UD-06a: Generación dinámica de imágenes

Desarrollo Web en Entorno Servidor

Curso 2020/2021

Introducción

Existe un gran abanico de posibilidades que pueden dotar a la página de **contenido multimedia** (imágenes y sonidos) **y que, además, pueden ser procesados por PHP.**

En cuanto a interacción con las **imágenes**, PHP puede **realizar gráficos de barras** para mostrar resultados estadísticos, **añadir una marca de agua** a todas nuestras fotografías o **mostrar una réplicas reducidas** de un conjunto de imágenes.

PHP incorpora su propia versión de la biblioteca GD2 para crear nuevas y **manipular imágenes**. También permite usar otras bibliotecas más complejas y con más funcionalidad: ***ImageMagick***, ***Gmagick*** y ***Cairo***.

Librería GD

La **librería gráfica GD** permite crear y manipular gráficos fácilmente. **Permite importar y exportar gráficos de distintos tipos** (**GIF, JPG y PNG**).

Los archivos GIF y PNG guardan en memoria el conjunto completo de píxeles con su color concreto y comprimen el resultado para que el archivo no sea muy pesado. Los archivos JPG también están comprimidos, pero se basan en algoritmos complejos que permiten un mayor rendimiento.

Para ver si tenemos operativa la libreria GD y saber su versión podemos visualizar el array que devuelve la función *gd_info()* con print_r.

http://docs.php.net/manual/es/function.gd-info.php

Librería GD

Tipos MIME

El estándar MIME actualmente se usa para **describir el tipo de archivo que enviamos a través de Internet**.

Es importante conocer los tipos MIME para trabajar con las librerías GD. En general, antes de recibir la respuesta del servidor, recibiremos una cabecera anunciando el tipo de contenido que recibiremos, mediante Content-Type.

Por defecto, se envía el tipo MIME text/html (documento HTML). Sin embargo, si necesitamos usar la librería GD, la cabecera tendrá que ser del tipo de la imagen:

```php
<?php
    header("Content-Type: image/png");
?>
```

Formato de imagen	MIME
JPG	image/jpeg
PNG	image/png
GIF	image/gif
BMP	image/bmp
SVG	image/xml+svg

Librería GD

Mostrar una imagen en pantalla

En HTML usamos la etiqueta <img> para mostrar una imagen: .

La librería gráfica GD también permite **enviar imágenes mediante ciertas funciones**.

```php
<?php
    // Creación de la imagen
    $imagen = imagecreatefrompng("php_logo.png");
     // crea una nueva imagen a partir de un fichero o de una URL.
    // Envío de la imagen
    header("Content-Type: image/png");
     // envia la cabecera del tipo de archivo gráfico que se quiere mostrar.
    imagepng($imagen);
     // envia la imagen con imagepng().
    // Liberación de la imagen de memoria
    imagedestroy($imagen);
?>
```

Librería GD

Crear imágenes

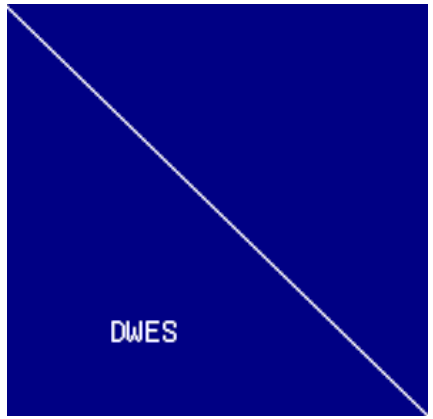
¿Por qué puede interesarnos generar imágenes desde el servidor?

- Aspectos de apariencia de la página
- Personalización de imágenes subidas por el usuario (por ejemplo, un texto).
- Generación de marcas de agua.
- Generación de gráficas estadísticas.
- Visualización de un CAPTCHA.
- Visualizar un texto que no queremos que se pueda copiar

El proceso de creación de una imagen presenta 4 pasos básicos:

1. **Crear** un lienzo sobre el que trabajar.
2. **Manipular** formas e imprimir texto en dicho lienzo.
3. **Generar** *(Enviar)* el gráfico final.
4. **Liberar** los recursos.

```php
<?php
    // a. Creación y configuración del lienzo
    $alto = 200;
    $ancho = 200;
    $imagen = imagecreatetruecolor($ancho, $alto);
    $blanco = imagecolorallocate($imagen, 255, 255, 255);
    $azul = imagecolorallocate($imagen, 0, 0, 164);

    // b. Dibujamos la imagen
    imagefill($imagen, 0, 0, $azul);
    imageline($imagen, 0, 0, $ancho, $alto, $blanco);
    imagestring($imagen, 4, 50, 150, 'DWES', $blanco);

    // c. Generación de la imagen
    header('content-type: image/png');
    imagepng ($imagen);

    // d. Liberamos la imagen de memoria
    imagedestroy($imagen);
?>
```



Librería GD

a. Crear un lienzo

Como vimos en el ejemplo anterior, para crear un lienzo se ha usado la función:

```
$imagen = imagecreatetruecolor($ancho, $alto);
```

Esta función crea un lienzo bidimensional, con color real RGB.

También podemos crear una imagen partiendo de otra, como en ejemplo de *pág. 5*, con **imagecreatefromgif**(), **imagecreatefromjpeg**(), **imagecreatefrompng**(), etc. donde las dimensiones serán las de la imagen cargada.

Además, existe la función **imagecreatefromstring**() que te permite generar una **imagen a partir de un campo *(imagen)* almacenado en una base de datos**.

Librería GD

b. Dibujar en la imagen

En primer lugar, hemos seleccionado los colores a utilizar en la imagen:

```
$blanco = imagecolorallocate($imagen, 255, 255, 255);
$azul = imagecolorallocate($imagen, 0, 0, 164);
```

Hay que pasarle el identificador de la imagen como primer parámetro y después el **color RGB** que usaremos. Esta función devuelve un código para el color que usaremos en las funciones de dibujo.

También podemos pasar un último parámetro alpha que indicará la opacidad.
Irá de 0 a 127 siendo 0 = opaco y 127 = transparente.

```
int imagecolorallocatealpha(resource img, int red, int green, int blue, int alpha);
```

Librería GD

Una vez seleccionado el color, podemos realizar un dibujo vectorial para líneas, arcos, polígonos o texto. En todas ellas se necesitan los siguientes parámetros:

- Identificador de la imagen.
- Coordenada de inicio (y, en ocasiones, de final).
- Color con que dibujar.
- Si se trata de un texto, información de fuente a utilizar.

En el ejemplo anterior, primero rellenamos el fondo de azul, comenzando en la coordenada dada (0,0). A continuación, dibujamos una línea entre dos puntos dados con el color que le pasamos.

```
imagefill($imagen, 0, 0, $azul);
imageline($imagen, 0, 0, $ancho, $alto, $blanco);
```

Finalmente, dibujamos un texto horizontal en la coordenada (**50**, **150**) usando **una fuente con tamaño que puede tener el valor de 1 a 5.**

```
imagestring($imagen, 4, 50, 150, 'DWES', $blanco);
```

Librería GD

c. Generar el gráfico final

El tipo se envía al navegador **indicando que vamos a representar una imagen**:

```
header('content-type: image/png');
```

Tras enviar el encabezado, devolvemos la imagen con:

```
imagepng($imagen);
```

También hay otras **funciones para otros formatos**: `imagejpeg`, `imagegif`, …

d. liberar recursos

Es importante eliminar de memoria la imagen generada dinámicamente con:

```
imagedestroy($imagen);
```

Si no se hace, y se crean muchas imágenes en servidor, **podría quedarse sin memoria**.

Librería GD

Transformación de imágenes: escalado y tipo

```php
<?php
    // Creamos una imagen cargándola desde el archivo gif
    $img_org = imagecreatefromgif("instituto.gif");

    // Obtenemos la mitad de las dimensiones de la imagen origen
    $ancho_dst = intval(imagesx($img_org) / 2);
    $alto_dst = intval(imagesy($img_org) / 2);

    // Creamos un lienzo para la imagen destino con las dimensiones calculadas
    $img_dst = imagecreatetruecolor($ancho_dst, $alto_dst);

    /* Escalamos la imagen gif origen sobre la imagen nueva destino especificando
       punto de inicio para destino y origen, y punto final para destino y origen. */
    imagecopyresampled($img_dst, $img_org, 0, 0, 0, 0,
                       $ancho_dst, $alto_dst, imagesx($img_org), imagesy($img_org));

    // Damos salida a la imagen final cambiando el formato a png
    header("Content-type: image/png");
    imagepng($img_dst);

    // Destruimos ambas imágenes
    imagedestroy($img_org);
    imagedestroy($img_dst);
?>
```

Librería GD

Transformación de imágenes: rotación

Para **rotar imágenes** tenemos la función:

```
imagerotate (resource $image, float $angle, int $background_color) : resource
```

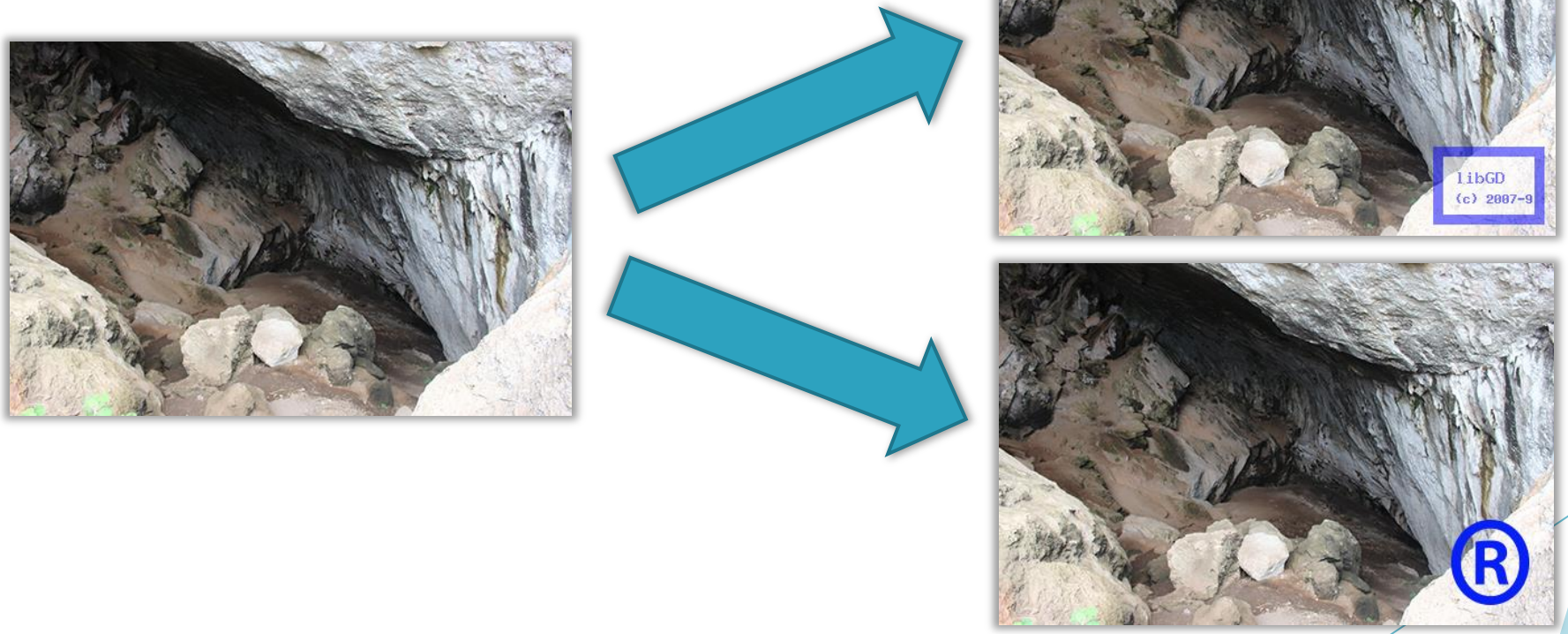
Devuelve una imagen rotada en **sentido anti-horario**, según el ángulo en grados que le pasemos como parámetro. El parámetro **color_fondo** especifica el **color de fondo de la zona no cubierta tras la transformación**.

El centro de la rotación es el centro de la imagen, y la imagen rotada es escalada para que quepa en el lienzo original sin recortar las esquinas de la imagen rotada.

Librería GD

Generar una marca de agua

Puede ser un texto o un gráfico insertado sobre la fotografía original.



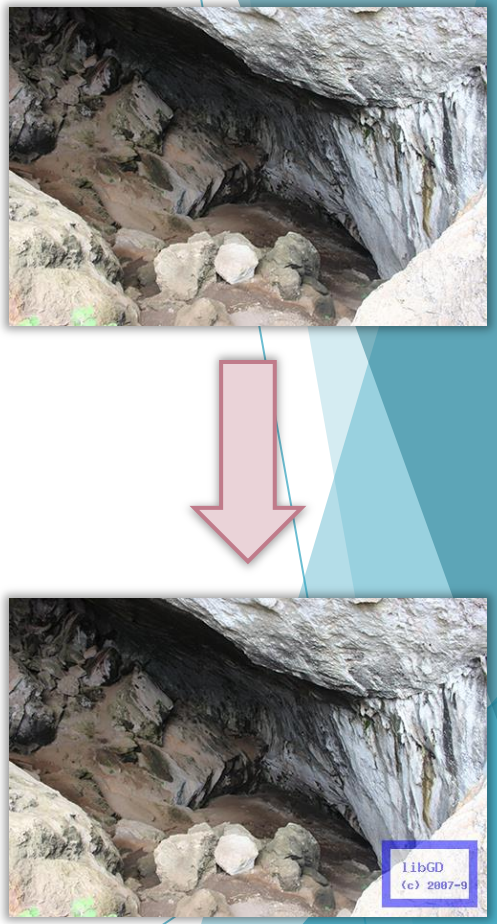
Ejemplo: marca de agua con una imagen.

```php
<?php
    // Cargar la estampa y la foto para aplicarle la marca de agua
    $estampa = imagecreatefrompng('registro.png');
    $img = imagecreatefromjpeg('cueva.jpg');

    // Establecer los márgenes para la estampa
    $margen_dcho = 10;
    $margen_inf = 10;

    // obtener el alto/ancho de la imagen de la estampa
    $sx = imagesx($estampa);
    $sy = imagesy($estampa);

    // Copiar la imagen de la estampa sobre nuestra foto usando los índices de
    // márgen y el ancho de la foto para calcular la posición de la estampa.
    imagecopy($img, $estampa,
              imagesx($img) - $sx - $margen_dcho,  imagesy($img) - $sy - $margen_inf,
              0, 0, imagesx($estampa), imagesy($estampa));

    // Imprimir y mostrar
    header('Content-type: image/png');
    imagepng($img);

    // Guardar en un archivo nuevo
    imagepng($img, $cueva_registrada.png);

    // Liberar memoria
    imagedestroy($img);
?>
```

https://www.php.net/manual/es/function.imagecopy.php

Ejemplo: marca de agua **con texto**.

```php
<?php
    // idem al anterior pero modificando la variable $estampa
    …

    // Primero crearemos nuestra imagen de la estampa manualmente desde GD
    $estampa = imagecreatetruecolor(100, 70);

    // dibuja un rectangulo con relleno
    imagefilledrectangle($estampa, 0, 0, 99, 69, 0x0000FF);
    imagefilledrectangle($estampa, 9, 9, 90, 60, 0xFFFFFF);

    //dibuja un texto horizontal
    imagestring($estampa, 5, 20, 20, 'libGD', 0x0000FF);
    imagestring($estampa, 3, 20, 40, '(c) 2007-9', 0x0000FF);

    …

    // fusiona la estampa con nuestra foto con una opacidad del 50%
    imagecopymerge($img, $estampa,
                imagesx($img) - $sx - $margen_dcho,  imagesy($img) - $sy - $margen_inf,
                0, 0, imagesx($estampa), imagesy($estampa), 50);
    …

?>
```



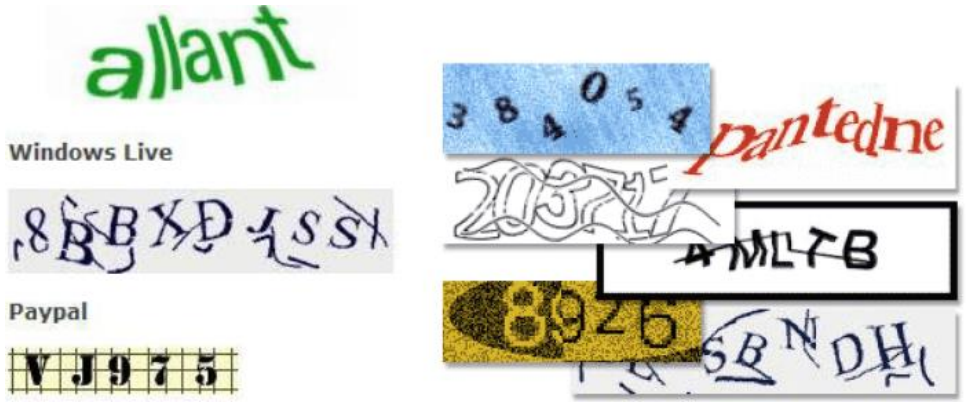
Librería GD

Generación de un CAPTCHA

Completely **A**utomated **P**ublic **T**uring-test to tell **C**omputers and **H**umans **A**part.

Prueba de Turing pública y completamente automatizada para diferenciar a compuntadoras y humanos.

Es una **prueba** desafío-respuesta **para determinar si el usuario es o no humano**. Consiste en que el usuario introduzca un **conjunto de caracteres que se muestran en una imagen distorsionada** que aparece en pantalla. Se supone que una máquina no es capaz de comprender la secuencia de forma correcta.



Librería GD

Generar un CAPTCHA

Podríamos resumir el proceso en 6 pasos:

1. Generamos un texto aleatorio para el CAPTCHA.
2. El texto se imprime en la imagen.
3. Se almacena en alguna variable de sesión.
4. Se muestra la imagen.
5. El usuario introduce el código.
6. Se comprueba que el código es igual al almacenado en la sesión.

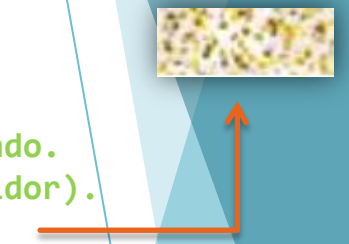
EJEMPLO: creando un CAPTCHA paso a paso.

Paso 1: Creamos Texto Aleatorio

```php
<?php
    session_start();
    // Creación de cadena aleatoria
    // mktime: marca de tiempo Unix de una fecha
    $crypt = md5(mktime() * rand());

    // Solo necesitamos 5 caracteres aleatorios.
    $string = substr($crypt, 0, 5);
?>
```

Paso 2: Creamos la imagen

```php
<?php
    // Creamos una imagen partiendo de una de fondo.
    // Debemos subir una imagen de fondo al servidor).
    $captcha = imagecreatefrompng("captcha.png");
    // Colores usados para generan las líneas (RGB).
    $brown = imagecolorallocate($captcha, 80, 70, 30);
    // Añadimos líneas a nuestra imagen.
    imageline($captcha, 0, 0, 39, 29, $brown);
    imageline($captcha, 40, 0, 64, 29, $brown);
?>
```



Paso 3: Insertamos Texto en la imagen

```php
<?php
  // Escribimos la cadena aleatoriamente en la imagen
  imagestring($captcha, 5, 20, 10, $string, $brown);

  // Encriptamos y almacenamos el valor en una variable de sesión
  $_SESSION['key'] = md5($string);

  // Devolvemos la imagen para mostrarla
  header("Content-type: image/png");
  imagepng($captcha);

?>
```

Paso 4: Chequeamos el valor dado por el usuario en el formulario

```php
<?php
  session_start();

  if ( md5($_POST['code']) != $_SESSION['key'] ) {
     die("Error: No has introducido el código correcto.");
  } else {
     echo 'Correcto, parece que eres un humano.';
  }
?>
```

EJERCICIO

Crea una nueva **carpeta** llamada **ud06ejer01** que contendrá varios archivos en los que probarás los ejemplos de este pdf y personalizarás a tu gusto, modificando parcialmente el ejemplo original.