GOBIERNO DE ESPAÑA
MINISTERIO DE EDUCACIÓN

PHP AVANZADO

PROGRAMACIÓN

AULA MENTOR

educacion.es

Nipo: 660-08-008-2

Autoría:
Clodoaldo Robledo Sacristán
David Robledo Fernández

Edición y maquetación:
Amalia B. Dueñas Luengo

Diseño gráfico de portada:
Lorena Gordoa López

Teoría: Tratamiento de gráficos

1 - Objetivos

Objetivos
• **Aprender a tratar gráficos** desde PHP. • **Conocer y saber aplicar las principales funciones de la librería GD** para incorporar gráficos e imágenes dentro del código PHP. • **Conocer y saber utilizar las funciones básicas de la librería Ming** para incluir películas animadas Shockwave Flash.

2 - Tratamiento de imágenes y gráficos en PHP

En el Curso de iniciación a PHP 5 hemos usado algunas imágenes y tratado gráficos sólo de una forma elemental y sin detalle. En esta **Unidad 1** del **Curso avanzado de PHP 5** vamos a estudiar con mayor detenimiento este tema.

La incorporación de componentes gráficos y de imágenes a las páginas web comerciales y profesionales es algo fundamental e imprescindible, no sólo para mejorar estéticamente la presencia de los portales haciéndoles más atractivos y sugerentes, sino también para permitir un acceso más rápido e intuitivo o para presentar la información de una forma más clara y ordenada. En general, la aparición de elementos gráficos en una página mejora mucho su presentación y eficacia.

En el capítulo 62 del **Manual de PHP 5** , denominado "Funciones para imágenes", se explica que PHP no sólo puede generar páginas dinámicas web para HTML, sino que, además, permite crear y manipular ficheros de imágenes en diferentes formatos: PNG, JPG, WBMP, XPM, etcétera. PHP puede incluso mandar flujos de imágenes directamente al navegador.

Para que esto sea posible, es necesario compilar PHP con la biblioteca de funciones de imágenes **GD Graphics Library**. En este curso avanzado ya se ha previsto esto, por lo cual el alumn@ sólo necesita instalar el CD del curso para que éste funcione correctamente al crear o tratar ficheros de imagen.

La librería **GD Graphics Library** no es exclusiva de PHP, sino que puede usarse con cualquier otro lenguaje de programación. Puede recogerse en la dirección http://www.libgd.org/ y utilizarse de forma libre y gratuita. Es una librería de tipo ANSI C que permite, como hemos dicho, crear imágenes dinámicas en formatos PNG y JPEG, básicamente, si bien también puede hacerlo en otros formatos, como los citados anteriormente.

En general, la escritura del código necesario para la creación de imágenes no es demasiado difícil. Utilizando el Ejemplo 1 veamos los pasos que debemos dar para generar una imagen desde PHP.

1. En la cabecera del script (segunda página de este ejemplo) indicamos que se va a crear una imagen del tipo que se especifique, en este caso **jpeg**:

```
Header( "Content-type: image/jpeg");
```

2. Creamos la imagen usando alguna de las funciones que realizan esta operación:

```
$imagen = imagecreate(400, 150);
```

La función `imagecreate(x,y)` devuelve un identificador de imagen de tipo numérico entero que contiene una imagen en blanco con las medidas especificadas en los dos parámetros, que representan las coordenadas **`x`** (anchura de izquierda a derecha) e **`y`** (altura de arriba abajo) de la imagen. El valor de este identificador lo asignamos a la variable `$imagen`.

3. Diseñamos la imagen según la apariencia que debe tener en cuanto a:

Color

```
$color_azul = ImageColorAllocate($imagen,0,0,255);
$color_blanco = ImageColorAllocate($imagen,255,255,255);
$color_violeta = ImageColorAllocate($imagen,192,192,255);
```

La función `ImageColorAllocate(identificador de imagen,rojo,verde,azul)` devuelve un identificador del color representado por la mezcla de los componentes **RGB** (Red, Green, Blue) especificados. Más adelante, dentro de esta misma Unidad, se explica cómo crear colores mixtos mezclando los tres colores básicos.

Luego, rellenamos el fondo de la imagen completo con el color azul:

```
ImageFilledRectangle($imagen,0,0,400,150,$color_azul);
```

La función `ImageFilledRectangle(id. imagen, x1, y1, x2, y2, id. color)` crea en la imagen un rectángulo relleno con el color indicado comenzando con la coordenada superior izquierda x1,y1 y acabando en la coordenada inferior derecha x2,y2. 0,0 es la esquina superior izquierda de la imagen.

Con la misma función dibujamos otro rectángulo en la parte derecha de la imagen, de color violeta, que al sobreponerse en parte sobre el color azul anterior queda de color plateado:

```
ImageFilledRectangle($imagen,200,20,380,130,$color_violeta);
```

Borde

Por dentro del rectángulo más grande que hemos rellenado de color azul dibujamos una línea de forma rectangular a modo de reborde:

```
ImageRectangle($imagen,5,10,390,140,$color_blanco);
```

La función `ImageRectangle(id. imagen, x1, y1, x2, y2, id. color)` dibuja un rectángulo del color especificado comenzando en la coordenada superior izquierda x1,y1 y acabando en la coordenada inferior derecha x2,y2. 0,0 es la esquina superior izquierda del rectángulo.

Texto

Dentro de la imagen podemos incluir un texto fijo o que pasemos desde otra página en la que el usuario lo ha escrito a través de un formulario, como en este ejemplo. En todo caso, hay que cuidar que el tamaño de la imagen sea adecuado al texto que se quiere mostrar en la misma.

En nuestro ejemplo, en primer lugar escribimos cinco veces con un bucle el texto que se ha pasado desde el formulario de la primera página:

```
for ($i=1; $i <= 5; $i++)
    ImageString($imagen,$i,15,$i*20,$_REQUEST["palabra"],$color_blanco);
```

La función `ImageString(id. imagen,id. fuente,x,y,var,id. color)` dibuja la cadena var dentro de la imagen desde la coordenada x,y (0,0 es la posición superior) con el color indicado. Si la fuente es 1, 2, 3, 4 o 5, como en este caso, se emplea una fuente interna.

Después, con el mismo procedimiento dibujamos el texto fijo:

```
ImageString($imagen,5,235,52,"CURSO DE PHP 5 ",$color_azul);

ImageString($imagen,5,245,72,"(avanzado)",$color_azul);
```

En este paso, como es lógico, pueden incluirse muchas cosas más, a medida del usuario, para que la imagen resulte más llamativa, completa o estéticamente más acertada.

4. Enviamos la imagen al navegador del usuario con la orden

```
Imagejpeg($imagen, "", 98);
```

La función `imagejpeg(id. imagen[, nombre de fichero], índice de calidad])` envía la imagen creada a la pantalla o a un fichero. Los dos últimos parámetros son opcionales. Más adelante se explica con detalle esta función.

5. Destruimos la imagen con la orden

```
ImageDestroy($imagen);
```

Una vez creada la imagen y mostrada o guardada en un fichero, conviene eliminar de la memoria ésta liberando espacio que puede ser necesario para otras operaciones.

En el Ejemplo 2 no creamos directamente una imagen, sino que de una forma un tanto curiosa aprovechamos diez imágenes de tipo PNG para combinarlas de forma que se muestre un contador que tiene apariencia de imagen completa con la información almacenada en un fichero de texto. Conviene que el alumn@ estudie y comprenda bien el código fuente, pues en el mismo se usan funciones de lectura y escritura en ficheros de texto, así como otros elementos del lenguaje PHP que permiten articular una sintaxis eficiente y breve, como en la instrucción

```
$resultado_html .=  "<IMG SRC=\"$IMG_DIGITS$count[$i].png\">";
```

3 - Funciones más importantes para tratar gráficos

En el apartado anterior ya hemos usado y comentado algunas funciones que permiten crear, dibujar o mostrar gráficos en la pantalla o mandarlos a un fichero. En este apartado vamos a explicar de una forma más sistemática las principales funciones de tratamiento de gráficos agrupándolas según el tipo de operación que llevan a cabo.

En el capítulo **LXII. Funciones para imágenes** del Manual de PHP 5 se abordan todas las funciones que permiten crear imágenes. En este **Curso avanzado** el alumn@ debe recurrir al Manual de PHP siempre que lo precise, pues en los contenidos del curso no pretendemos abordar todo. Además, hay páginas web en Internet que tratan también estos contenidos.

1. Funciones que permiten crear imágenes

La función **`imagecreate(x,y)`** devuelve un identificador de imagen de tipo numérico entero que contiene una imagen en blanco con las medidas especificadas en los dos parámetros, que representan las coordenadas **`x`** (anchura de izquierda a derecha) e **`y`** (altura de arriba abajo) de la imagen. Ya la hemos utilizado y explicado en el Ejemplo 1. Lógicamente, la usaremos en la mayoría de los ejercicios de esta Unidad.

Esta función debe usarse cuando se quiere partir de cero, es decir, cuando se crea una imagen que antes no existía. En el Ejemplo 3 también se usa esta función. A la vez que el alumn@ lo comprueba, mire cómo creamos una imagen sencilla que representa un reloj. Tomamos la hora del sistema y cada vez que se presenta o actualiza la página se muestra la hora real que tenga el ordenador. En este ejercicio se utilizan, además, otras funciones que explicaremos después y que sirven para diseñar imágenes, en este caso la del reloj. Estudiando el código de este ejercicio el alumno o alumna puede ver en la práctica cómo se

escribe la sintaxis de las diferentes instrucciones.

Hay también funciones que permiten **crear o modificar una imagen a partir de otra ya existente**, archivada en un fichero, o desde una URL. Veamos algunas de las más usadas.

La función **`imagecreatefromjpeg(`**`nombre de fichero`**`)`** devuelve un identificador de imagen que referencia a la imagen contenida en el archivo indicado o URL, que debe ser de tipo JPG. Así, pues, se crea una nueva imagen a partir de la original. Esta función devuelve una cadena vacía si la operación no se ha podido llevar a cabo, si por ejemplo no existiera el fichero especificado. En el Manual de PHP se ofrece una forma elegante de resolver el mensaje de error que se muestra en caso de fallo.

El formato de imágenes JPG o JPEG (*Joint Photographics Expert Group File Interchange Format*) permite usar hasta 16.777.216 colores (24 bits). Es el formato más adecuado para comprimir imágenes fotográficas con todo detalle. Los índices de compresión son mucho mayores que los conseguidos con el formato GIF. No obstante, al comprimir imágenes de este formato se pierde información, por lo cual, al descomprimir el fichero donde se han guardado, las imágenes obtenidas no son exactamente iguales que las originales. Se puede obtener más información sobre formatos gráficos en http://es.wikipedia.org/wiki/Joint_Photographic_Experts_Group y en http://pub.ufasta.edu.ar/SISD/jpeg/jpg.htm.

En el Ejemplo 4 se usa también esta función para crear cuatro nuevas imágenes a partir de otras ya existentes. Conviene estudiar este código y mirar cómo usamos esta función y otras que se explicarán después.

PHP dispone de funciones que permiten crear imágenes nuevas a partir de otros formatos gráficos. Funcionan prácticamente de la misma forma que las imágenes JPG, por lo cual sólo citamos las más importantes.

La función **`imagecreatefromgif(`**`nombre de fichero`**`)`** devuelve un identificador de imagen que referencia a la imagen contenida en el archivo indicado o URL, que debe ser de tipo GIF. El formato de imágenes GIF (*Graphic Interchange format*), creado por la empresa Compuserve, utiliza un sistema de compresión (LZW) sin pérdida de información de las imágenes. Hay que tener en cuenta que el soporte para el formato GIF ha sido eliminado en la librería GD, a partir de la versión 1.6 y vuelta a incluir en la versión 2.0.28 de este curso, por lo cual esta función está disponible para usarla en el curso ya que usamos la versión de GD más reciente.

La función **`imagecreatefrompgn(`**`nombre de fichero`**`)`** devuelve un identificador de imagen que referencia a la imagen contenida en el archivo indicado o URL, que debe ser de tipo PNG. El formato de imágenes PNG (*Portable Network Graphic*) nace debido a los problemas de patente del algoritmo de compresión LZW (Lempel-Ziv-Welch) que emplean las imágenes de tipo GIF, utilizadas desde 1987, para seguir disponiendo de un formato gráfico de difusión gratuita. El formato PNG sólo es soportado por los navegadores más recientes. Se recomienda emplear MS Explorer 4.0 o superior para ver imágenes de este formato.

La función **`imagecreatefromwbmp(`**`nombre de fichero`**`)`** devuelve un identificador de imagen que referencia a la imagen contenida en el archivo indicado o URL, que debe ser de tipo BMP. Este formato estándar de imágenes de mapa de bits se usa en los equipos compatibles con Windows. Las imágenes de mapa de bits se pueden guardar en sistemas Windows u OS/2 y aceptan colores de 24 bits.

La función **`imagecreatetruecolor(x,y)`** devuelve un identificador de imagen de tipo numérico entero que referencia a una imagen en blanco con las medidas especificadas en los dos parámetros, que representan las coordenadas **`x`** (anchura de izquierda a derecha) e **`y`**(altura de arriba abajo) de la imagen. Es una función equivalente a la función `imagecreate(x,y)`.

2. Funciones que permiten elaborar imágenes

El segundo paso consiste en elaborar la imagen que se quiere crear. Para ello hay que determinar con todo tipo de detalle las características de la misma, como tamaño, color, formas geométricas, dibujos, etcétera. Veamos ahora las principales funciones que permiten elaborar imágenes.

La función **`imagearc`**`(id. imagen,x,y,anchura,altura,inicio,final,color)` dibuja una elipse parcial (arco)

centrada en `x,y` (la esquina superior izquierda es 0,0) según el tamaño indicado en `anchura` y `altura` (ambas referidas a la elipse), desde `inicio` hasta `final` (ambos indicados en grados) del `color` especificado. Todos los parámetros son de tipo numérico entero.

En el Ejemplo 6 usamos esta función conjuntamente con otras muchas de este tipo aplicadas a la elaboración de gráficos. Por ejemplo, la instrucción `imagearc($imagen, 15, 20, 30, 30, 90, 150, $rojo)` dibuja un arco de color rojo centradas en el punto de las coordenadas 15, 20 de 30 por 30 pixels (la elipse completa tendría forma de circunferencia) desde 90 hasta 150 grados.

La función **`imageellipse`**`(id. imagen,x,y,anchura,altura,color)` dibuja una elipse centrada en `x,y` (la esquina superior izquierda es 0,0) según el tamaño indicado en `anchura` y `altura` del `color` especificado. Todos los parámetros son de tipo numérico entero. Como vemos, con `imagearc()` se puede conseguir prácticamente lo mismo.

La función **`imagerectangle`**`(id. imagen,x1,y1,x2,y2,color)` dibuja un rectángulo del `color` especificado cuyo ángulo superior izquierdo está en la coordenada **x1,y1** y cuyo ángulo inferior derecho está en la coordenada `x2,y2`.

La función **`imageline`**`(id. imagen,x1,y1,x2,y2,color)` dibuja una línea recta continua del `color` especificado desde la coordenada `x1,y1` hasta la coordenada `x2,y2`.

La función **`imagedashedline`**`(id. imagen,x1,y1,x2,y2,color)` dibuja una línea recta discontinua del `color` especificado desde la coordenada `x1,y1` hasta la coordenada `x2,y2`.

La función **`imagechar(`**`id. imagen,fuente,x,y,cadena,color`**`)`** dibuja horizontal-mente en la imagen el primer carácter de `cadena` con su esquina superior izquierda en `x,y` (arriba izquierda es 0,0) del `color` especificado. En el tamaño de la fuente hay que tener en cuenta lo siguiente:

1. Si el valor de la fuente es 1, 2, 3, 4 ó 5, se usa una fuente predefinida interna. El tamaño menor de ésta corresponde al valor 1, y el mayor corresponde al valor 5.
2. Como hemos hecho en el Ejemplo 6 y se hace, igualmente, en el Ejercicio 5, podemos utilizar fuentes externas. Para esto necesitamos disponer de ese tipo de fuente e indicar el camino donde puede encontrarse. La utilización de fuentes externas ha dado bastantes problemas hasta la aparición de la versión 4.3.0 de PHP, que es la que usamos en este curso.

La función **`imagecharup(`**`id. imagen,fuente,x,y,cadena,color`**`)`** dibuja vertical-mente en la imagen el primer carácter de `cadena` con su esquina superior izquierda en `x,y` (arriba izquierda es 0,0) del `color` especificado.

Complementaria de las dos funciones anteriores es la función **`ImageLoadFont`**`(fichero)`, que carga una fuente de mapa de bits definida por el usuario. Devuelve un identificador de esa fuente, que siempre debe ser mayor que 5, para que no pueda entrar en conflicto con las fuentes predefinidas.

El color tanto de las fuentes como de los dibujos o de los fondos es un factor estético importante de los gráficos. Veamos ahora algunas de la principales **funciones que permiten dar colorido** a éstos.

La función **`imagecolorallocate`**`(id. imagen,rojo,verde,azul)` devuelve un identificador del color representado por la mezcla de los componentes RGB (Red, Green, Blue) especificados.

Los colores se consiguen mezclando los tres colores básicos en el grado que necesitemos. La mezcla se obtiene asignando a cada uno de los tres últimos parámetros un valor desde 0 hasta 255. Por ejemplo, el color negro se obtiene con la instrucción

```
$negro= imagecolorallocate(id. imagen, 0, 0, 0);
```

En cambio, el color blanco se obtiene con la instrucción

```
$blanco = ImageColorAllocate(id. imagen,255,255,255);
```

Los tres colores básicos se obtienen así:

```
$rojo = ImageColorAllocate(id. imagen, 255 ,0, 0);
$verde = ImageColorAllocate(id. imagen, 0, 255,0);
$azul = ImageColorAllocate(id. imagen, 0, 0, 255);
```

Los valores numéricos que permiten mezclar los colores pueden expresarse en base decimal, como hemos hecho, o en base hexadecimal. Por ejemplo, también podemos fijar los colores así:

```
$white = imagecolorallocate($imagen, 0xFF, 0xFF, 0xFF);
$navy  = imagecolorallocate($imagen, 0x00, 0x00, 0x80);
$black = imagecolorallocate($imagen, 0x00, 0x00, 0x00);
$gray  = imagecolorallocate($imagen, 0xC0, 0xC0, 0xC0);
```

Para una mayor información sobre los valores que hay que asignar de cada parámetro de esta función a fin de conseguir el color deseado, puede verse la página http://eies.njit.edu/~kevin/rgb.txt.html, donde se muestra un completo mapa de colores con los correspondientes valores.

En el Ejemplo 5 puede estudiarse cómo hemos establecido los colores y cómo los hemos usado, después, en las funciones que los incluyen como parámetros. Prácticamente en todos los ejemplos y ejercicios de esta Unidad se usa esta función.

En el Ejercicio 1 puede verse cómo se fijan también los colores directamente con un número hexadecimal así:

```
$color_inicio='0000FF';
```

donde los dos primeros dígitos de la izquierda (00x, 0d) corresponden al color Rojo, los dos siguientes (00x, 0d) al Verde y los dos últimos (FFx, 255d) al Azul. Puede mirarse en este ejercicio cómo se fijan los colores de una forma algo más profesional.

La función **`imagecolordeallocate`**`(id. imagen, id. color)` elimina de una imagen el color que se le haya asignado con la función `imagecolorallocate()`.

La función **`imagecolortransparent`**`(id. imagen[, id. color])` define un color como transparente para `imagen`. Devuelve el identificador del color transparente. Si no se especifica ningún color, devuelve como transparente el color actual.

La imágenes se pueden **rellenar** total o parcialmente con colores. La función **`imagefill`**`(id. imagen,x,y,color)` permite rellenar una `imagen` empezando desde las coordenadas `x,y` del `color` que se especifique. En casi todos los ejemplos y ejercicios de esta Unidad puede encontrarse esta función.

Complementarias de las funciones que permiten dibujar y rellenar imágenes son las siguientes:

La función **`imagefilledarc`**`(id. imagen,x,y,anchura,altura,inicio,final,color,estilo)` dibuja una elipse parcial (arco) centrada en `x,y` (la esquina superior izquierda es 0,0) según el tamaño indicado en `anchura` y `altura` (ambas referidas a la elipse), desde `inicio` hasta `final` (ambos indicados en grados) y la rellena del `color` especificado en el `estilo` indicado. Todos los parámetros son de tipo numérico entero.

Hay cuatro tipos de estilos:

Valor 1: IMG_ARC_PIE. Rellena con el color indicado el ángulo que va desde el centro `x,y` hasta `inicio` y `final` trazando una línea entre estos dos puntos sin rellenar la zona del arco de forma que se da al arco una apariencia de tarta.

Valor 2: IMG_ARC_CHORD. Une con una línea curva del color indicado el `inicio` y el `final` del ángulo trazado. Los valores 1 y 2 se excluyen.

Valor 3: IMG_ARC_NOFILL. Une con una línea recta del color indicado el `inicio` y el `final` del ángulo trazado.

Valor 4: IMG_ARC_EDGED. Rellena por completo con el color indicado el ángulo que va desde el centro `x,y` hasta el borde del arco de forma que se da al arco una apariencia de tarta.