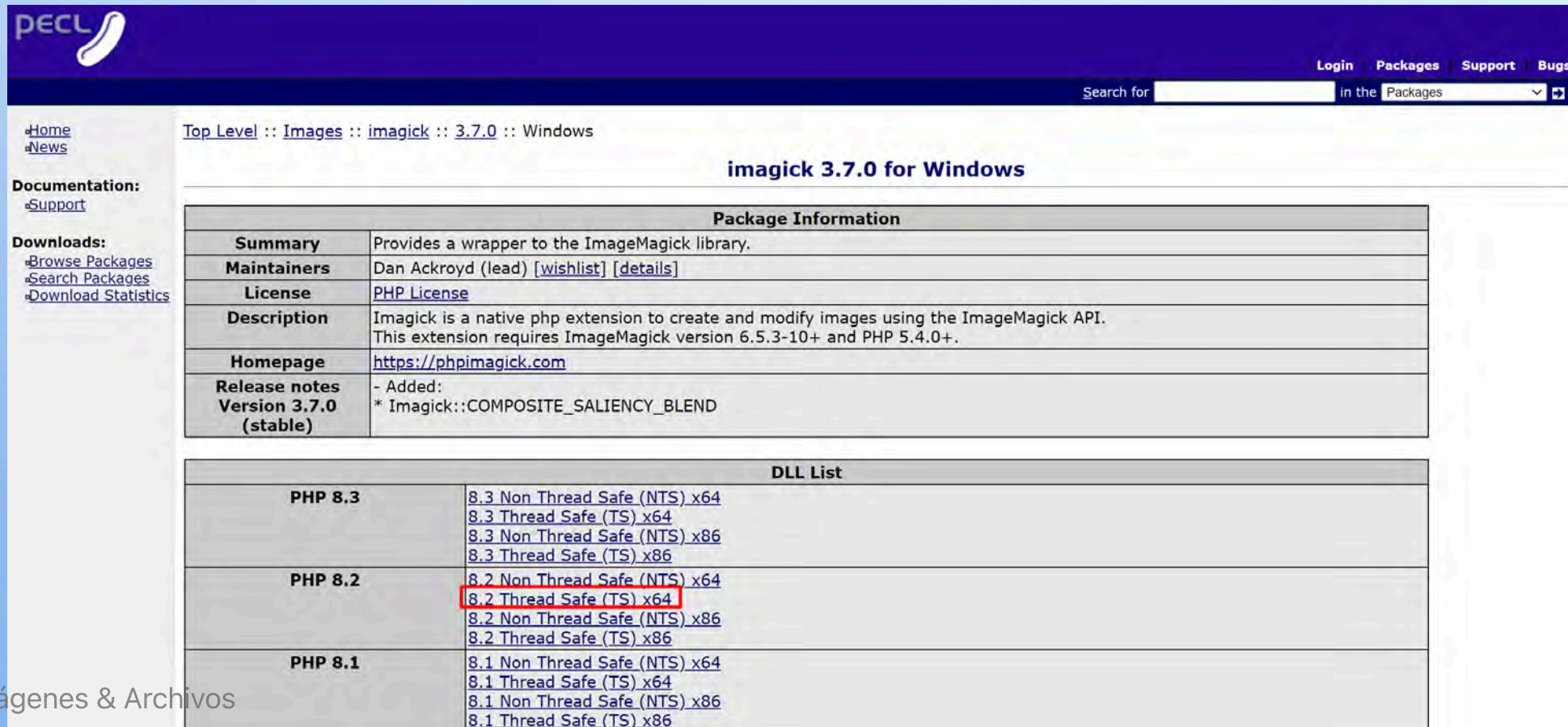
Aparecerá una página en la que se enumeran las distintas versiones de la extensión Imagick que puede descargar para Windows. Selecciona la versión que corresponda:

- Tu versión de PHP
- Si «Thread Safety» está activado
- Arquitectura del sistema operativo (x64 o x86)

Cuando hagas clic en el enlace correspondiente a tu versión, se descargará un archivo ZIP.

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick



PECL

Login Packages Support Bugs

Search for in the Packages

Home News

Documentation: Support

Downloads: Browse Packages Search Packages Download Statistics

Top Level :: Images :: imagick :: 3.7.0 :: Windows

imagick 3.7.0 for Windows

Package Information	
Summary	Provides a wrapper to the ImageMagick library.
Maintainers	Dan Ackroyd (lead) [wishlist] [details]
License	PHP License
Description	Imagick is a native php extension to create and modify images using the ImageMagick API. This extension requires ImageMagick version 6.5.3-10+ and PHP 5.4.0+.
Homepage	https://phpimagick.com
Release notes Version 3.7.0 (stable)	- Added: * Imagick::COMPOSITE_SALIENCY_BLEND

DLL List	
PHP 8.3	8.3 Non Thread Safe (NTS) x64 8.3 Thread Safe (TS) x64 8.3 Non Thread Safe (NTS) x86 8.3 Thread Safe (TS) x86
PHP 8.2	8.2 Non Thread Safe (NTS) x64 8.2 Thread Safe (TS) x64 8.2 Non Thread Safe (NTS) x86 8.2 Thread Safe (TS) x86
PHP 8.1	8.1 Non Thread Safe (NTS) x64 8.1 Thread Safe (TS) x64 8.1 Non Thread Safe (NTS) x86 8.1 Thread Safe (TS) x86

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick

3. Instalar el software

Abre el archivo ZIP de Imagick y encuentra el archivo llamado `php_imagick.dll`.

Busca tu carpeta `php\ext`. La función `phpinfo()` te mostrará la ruta a esta carpeta si buscas `extension_dir`

Típicamente para XAMPP, esta carpeta es `C:\xampp\php\ext`

Copia el archivo `php_imagick.dll` en tu carpeta `php\ext`.

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick

Encuentra tu carpeta apache\bin.

Cuando XAMPP instala Apache, suele estar aquí: `C:\xampp\apache\bin`

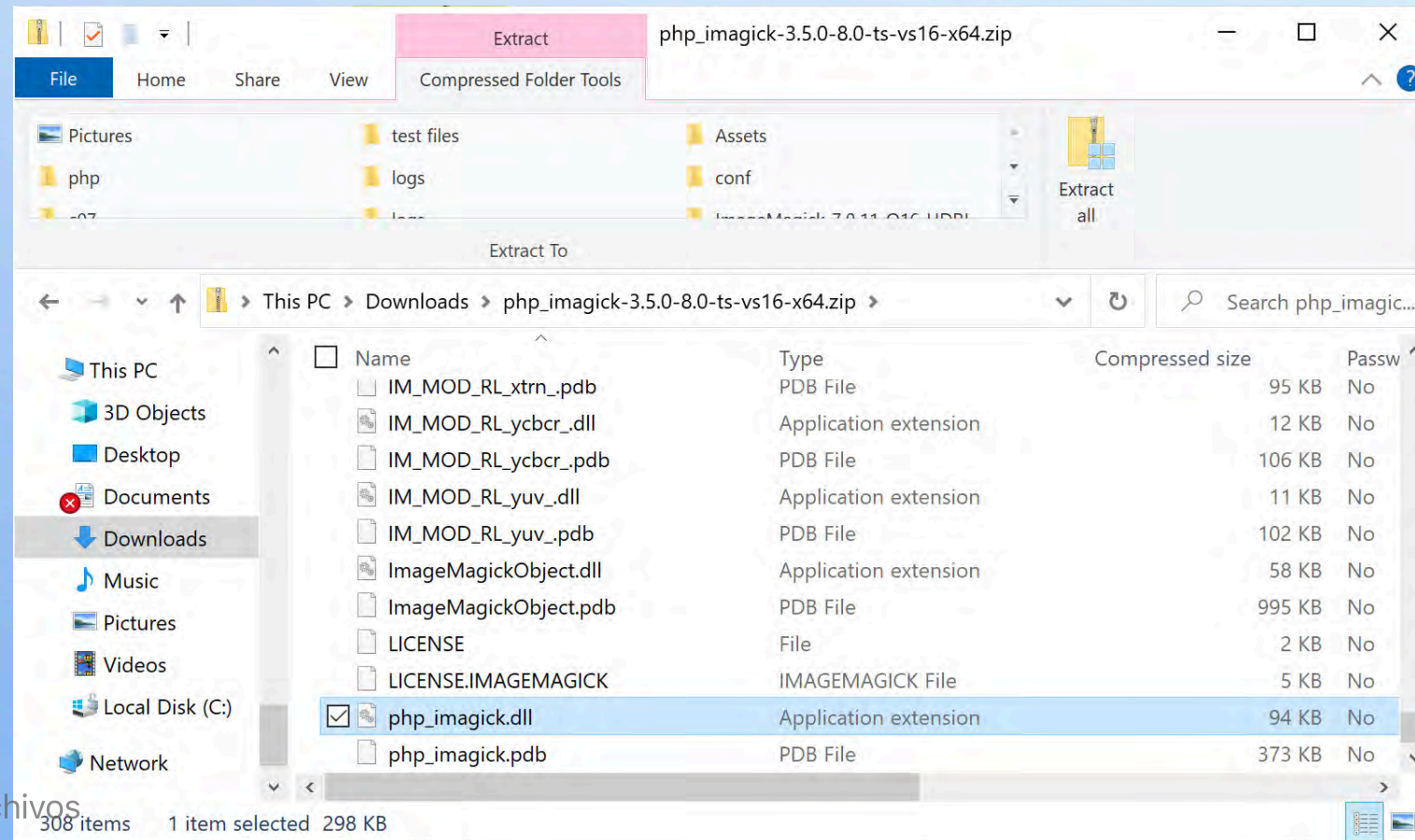
Copia los siguientes archivos de la descarga de Imagick, en la carpeta apache\bin:

- Todos los archivos con el prefijo `CORE_*` y el sufijo `.dll`
- Todos los archivos con el prefijo `IM_MOD_*` y el sufijo `.dll`

En total son unos 149 archivos.

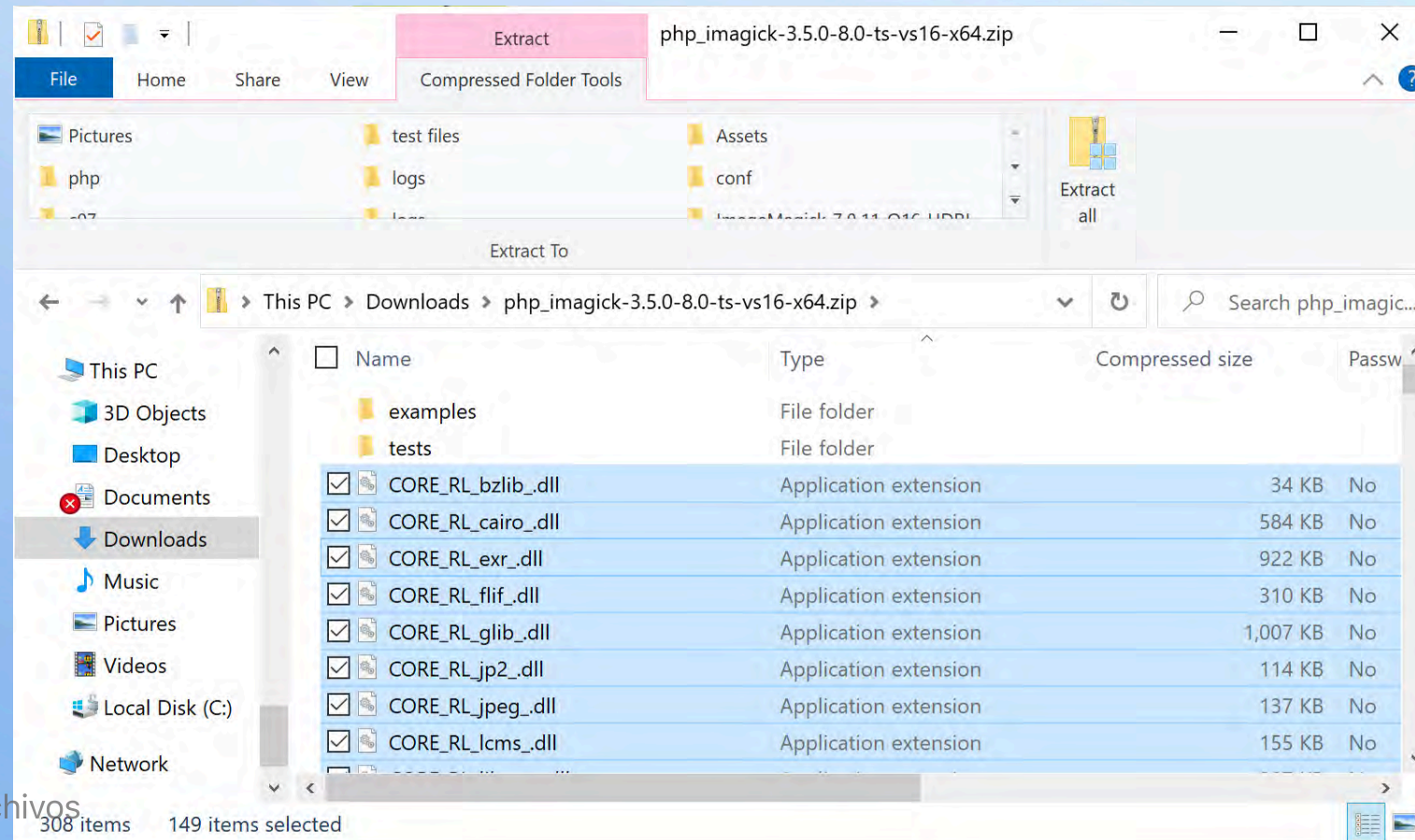
Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick



Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick



Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick

4. Habilitar Imagick en php.ini

Una vez instalada la extensión Imagick, hay que indicar al intérprete PHP que se ha instalado. Para ello, abre tu archivo php.ini.

Busca `extension=` y verás un montón de líneas que empiezan por ahí.

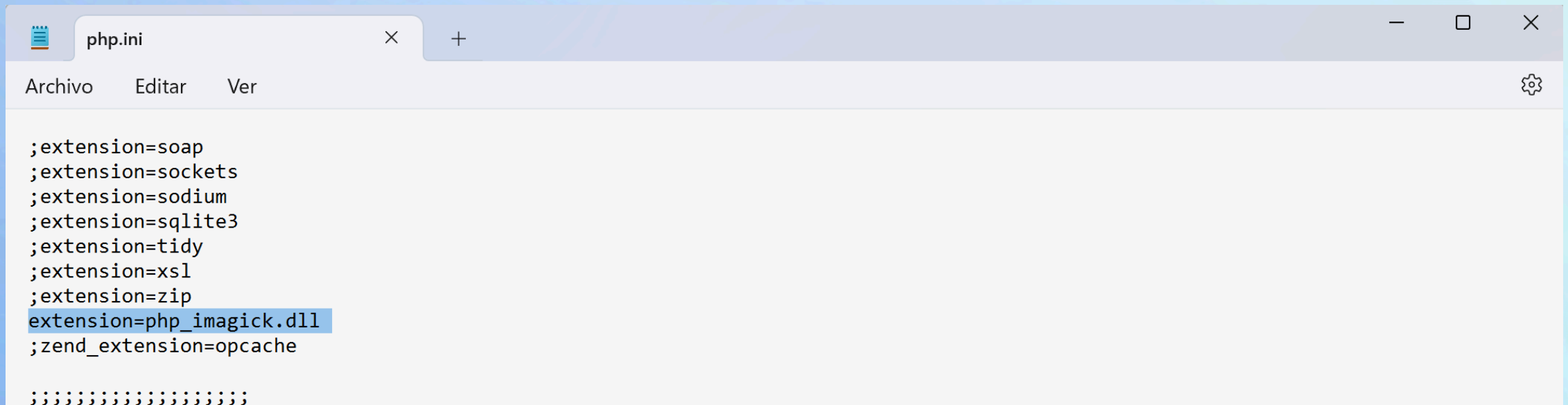
Añade lo siguiente en una nueva línea después de las que hay:

```
extension=php_imagick.dll
```

Guarda el archivo php.ini, luego reinicia XAMPP para aplicar los cambios que hiciste en el archivo php.ini.

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Instalando Imagick



A screenshot of a code editor window titled 'php.ini'. The window has a menu bar with 'Archivo', 'Editar', and 'Ver'. The code content is as follows:

```
;extension=soap
;extension=sockets
;extension=sodium
;extension=sqlite3
;extension=tidy
;extension=xsl
;extension=zip
extension=php_imagick.dll
;zend_extension=opcache

;;;;;;;;;;;;;;;;;
```

The line `extension=php_imagick.dll` is highlighted in blue.

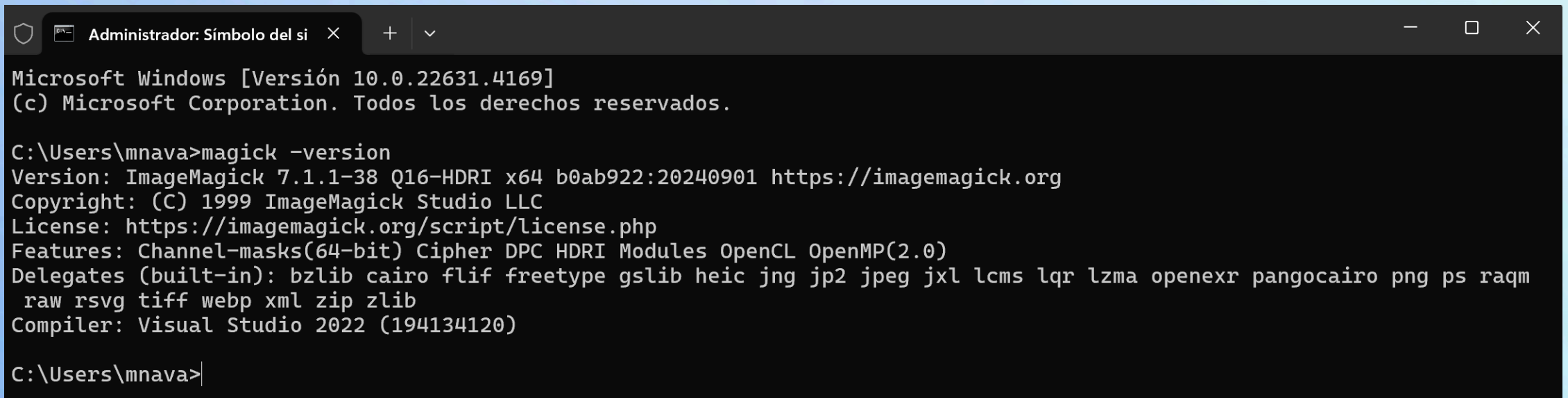
Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Comprobación de que ImageMagick está instalado

Para comprobar que ImageMagick está instalado, puedes escribir `magick -version` en la línea de comandos. Si aparece un mensaje indicando la versión, significa que está instalado.

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Comprobación de que ImageMagick está instalado



```
Administrador: Símbolo del si × + ▾  
Microsoft Windows [Versión 10.0.22631.4169]  
(c) Microsoft Corporation. Todos los derechos reservados.  
  
C:\Users\mnava>magick -version  
Version: ImageMagick 7.1.1-38 Q16-HDRI x64 b0ab922:20240901 https://imagemagick.org  
Copyright: (C) 1999 ImageMagick Studio LLC  
License: https://imagemagick.org/script/license.php  
Features: Channel-masks(64-bit) Cipher DPC HDRI Modules OpenCL OpenMP(2.0)  
Delegates (built-in): bzlib cairo flif freetype gslib heic jng jp2 jpeg jxl lcms lqr lzma openexr pangocairo png ps raqm  
raw rsvg tiff webp xml zip zlib  
Compiler: Visual Studio 2022 (194134120)  
  
C:\Users\mnava>
```

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Comprobar que Imagick está instalado

Para comprobar que Imagick está instalado, utiliza la función `phpinfo()` y busca el encabezado `imagick`. Si encuentras este encabezado seguido de una tabla con información como la versión, las clases soportadas y los formatos de imagen, entonces está instalado.

Anexo II: Cómo instalar Imagick e ImageMagick para XAMPP en un PC

Comprobación de que ImageMagick está instalado

imagick		
imagick module	enabled	
imagick module version	3.7.0	
imagick classes	Imagick, ImagickDraw, ImagickPixel, ImagickPixelIterator, ImagickKernel	
imagick compiled with ImageMagick version	ImageMagick 7.1.0-18 Q16 x64 2021-12-14 https://imagemagick.org	
imagick using ImageMagick library version	ImageMagick 7.1.0-18 Q16 x64 2021-12-14 https://imagemagick.org	
ImageMagick copyright	(C) 1999-2021 ImageMagick Studio LLC	
ImageMagick release date	2021-12-14	
ImageMagick number of supported formats:	263	
ImageMagick supported formats	3FR, 3G2, 3GP, AAI, AI, APNG, ART, ARW, ASHLAR, AVI, AVIF, AVS, BGR, BGRA, BGRO, BIE, BMP, BMP2, BMP3, BRF, CAL, CALS, CANVAS, CAPTION, CIN, CIP, CLIP, CLIPBOARD, CMYK, CMYKA, CR2, CR3, CRW, CUBE, CUR, CUT, DATA, DCM, DCR, DCRRAW, DCX, DDS, DFONT, DJVU, DNG, DOT, DPS, DPX, DXT1, DXT5, EMF, EPDF, EPI, EPS, EPS2, EPS3, EPSF, EPSI, EPT, EPT2, EPT3, ERF, EXR, FARBFELD, FAX, FF, FILE, FITS, FL32, FLIF, FLV, FPX, FRACTAL, FTP, FTS, G3, G4, GIF, GIF87, GRADIENT, GRAY, GRAYA, GROUP4, GV, HALD, HDR, HEIC, HEIF, HISTOGRAM, HRZ, HTM, HTML, HTTP, HTTPS, ICB, ICO, ICON, IIQ, INFO, INLINE, IPL, ISOBRL, ISOBRL6, J2C, J2K, JBG, JBIG, JNG, JNX, JP2, JPC, JPE, JPEG, JPG, JPM, JPS, JPT, JSON, JXL, K25, KDC, KERNEL, LABEL, M2V, M4V, MAC, MAP, MASK, MAT, MATTE, MEF, MIFF, MKV, MNG, MONO, MOV, MP4, MPC, MPEG, MPG, MRW, MSL, MSVG, MTV, MVG, NEF, NRW, NULL, ORA, ORF, OTB, OTF, PAL, PALM, PAM, PANGO, PATTERN, PBM, PCD, PCDS, PCL, PCT, PCX, PDB, PDF, PDFa, PEF, PES, PFA, PFB, PFM, PGM, PGX, PHM, PICON, PICT, PIX, PJPEG, PLASMA, PNG, PNG00, PNG24, PNG32, PNG48, PNG64, PNG8, PNM, POCKETMOD, PPM, PS, PS2, PS3, PSB, PSD, PTIF, PWP, RADIAL-GRADIENT, RAF, RAS, RAW, RGB, RGB565, RGBA, RGBO, RGF, RLA, RLE, RMF, RSVG, RW2, SCR, SCREENSHOT, SCT, SFW, SGI, SHTML, SIX, SIXEL, SPARSE-COLOR, SR2, SRF, STEGANO, SUN, SVG, SVGZ, TEXT, TGA, THUMBNAIL, TIFF, TIFF64, TILE, TIM, TM2, TTC, TTF, TXT, UBRL, UBRL6, UIL, UYVY, VDA, VICAR, VID, VIFF, VIPS, VST, WBMP, WEBM, WEBP, WMF, WMV, WPG, X3F, XBM, XC, XCF, XPM, XPS, XV, YAML, YCbCr, YCbCrA, YUV	

Directive	Local Value	Master Value
imagick.allow_zero_dimension_images	0	0
imagick.locale_fix	0	0
imagick.progress_monitor	0	0
imagick.set_single_thread	1	1
imagick.shutdown_sleep_count	10	10
imagick.skip_version_check	0	0