第 9 章　PHP 与图像操作

PHP 不仅可以生成 HTML 页面，PHP 还可以用来创建和操作二进制文件，例如图片。使用 PHP 程序动态生成图片可以实现生成缩略图、验证码等多种功能，在实际应用中经常被用到。在 PHP 中，通常使用 GD 库来实现对图像的操作。本章将介绍如何使用 GD 库来操作图像文件。

9.1　PHP 图像函数库简介

在 PHP 中，使用 GD 库来对图像进行操作，GD 库可以从官方网站 http://www.boutell.com/gd/上获得更详细的信息。GD 库是一个开放的动态创建图像的源代码公开的函数库。GD 库使用 C 语言开发，可以在 Perl、PHP 等多种程序语言中调用。目前的 GD 库支持 PNG、JPEG、GIF 等多种图像格式。GD 库通常用于创建图片、文字以及对其他的图片进行处理。

GD 库提供了多种图像创建与操作函数，本章将在后面详细介绍。

9.2　GD 库的配置

GD 库在 PHP5 中是被默认安装的，但是要想激活 GD 库的使用，必须修改 PHP 安装路径下的 php.ini 文件使其增加 GD 库的扩展，如下所示。

```
extension=php_gd2.dll
```

在 PHP5 中，GD1 已经不再受支持，而是使用功能更加强大的 GD2。修改好 php.ini 文件以后，重新启动 Apache 服务器后即可使改动生效。

GD 库提供了一个函数 gd_info 来显示 GD 库的安装信息，其语法格式如下所示。

```
array gd_info();
```

该函数返回一个包含有完整 GD 安装信息的数组，具体代码如下所示。

```
<?php
var_dump(gd_info());
?>
```

运行结果如下所示。

```
array(12) {
  ["GD Version"]=>
  string(27) "bundled (2.0.28 compatible)"
  ["FreeType Support"]=>
  bool(true)
  ["FreeType Linkage"]=>
  string(13) "with freetype"
  ["T1Lib Support"]=>
  bool(true)
```

```
  ["GIF Read Support"]=>
  bool(true)
  ["GIF Create Support"]=>
  bool(true)
  ["JPG Support"]=>
  bool(true)
  ["PNG Support"]=>
  bool(true)
  ["WBMP Support"]=>
  bool(true)
  ["XPM Support"]=>
  bool(false)
  ["XBM Support"]=>
  bool(true)
  ["JIS-mapped Japanese Font Support"]=>
  bool(false)
}
```

从上面的运行结果中可以看到当前 GD 库的安装版本信息，以及其所支持的图片、字体格式。

除此之外，使用 phpinfo 函数也可以得到当前 GD 库的安装信息，代码如下所示。

```
<?php
phpinfo();
?>
```

运行结果如图 9-1 所示。

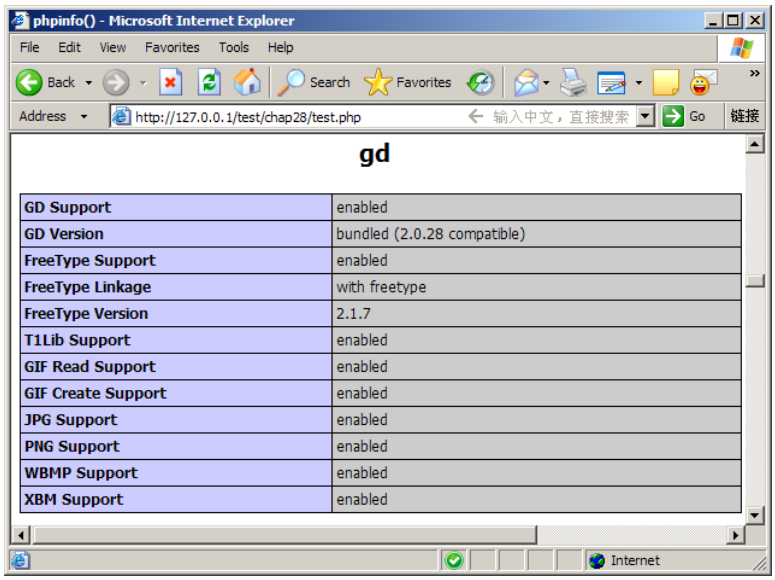

图 9-1　GD 库的安装信息

可以看到上述的运行结果与使用 gd_info 函数所得到的结果相同。

9.3　一个简单图像创建程序

本节将从一个简单的图像创建程序开始介绍一些常用的图像输出函数，下一节将用几个较复杂的例

子来说明 GD 库的强大功能。

在 GD 库配置好以后，就可以使用 PHP 代码来输出图像了。以下代码是输出了一个简单的图像，使用 GD 库提供的函数输出了一个包含文字“Hello World”的 PNG 图片。

```
<?php
header("Content-type: image/png");                          //声明图片格式为 PNG
$image = imagecreatetruecolor(200, 100);                    //创建一个宽 200 像素，长 100 像素的图像
$text_color = imagecolorallocate($image, 255, 255, 255);    //设置当前颜色为白色
imagestring($image, 5, 0, 0, "Hello World!", $text_color);  //输出 Hello World 文字
imagepng($image);                                           //输出图像
imagedestroy($image);                                       //销毁图像对象
?>
```

代码的运行结果如图 9-2 所示。

图 9-2　简单的图像输出例子

代码说明：

- ❑ header("Content-type: image/png")用于声明当前图片的格式。
- ❑ imagecreatetruecolor 函数用于建立一个真彩色的空白图像，其语法格式如下所示。

```
imagecreatetruecolor(int x, int y)
```

其中，x 为图片的宽，y 为图片的高，该函数返回一个图像资源。在前面的例子中，使用了 imagecreatetruecolor(200, 100)代码来新建一个宽为 200，高为 100 的图像。

- ❑ imagecolorallocate 函数用于确定当前的颜色设置，其语法格式如下所示。

```
imagecolorallocate($image, int red, int green, int blue)
```

其中$image 为前面创建的图像对象，red、green 和 blue 为用于确定三种颜色的值。确定这三种颜色可以通过 Windows 自带的画图板中的颜色设置来获得，如图 9-3 所示。

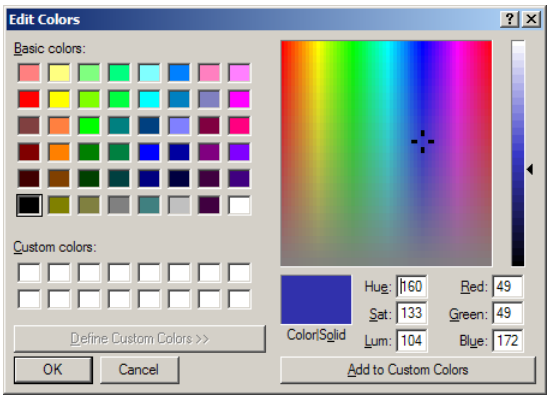

图 9-3　调色板

从图中可以看到，当前选择的颜色的三个值分别为 49、49、172。因此，如果要设置当前的颜色为这种颜色，应使用以下代码来实现。

```
<?php
$text_color = imagecolorallocate($image, 49, 49, 172);
?>
```

- imagestring 函数用于水平输出一行字符串，其语法如下所示。

```
imagestring($image, int font, int x, int y, string str, int color)
```

其中$image 为前面创建的图像对象，font 是所用的字体，x 和 y 是字符串所在图像的坐标，str 是要输出的字符串，color 是前面设置的颜色。

- imagedestroy 函数用于销毁前面创建的图像对象。

9.4　GD 库的应用实例

本节将要通过几个常用的 GD 库应用实例来说明 GD 库的常见应用方法。

9.4.1　使用 GD 库创建图片缩略图

在第 6 章中介绍过文件上传的方法，本小节将实现这样一个实例。首先将图片从本地上传到服务器端，上传后生成一个缩略图并显示在页面上。这个功能在相册程序中非常常用。

上传页面的静态 HTML 代码如下所示。

```
<html>
<head>
<title>文件上传</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<H1>文件上传</H1>
<form enctype="multipart/form-data" action="Upload.php" method="post">
      <input name="upfile" type="file"><BR>
      <input type="submit" value="Submit">
</form>
<body>
</body>
</html>
```

相应的 PHP 代码如下所示。

```
<?php
$uploadfile = "upfiles/".$_FILES['upfile']['name'];                    //上传后文件所在的文件名和路径
$smallfile = "upfiles/small_".$_FILES['upfile']['name'];               //上传后缩略图文件所在的文件名和路
径

if($_FILES['upfile']['type'] != "image/pjpeg")
{
       echo "文件类型错误";                                            //输出错误信息
}
else
{
```

```
    move_uploaded_file($_FILES['upfile']['tmp_name'], $uploadfile);      //上传文件

    $dstW = 200;                                                         //设定缩略图的宽度
    $dstH = 200;                                                         //设定缩略图的高度

    $src_image = ImageCreateFromJPEG($uploadfile);                       //读取 JPEG 文件并创建图像对象
    $srcW = ImageSX($src_image);                                         //获得图像的宽
    $srcH = ImageSY($src_image);                                         //获得图像的高
    $dst_image = ImageCreateTrueColor($dstW,$dstH);                      //创建新的图像对象

    //将图像重定义大小后写入新的图像对象
    ImageCopyResized($dst_image,$src_image,0,0,0,0,$dstW,$dstH,$srcW,$srcH);
    ImageJpeg($dst_image,$smallfile);                                    //创建缩略图文件

    echo "文件上传完成<BR>";                                             //输出上传成功的信息
    echo "<img src='$smallfile'></img>";                                 //在页面上显示缩略图
}
?>
```

在浏览器上运行静态 HTML 页面如图 9-4 所示。

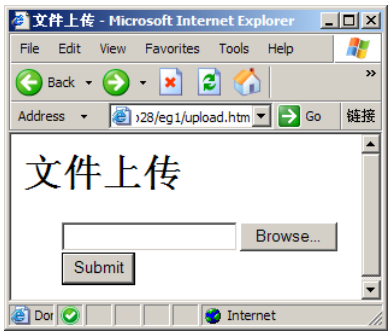

图 9-4　文件上传页面

在本地选择一个图片文件并上传后，页面如图 9-5 所示。

图 9-5　文件上传后的页面

代码说明：

- ImageCreateFromJPEG 是一个用于从图片文件创建图片对象的函数，其语法格式如下所示。

```
ImageCreateFromJPEG($filename)
```

其中$filename 是源文件名。类似的函数还包括以下几个：

imagecreatefromgd2：从 GD2 文件或 URL 新建一图像

imagecreatefromgd：从 GD 文件或 URL 新建一图像

imagecreatefromgif：从 GIF 文件或 URL 新建一图像

imagecreatefromjpeg：从 JPEG 文件或 URL 新建一图像

imagecreatefrompng：从 PNG 文件或 URL 新建一图像

imagecreatefromstring：从字符串中的图像流新建一图像

imagecreatefromwbmp：从 WBMP 文件或 URL 新建一图像

imagecreatefromxbm：从 XBM 文件或 URL 新建一图像

imagecreatefromxpm：从 XPM 文件或 URL 新建一图像

- ImageSX 和 ImageSY 分别用于获得图像的宽与高，其语法格式如下所示。

```
Int ImageSX($image)
Int ImageSY($image)
```

其中，$image 是已经创建好的图像对象。

- ImageCopyResized 函数用于读取源图像的全部或部分并调整大小，其语法格式如下所示。

```
int imagecopyresized(dst_im, src_im, int dstX, int dstY, int srcX, int srcY, int dstW, int dstH, int srcW, int srcH)
```

其中，dst_im 是目标图像对象，src_im 是源图像对象，dstX 和 dstY 分别表示目标图像所在坐标，srcX 和 srcY 分别表示源图像的坐标，dstW 和 dstH 分别表示目标图像的宽和高，srcW 和 srcH 分别表示源图像的宽和高。

- ImageJpeg 函数用于创建一个新的 JPEG 图像。

9.4.2　使用 GD 库生成验证码

本节将介绍如何使用 GD 库生成验证码。验证码是一些网站在接受用户表单时限制用户尝试次数的一种方式。用户在提交表单时不得不反复输入每次都不相同的验证码。

验证码的工作原理往往是在生成图片时将验证的文字保存在 Session 中，然后在接受用户表单时将用户的提交内容与 Session 中的值进行比较以判断用户是否输入了正确的验证码。

用于生成验证码图片的 PHP 代码如下所示。

```
<?
session_start();
header("Content-type: image/png");

$img_width=100;                                          //设置验证码图片的大小
$img_height=20;

srand(microtime() * 100000);                             //设置随机数种子
for($i=0; $i<4; $i++)                                    //循环产生四位验证码
{
      $new_number.=dechex(rand(0,15));
}

$_SESSION[check_auth]=$new_number;                       //将验证码写入 Session
$number_img=imageCreate($img_width, $img_height);        //创建图片对象
ImageColorAllocate($number_img, 255, 255, 255);          //设置背景颜色为白色
```

```
for($i=0;$i<strlen($_SESSION[check_auth]);$i++)                    //循环输出四位验证码
{
    $font = mt_rand(3,5);                                          //设定随机字体
    $x = mt_rand(1,8) + $img_width*$i/4;                           //设定字符所在位置 X 坐标
    $y = mt_rand(1,$img_height/4);                                 //设定字符所在位置 Y 坐标
    //设定字符颜色
    $color = imageColorAllocate($number_img,mt_rand(0,100),mt_rand(0,150),mt_rand(0,200));
    imageString($number_img, $font, $x, $y, $_SESSION[check_auth][$i], $color);          //输出字符
}

ImagePng($number_img);                                             //以 PNG 格式输出
ImageDestroy($number_img);                                         //销毁图像对象
?>
```

从上面的代码可以看到，在进行验证码输出时，每个字符所在位置、颜色、字体都是使用随机数来实现的。这样做的好处是在浏览器上生成多种多样的验证码，以防止恶意用户通过识别技术等工具进行密码破获等行为。

表单的静态 HTML 页面如下所示。

```
<html>
<head>
<title>Form</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<H1>请输入验证码</H1>
<form action="form.php" method="post">
    <input name="auth" type="text" size="4" maxlength="4">
    <img src="auth.php" border=0>
    <input type="submit" value="Submit">
</form>
<body>
</body>
</html>
```

可以看到，这里将 auth.php 作为一个图片来输出。用于验证用户输入的 PHP 页面代码如下所示。

```
<?php
session_start();
$auth = $_POST["auth"];                                            //获取输入的验证码

if($auth=="")                                                      //检查是否为空
{
    echo "错误：验证码为空";
}

if($auth==$_SESSION[check_auth])                                   //检查是否输入正确
{
    echo "正确：验证码输入正确";
}
else
{
    echo "错误：验证码输入不正确";
```

```
}
?>
```

在浏览器中运行表单页面，结果如图 9-6 所示。

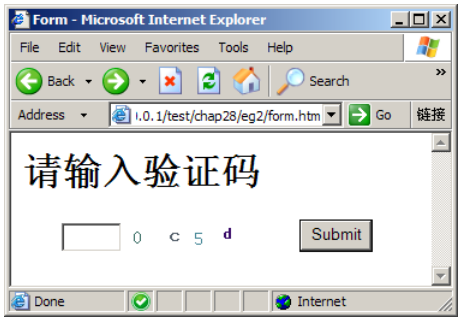

图 9-6　验证码表单

在文本框中输入“0c5d”后单击 Submit 按钮，可以看到浏览器上输出了如下文字。

```
正确：验证码输入正确
```

9.4.3　使用 GD 库下载远程图片

在网络上存放着大量的图片资源，在一个静态网页上很容易可以通过<img>标签来在页面上显示一个远程图片，如下所示。

```
<img src=”http://www.example.com/image.jpg”></img>
```

但是，由于网络的不稳定性，存放在其他的服务器上的图片可能会因为服务器问题、网站改版等多种原因不可访问。这时，原本正常的网页上的图片就会变成一个不可用标志，如图 9-7 所示。

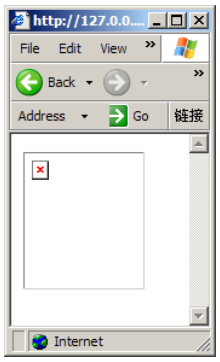

图 9-7　不可用图片

为了解决这个问题，往往将图片下载到本地服务器上，本节将介绍如何使用 GD 库来实现图片文件的下载。以下代码实现了图片下载的功能。

```
<?php
$imgname = "http://127.0.0.1/test/chap28/eg1/upfiles/small_IMG_0075.JPG";
$src_im = imagecreatefromjpeg($imgname);
$srcW = ImageSX($src_im);                                    //获得图像的宽
$srcH = ImageSY($src_im);                                    //获得图像的高

$dst_im = ImageCreateTrueColor($srcW,$srcH);                 //创建新的图像对象

imagecopy($dst_im, $src_im, 0, 0, 0, 0, $srcW, $srcH);
```

```
imagejpeg($dst_im, "newpic.jpg");                                    //创建缩略图文件

echo "<img src='newpic.jpg'></img>";
?>
```

可以看出，实现图片下载的功能与前面生成缩略图的功能几乎相同。不同的只是在打开源图片文件时的文件路径为一个 URL 链接。

9.4.4　使用 GD 库为页面增加水印

很多网站都希望将自己的原创信息只保存在自己的网站上，或者在相应的位置增加版权信息。本小节将实现将文章系统数据库中的数据以图片的形式显示在网页上，并且在相应的位置增加水印以保证版权信息的显示。

首先，需要使用图形编辑工具创建一个透明的图片，该图片需要保证足够透明。也就是说，如果将该图片放置在页面上，将不会对图片下边的内容有阻挡效果，如图 9-8 所示。

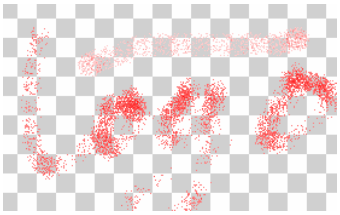

图 9-8　水印图片

具体实现的 PHP 代码如下所示。

```
<?php
header ("Content-type: image/png");
$conn = mysql_connect("localhost", "root", "");                      //连接数据库
$colname_rs_article = $_GET['id'];                                   //获取参数 id

mysql_select_db("cms", $conn);                                       //执行 SQL
$query_rs_article = sprintf("SELECT * FROM articles WHERE article_id = %s", $colname_rs_article);
$rs_article = mysql_query($query_rs_article, $conn) or die(mysql_error());
$row_rs_article = mysql_fetch_assoc($rs_article);
$totalRows_rs_article = mysql_num_rows($rs_article);

$image = ImageCreateTrueColor(700, 1000);                            //创建画布
$bg = ImageColorAllocate($image, 255, 255, 255);                     //设置背景为白色
ImageFill($image, 0, 0, $bg);
$text_color = ImageColorAllocate($image, 0, 0, 0);                   //设置文字颜色为黑色
imagestring($image, 5, 0, 0, $row_rs_article['title'], $text_color);  //输出文章标题
imagestring($image, 3, 0, 20, $row_rs_article['author'], $text_color); //输出文章作者
imagestring($image, 4, 0, 60, $row_rs_article['content'], $text_color); //输出文章内容
$logo = ImageCreateFromPNG('logo.png');                              //获得水印图片
$logoW = ImageSX($logo);
$logoH = ImageSY($logo);
ImageCopy($image, $logo, 0, 0, 0, 0, $logoW, $logoH);                //合并文字图片与水印图片
ImageJPEG($image); // output to browser
ImageDestroy($logo);
```

```
ImageDestroy($image);
?>
```

运行效果如图 9-9 所示。

图 9-9　使用图片显示文章

使用图片显示数据库中的数据一方面可以防止网站内容被非法盗用，也可以有效的防止被非法篡改。但是，由于图片的字节数远远大于文章本身，使用这种方法难免会导致页面打开速度过慢等问题。

9.5　小结

本章介绍了在 PHP 中创建图像的最常用方法——使用 GD 库来创建图像。读者可以通过仔细研究 9.4 节中的实际应用实例来掌握 GD 库中常见应用的设计方法。

下一章将详细介绍 PHP 中的另一个用于实现图像操作的库 Jpgraph。与 GD 不同，Jpgraph 将主要用于统计图的创建。