

GD 图像处理

GD 图像处理基本技术

GD 库引入与介绍

API：外部提供的接口：已经准备好了一套处理某些功能的机制，用户只需要按照指定的数据要求，调用指定的函数或者方法（类）就可以实现某个功能。

1) GD 库的概念：Graphic Device，图像处理扩展（外部提供的 API），能够允许 PHP 在脚本中使用对应的函数来实现某些图像制作功能

2) GD 库的引入：GD 库是外部提供的 API，已经被集成到 PHP 扩展库中（不需要下载），但是需要在 PHP 的配置文件中开启对应的扩展。GD 扩展（GD2）。

```
372 ;extension=php_fileinfo.dll
373 extension=php_gd2.dll
374 ;extension=php_gettext.dll
```

注意：记得重启 Apache 服务器

GD 图像处理基本技术

画图的基本流程：画图本质是在内存开辟一块很大的内存区域用于图片制作

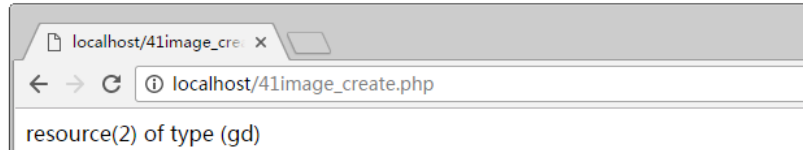
- 1、准备画布
- 2、开始作画
- 3、保存内容
- 4、销毁画布

创建画布资源

1) ImageCreate(宽, 高)：创建一个空白画布（背景色是白色的）

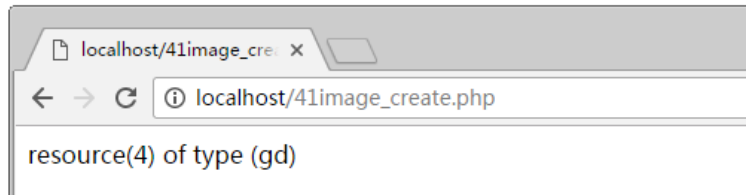
2) ImageCreateTrueColor(宽, 高)：创建一个真彩画布（背景色是黑色的，需要填充）

```
//创建一个真彩图片
$img = imagecreatetruecolor(100,100) or die('图片初始化失败!');
var_dump($img);
```



3) ImageCreatefromJpeg(图片文件路径): 打开一个 jpeg 格式的图片资源

```
//从以后jpeg图片中打开资源
$img = imagecreatefromjpeg('uploads/Koala.jpg');
var_dump($img);
```

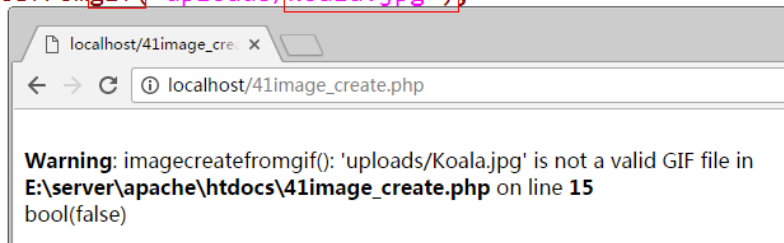


4) ImageCreatefromGif(图片文件路径): 打开个 gif 格式图片资源 (PHP 中无法实现动态)

5) ImageCreatefromPng(图片文件路径): 打开 png 格式图片资源

如果从已知文件创建图片资源, 那么一定要匹配对应的打开方式, 否则会出错

```
//错误尝试
$img = imagecreatefromgif('uploads/Koala.jpg');
var_dump($img);
```



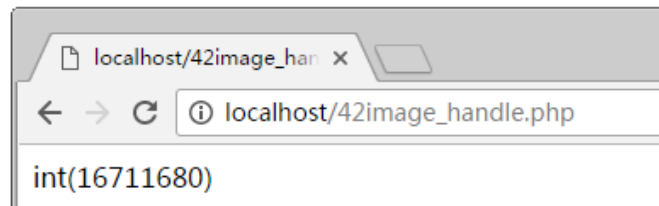
GD 图像处理基本技术

操作画布资源

说明: 所有的画布资源操作都是需要指定画布资源, 而且都是第一个参数。

1) 分配颜色: imageColorAllocate(画布资源, 红色, 绿色, 蓝色), 根据 RGB 三色组给指定画布资源分配一组颜色, 会返回一个颜色句柄 (一组整数), 每一个色组都是从 0 到 255

```
//分配颜色
$color = imagecolorallocate($img,255,0,0);
var_dump($color);
```



在真彩图片资源中，所有分配的颜色都不会自动给图片资源上色，是用来后续操作图片资源的时候，指定着色的：但是如果当前使用的 `imagecreate` 创建的图片资源，那么第一个分配的颜色，会自动被着色为图片背景色。

注意：凡是给图片上增加内容，基本都需要分配颜色（每一个操作图片的函数之前，都需要先调用分配颜色的函数得到一个颜色）

2) 填充区域: `imageFill`(画布资源, 起始 X 坐标, 起始 Y 坐标, 颜色句柄): 指定位置开始填充指定颜色

```
//填充背景色
imagefill($img,0,0,$bg_color);
```

`Imagefill` 的填充逻辑：从指定点开始，自动匹配相邻点，如果颜色一致，自动渲染，扩展到全图。

3) 画直线: `imageLine`(图片资源, 起始点 X, 起始点 Y, 终点 X, 终点 Y, 颜色)，制作一条直线

```
.4
.5 //制作直线
.6 $line_color = imagecolorallocate($img,0,255,0);
.7 imageline($img,10,10,90,90,$line_color);
```

4) 画矩形: `imageRectangle`(图片资源, 左上角 X, 左上角 Y, 右下角 X, 右下角 Y, 颜色): 制作一个矩形

```
.8
.9 //制作矩形
!0 $rec_color = imagecolorallocate($img,0,0,255);
!1 imagerectangle($img,10,10,90,90,$rec_color);
```

5) 画圆弧: `imageArc`(图像资源, 轴点 X, 轴点 Y, 宽度, 高度, 弧度起点, 弧度终点, 颜色): 制作弧度图像

```
//制作一段圆弧（圆）
$arc_color = imagecolorallocate($img,0,0,0);
imagearc($img,50,50,80,80,180,360,$arc_color);
```

6) 在画布上写字: imageString(), imageTtfText()

Imagestring(图片资源, 文字大小, 起始 X, 起始 Y, 内容, 颜色): 用来书写 ASCII 对应的字符 (英文)

```
//写入英文字符串
$str_color = imagecolorallocate($img,100,100,100);
imagestring($img,5,20,20,'abcd',$str_color);
```

Imagettfttext(图片资源, 文字大小, 旋转角度, 起始 X, 起始 Y, 颜色, 字体, 内容): 可以书写任意文字 (中文): 需要指定字体路径 (ttf 文件: 默认是 windows/fonts/)

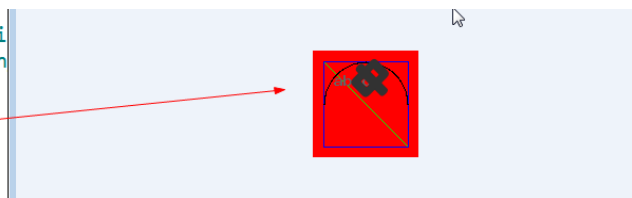
```
//写入中文汉字
$ch_color = imagecolorallocate($img,50,50,50);
imagettftext($img,30,45,50,50,$ch_color,'STHUP0.TTF','中');
```

GD 图像处理基本技术

输出画布资源

1) 输出为图片文件: 以图片文件形式保存到本地文件夹

```
//保存输出
imagepng($img,'my.png');
//存成原来类型
imagepng($img);
```



2) 输出为网页图片: 将图片展示给 HTML (用户): 服务器需要告知服务器当前内容是图片 (修改响应头)

```
//输出给浏览器必须告知响应头: 图
header('Content-type:image/png')
imagepng($img);
```



Image+图片格式

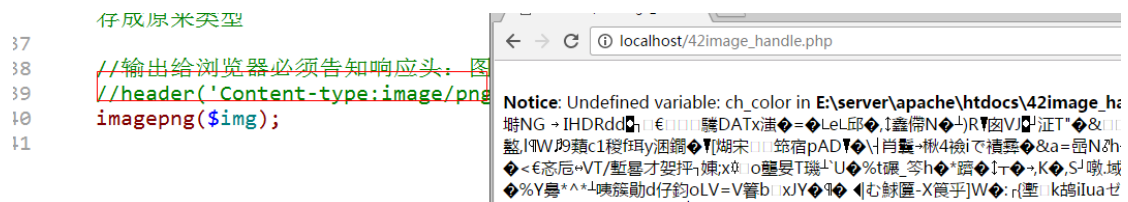
Imagejpeg(): 保存成 jpeg 格式图片

Imagepng(): 保存成 png 格式图片

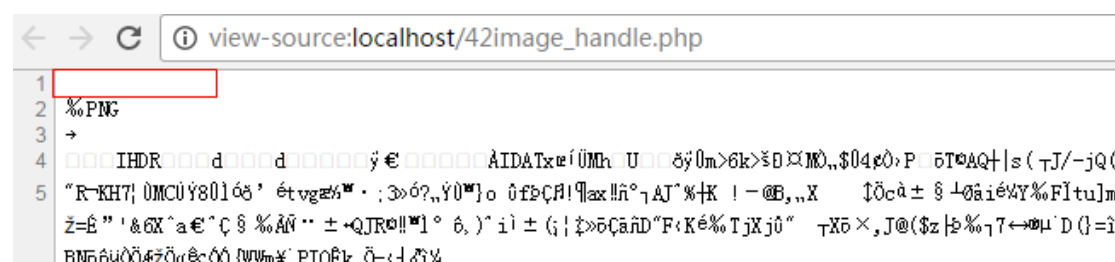
Imagegif(): 保存 gif 格式图片

如果图片只是提供了图片资源, 不指定保存的文件位置, 那么系统认为是输出给浏览器; 如果指定了位置, 那么系统认为是保存到本地 (第二个参数);

细节 1: 如果图片输出或者保存出错, 浏览器看到的永远是告诉你图片错了, 但是绝对不会告知错误原因在哪, 需要关闭 header 图片输出, 再看问题



细节 2: 如果图片输出之后没有成功, 但是关闭 header 也看不到错误: 最大的可能是图片输出之前输出的别的额外的内容 (喜欢输出 pre), 应该查看网页原码, 看看图片输出之前是否有任何输出: 尤其是空格空行



销毁画布资源

从内存中将画布资源清理掉, 释放内存。

Imagedestroy(画布资源):

```

41
42     //销毁资源
43     imagedestroy($img);

```

GD 图像处理基本技术

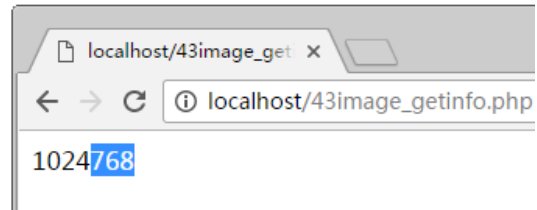
获取图片信息

1) 获取画布尺寸: imagesx(), imagesy()

```
//定义图片
$image = 'uploads/Koala.jpg';

//打开图片资源
$img = imagecreatefromjpeg($image);

//获取图片信息
echo imagesx($img);
echo imagesy($img);
```

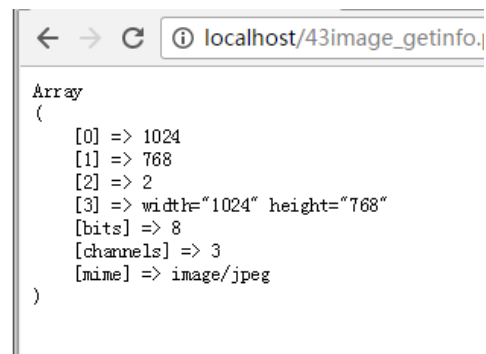


2) 获取图片尺寸: getimagesize()

```
//echo imagesy($img);

//获取图片全部信息
echo '<pre>';
$info = getimagesize($image);

print_r($info);
```



GD 图像处理应用案例

验证码的实现

验证码 (CAPTCHA) 是 “Completely Automated Public Turing test to tell Computers and Humans Apart” (全自动区分计算机和人类的图灵测试) 的缩写, 是一种区分用户是计算机还是人的公共全自动程序。可以防止: 恶意破解密码、刷票、论坛灌水, 有效防止某个黑客对某一个特定注册用户用特定程序暴力破解方式进行不断的登陆尝试, 实际上用验证码是现在很多网站通行的方式, 我们利用比较简易的方式实现了这个功能。这个问题可以由计算机生成并评判, 但是必须只有人类才能解答。由于计算机无法解答 CAPTCHA 的问题, 所以回答出问题的用户就可以被认为是人类。

图片验证码: 计算机将拿到的验证码存放到图片中, 然后用户看到然后识别, 然后提交给服务器, 服务器再根据用户提交的和服务器之前生成的进行比较。

1) 实现验证码图片的展示

a. 生成图片资源: 背景色设定

```
//创建图片资源
$img = imagecreatetruecolor(200,50);
```

背景色处理

```
3
4 //背景色
5 $bg_color = imagecolorallocate($img,220,220,220);
6 imagefill($img,0,0,$bg_color);
7
```

b. 写入文字: imagestring 写的效果一般, 所以一般可以使用 imagettftext

```
12
13 //写入内容
14 $font = 'STHUPO.TTF';
15 $str_color = imagecolorallocate($img,100,100,100);
16 imagettftext($img,30,15,60,40,$str_color,$font,'中');
17 imagettftext($img,30,-15,140,40,$str_color,$font,'国');
18 |
```

c. 输出图片给浏览器

```
19
20 //输出资源
21 header('Content-type:image/png');
22 imagepng($img); //png格式图片是透明的, 比较小
23
```

d. 关闭资源

```
3 //销毁资源
4 imagedestroy($img);
```

2) 实现验证码文字的随机变化: 有一串文字可以随机选择

a. 制作目标字符串集: 从哪里选内容

```
2
3 //获得随机文字
4 $str = '我家住在黄土高坡大风从坡上刮过无论是东南风还是西北风都是我的歌';
5
```

b. 如何随机从字符串中取出对应的汉字: 汉字在 utf-8 字符集中一个字占用 3 个字节, 英文字母只占一个字节: 确定字符数

```
16 //获取字符串长度
17 $len = strlen($str); //字节长度
18 //当前所有的内容都是中文, 因此每个字必定占三个字节
19 $c_len = $len / 3; //字符数量 (UTF-8)
20 |
```

c. 随机取出对应的字符 (中文汉字)

```
21
22 //随机取
23 $char1 = substr($str,mt_rand(0,$c_len - 1) * 3,3);
24 $char2 = substr($str,mt_rand(0,$c_len - 1) * 3,3);
25
26 |
```

d. 将取到的字符放到指定的图片位置即可

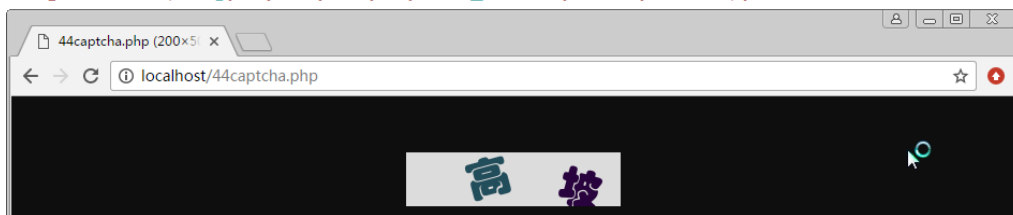
```
//写入内容
$font = 'STHUPO.TTF';
$str_color = imagecolorallocate($img,100,100,100);
imaggittfttext($img,30,15,60,40,$str_color,$font,$char1);
imaggittfttext($img,30,-15,140,40,$str_color,$font,$char2);

//输出资源
```



3) 实现验证码文字颜色的随机变化：每次产生不同的文字颜色

```
//写入内容
$font = 'STHUPO.TTF';
$str_color1 = imagecolorallocate($img,mt_rand(0,100),mt_rand(0,100),mt_rand(0,100));
$str_color2 = imagecolorallocate($img,mt_rand(0,100),mt_rand(0,100),mt_rand(0,100));
imaggittfttext($img,30,15,60,40,$str_color1,$font,$char1);
imaggittfttext($img,30,-15,140,40,$str_color2,$font,$char2);
```



4) 实现验证码背景或干扰噪点：增加一些额外的不影响用户看但是会产生模糊效果的内容（点或者直线）：imagestring/imageline

a. 增加干扰点

```
//干扰：点和线
//干扰点
for($i = 0;$i < 50;$i++){
    //点：写字符
    $dots_color =
        imagecolorallocate($img,mt_rand(150,250),mt_rand(150,250),mt_rand(150,250));
    imagestring($img,mt_rand(1,5),mt_rand(0,200),mt_rand(0,50),'*',$dots_color);
}
```

b. 增加干扰线

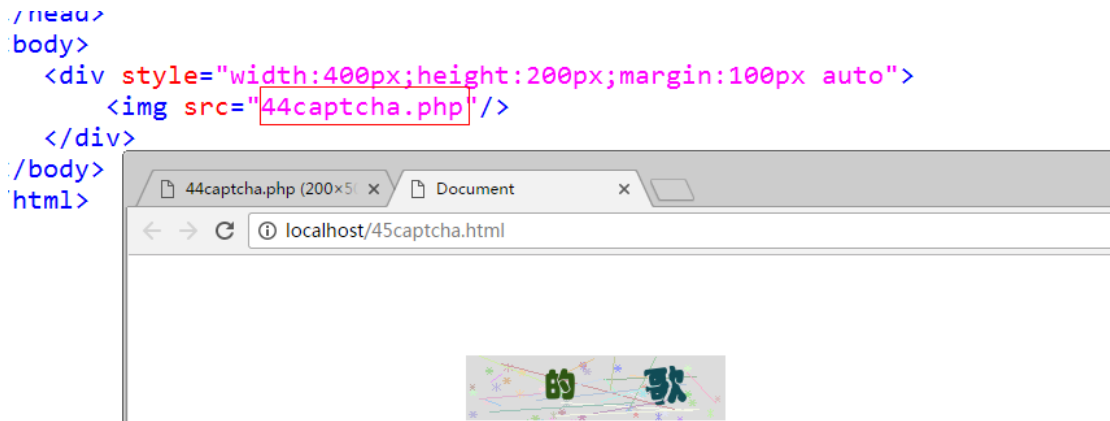
```
//干扰线
for($i = 0;$i < 20;$i++){
    //线：画直线
    $line_color =
        imagecolorallocate($img,mt_rand(150,250),mt_rand(150,250),mt_rand(150,250));
    imageline($img,mt_rand(0,200),mt_rand(0,50),mt_rand(0,200),mt_rand(0,50),$line_color);
}
```

c. 改变文字的大小和位置以及其他可变的信息

```
//写入内容
$font = 'STHUPO.TTF';
$str_color1 = imagecolorallocate($img,mt_rand(0,100),mt_rand(0,100),mt_rand(0,100));
$str_color2 = imagecolorallocate($img,mt_rand(0,100),mt_rand(0,100),mt_rand(0,100));
imaggittfttext($img,mt_rand(10,25),mt_rand(-15,15),60,mt_rand(30,40),$str_color1,$font,$char1);
imaggittfttext($img,mt_rand(10,25),mt_rand(-15,15),140,mt_rand(30,40),$str_color2,$font,$char2);
```

5) 实现点击刷新验证码功能：实现验证码在浏览器显示的功能

a. 创建一个表单文件，里面有一个 img 标签能够显示图片：宽 200，高 50



b. 实现点击更换验证码：让 src 重新请求 PHP 脚本，产生一张新的图片。因此需要给 img 标签增加一个点击事件：img 的 src 是否重新发起请求，取决于浏览器；浏览器是否重新发起请求取决于 src 是否改变。

```

<div style="width:400px;height:200px;margin:100px auto">
  
</div>
</body>
</html>

```

c. 如何让 img 标签的 src 每次点击都不一样？就可以解决问题

```

</head>
<body>
  <div style="width:400px;height:200px;margin:100px auto">
    
  </div>
</body>
</html>

```

GD 图像处理应用案例

缩略图的实现

1) 制作图片缩略图的原理

缩略图：将原图得到一个较小的图（尺寸上）

缩略图原理：将原图打开，然后放到另外一个较小的图片资源中，最后进行保存即可。

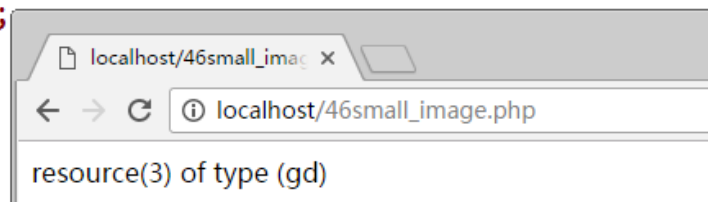
2) 实现固定宽高的缩略图

a. 得到一张原图资源

```

5 //得到原图资源
5 $src_image = 'uploads/koala.jpg';
7
3 $src = imagecreatefromjpeg($src_image);
3 var_dump($src);

```



b. 得到一个缩略图资源（较小）

```

0
1 //制作缩略图资源
2 $dst = imagecreatetruecolor(100,100);|

```

c. 图片采样复制: GD 提供了一个函数

imagecopyresampled(缩略图资源, 原图资源, 缩略图开始放 X, 放 Y, 原图采样其实 X, 起始 Y, 缩略图存放宽, 存放高, 原图采样宽, 采样高)

```

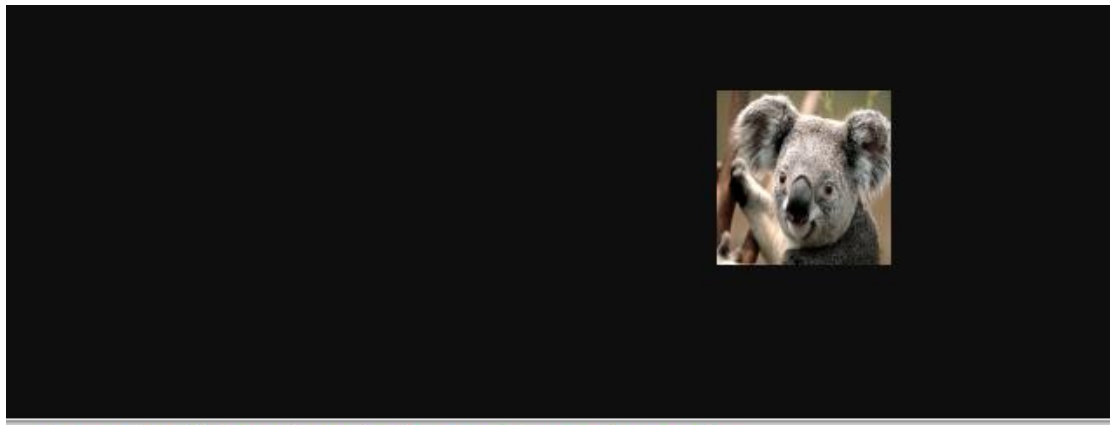
//采样复制
$res = imagecopyresampled($dst,$src,0,0,0,0,100,100,imagesx($src),imagesy($src));
var_dump($res);

```



```
;
```

d. 保存缩略图



```

header('Content-type:image/png');
imagepng($dst);

```

保存到本地

```

//保存
header('Content-type:image/png');
imagepng($dst,'uploads/thumb_koala.png');

```

```
↑
```

e. 销毁所有资源 (原图和缩略图)

```

22 //销毁资源
23 imagedestroy($src);
24 imagedestroy($dst);|

```

```
↑
```

3) 实现等比例缩放的固定宽或高的缩略图

优点: 图片不会失真 (变形)

缺点：缩略图有些部分需要进行额外填充（白色填充：补白）

```
4 //制作缩略图资源
5 $dst = imagecreatetruecolor(100,100);
6 //填充背景色
7 $dst_color = imagecolorallocate($dst,255,255,255);
8 imagefill($dst,0,0,$dst_color);
9
```

等比例缩略图与固定缩略图的制作区别：在于需要通过计算来得出缩略图的宽和高
算法原理：

- 1、 计算缩略图宽高比
- 2、 计算原图宽高比
- 3、 比较：
 - a) 如果缩略图宽高比大于原图宽高比，将缩略图中用原图的高尽可能填满：缩略图的高是完整的，宽度不够（补白）
 - b) 如果缩略图宽高比小于原图宽高比，将缩略图中用原图的宽尽可能填满：缩略图的宽是完整的，高度不够（补白）
- 4、 将图片放到缩略图中间

```
19
20 //计算缩略图从原图采样的宽和高（缩略图到底是宽还是高被填满）
21 $thumb_b = 100 / 100;
22 $src_b = $src_info[0] / $src_info[1];
23
24 //缩略图宽和高
25 $thumb_x = $thumb_y = 0; //宽高
26 $start_x = $start_y = 0; //其实位置
27
28 //比较
29 if($thumb_b >= $src_b){
30     //缩略图宽高比比原图宽高比大
31     $thumb_y = 100;
32     $thumb_x = floor($src_b * $thumb_y);
33     //宽度要调整
34     $start_x = floor((100 - $thumb_x)/2);
35 }else{
36     //缩略图宽高比比原图宽高比小
37     $thumb_x = 100;
38     $thumb_y = floor($thumb_x / $src_b);
39     $start_y = floor((100 - $thumb_y)/2);
40 }
41
```

计算缩略图宽和高

计算偏移位置

在采样过程中，将得到的数据进行采样复制

```
2 //采样复制
3 $res =
4 imagecopyresampled($dst,$src,$start_x,$start_y,0,0,$thumb_x,$thumb_y,$src_info[0],$src_info[1]);
5 //var_dump($res);
6
```

复制：按照等比例复制

采样：整个原图采样

GD 图像处理应用案例

水印图的实现

水印图：watermark，在某个图片上增加一个透明的印记（马赛克）

水印图用途：版权操作

1) 制作图片水印图的原理

水印图制作原理：将一个带有明显标志的图片放到另外一张需要处理的图片之上

- 获取原图资源（需要放上水印图）
- 获取水印图资源
- 合并图片（把水印图合到目标图上）
- 保存输出
- 清除资源

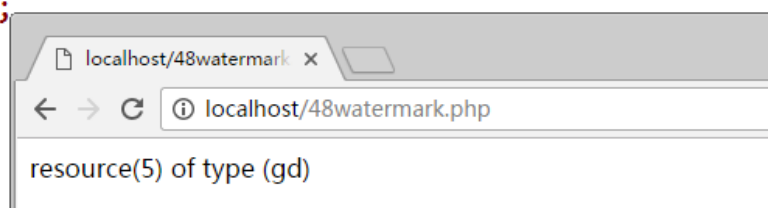
2) 实现固定位置的水印图：左上角

- 获取原图资源（需要放上水印图）

```
$src_image = 'uploads/Koala.jpg';  
$wat_image = 'watermark.png';  
  
//获取原图资源  
$dst = imagecreatefromjpeg($src_image);
```

- 获取水印图资源

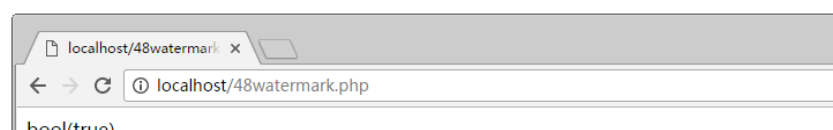
```
//获取水印图资源  
$wat = imagecreatefrompng($wat_image);  
var_dump($wat);
```



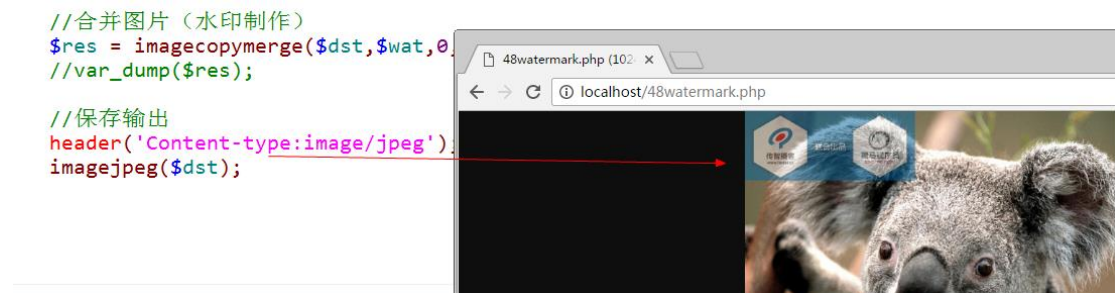
- 合并图片（把水印图合到目标图上）

imagecopymerge(目标资源, 水印资源, 目标起始X, 起始Y, 水印起始X, 起始Y, 目标宽, 目标高, 透明度)

```
//合并图片（水印制作）  
$res = imagecopymerge($dst,$wat,0,0,0,0,imagesx($wat),imagesy($wat),50);  
var_dump($res);
```



- 保存输出



e. 清除资源

```

3
4      //清除资源
5      imagedestroy($dst);
6      imagedestroy($wat);

```

3) 实现可选 9 个位置的水印图：封装制作水印图的函数

a. 创建一个制作水印图的函数结构：制作水印图需要提供多少条件？原图资源，水印图资源，位置选择？9 个，透明度（保存位置）

```

/*
 * 制作水印图
 * @param1 string $src_image, 原图路径
 * @param2 string $wat_image, 水印图路径
 * @param3 string $path, 水印图存储路径、
 * @param4 int $position = 1, 水印加载位置：1代表左上角以此类推9代表右下角
 * @param5 int $pct = 20, 透明度，默认20
 */
function watermark($src_image,$wat_image,$path,$position = 1,$pct = 20){
}

```

b. 结果是希望产生水印图：但是可能产生成功，返回文件保存名字；如果产生失败，应该返回 false，但是还需要告知外界原因：通过引用传参解决

```

5      /*
6      * 制作水印图
7      * @param1 string $src_image, 原图路径
8      * @param2 string $wat_image, 水印图路径
9      * @param3 string $path, 水印图存储路径
10     * @param4 string &$amp;error,记录错误信息的变量
11     * @param5 int $position = 1, 水印加载位置：1代表左上角以此类推9代表右下角
12     * @param6 int $pct = 20, 透明度，默认20
13     */
14     function watermark($src_image,$wat_image,$path,&$error,$position = 1,$pct = 20){
15     }
16

```

C. 水印图前提：原图和对应的水印图都存在

```

//验证原图和水印图都存在
if(!is_file($src_image)){
    $error = '原图不存在!';
    return false;
}

if(!is_file($wat_image)){
    $error = '水印图不存在!';
    return false;
}
}

```

d. 判定保存路径是否存在

```

//判定路径保存是否存在
if(!is_dir($path)){
    $error = '保存位置不正确!';
    return false;
}

```

e. 打开原图和水印图资源

1) 想办法确定要什么函数来打开图片资源：需要通过图片 MIME 类型类确定，获取图片信息

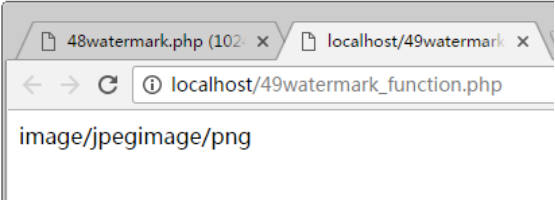
```

//获取图片信息
$src_info = getimagesize($src_image);
$wat_info = getimagesize($wat_image);

echo $src_info['mime'];
echo $wat_info['mime'];

//打开原图资源
//imagecreatefrom

```



2) 通过 MIME 类型得到要打开图片的函数：先设定一个数组进行匹配，匹配成功自动构造创建函数（保存函数），失败则提示错误

```

//定义一组数据：用来产生对应图片
$allow = array(
    'image/jpeg' => 'jpeg',
    'image/jpg' => 'jpeg',
    'image/gif' => 'gif',
    'image/png' => 'png',
    'image/pjpeg' => 'jpeg'
);

```

3) 匹配数据

```

46 //匹配数据
47 if(!array_key_exists($src_info['mime'],$allow)){
48     $error = '当前文件资源不允许制作水印图!';
49     return false;
50 }
51 if(!array_key_exists($wat_info['mime'],$allow)){
52     $error = '当前水印图不允许做资源使用!';
53     return false;
54 }

```

- 4) 组合函数名字: 打开原图资源函数, 打开水印图函数, 保存水印完成图函数

```

//组合函数
$src_open = 'imagecreatefrom' . $allow[$src_info['mime']];
$wat_open = 'imagecreatefrom' . $allow[$wat_info['mime']];
$src_save = 'image' . $allow[$src_info['mime']];

echo $src_open,$wat_open

```

```

//打开原图资源

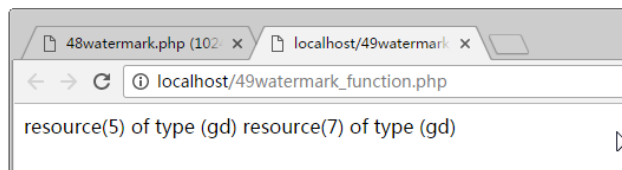
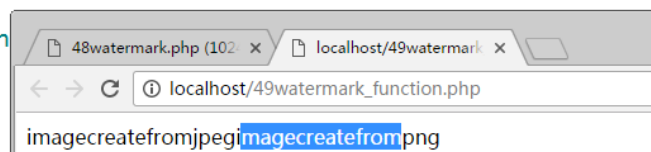
```

- 5) 打开图片资源

```

//echo $src_open,$wat_open;
//打开原图资源
$src = $src_open($src_image);
$wat = $wat_open($wat_image);
var_dump($src,$wat);
}

```



- f. 合并图片资源: 产生水印: 位置需要计算

- 1) 计算水印图在原图中的位置

```

67 //计算水印图在原图中出现的位置
68 $start_x = $start_y = 0;
69 switch($position){
70     case 1:
71         break; //左上角
72     case 2: //上中
73         $start_x = floor(($src_info[0] - $wat_info[0]) / 2);
74         break;
75     case 3: //右上
76         $start_x = $src_info[0] - $wat_info[0];
77         break;
78     case 4: //中左
79         $start_y = floor(($src_info[1] - $wat_info[1]) / 2);
80         break;
81     case 5: //中间
82         $start_x = floor(($src_info[0] - $wat_info[0]) / 2);
83         $start_y = floor(($src_info[1] - $wat_info[1]) / 2);
84         break;
85     case 6: //中右
86         $start_x = $src_info[0] - $wat_info[0];
87         $start_y = floor(($src_info[1] - $wat_info[1]) / 2);
88         break;

```

- 2) 合并图片资源

```

-
//合并图片资源
if(imagecopymerge($src,$wat,$start_x,$start_y,0,0,$wat_info[0],$wat_info[1],$pct)){
    //成功
}else{
    //失败
    $error = '水印图制作（合并）失败！';
    return false;
}
}

```

g. 保存水印图片和销毁资源

```

-
//合并图片资源
if(imagecopymerge($src,$wat,$start_x,$start_y,0,0,$wat_info[0],$wat_info[1],$pct)){
    //保存图片
    //header('Content-type:' . $src_info['mime']);
    $filename = 'watermark_' . trim(strrchr($src_image , '/'), '/');
    $src_save($src,$path . '/' . $filename);

    //销毁资源
    imagedestroy($src);
    imagedestroy($wat);
    return $filename;
}else{
    //失败
    $error = '水印图制作（合并）失败！';
    return false;
}
}

```

h. 测试：函数调用的时候，可以通过结果判定操作是否成功

```

121 //调用函数
122 $res = watermark('uploads/Koala.jpg','watermark.png',__DIR__,$error,5);
123 if($res){
124     echo $res;
125 }else{
126     echo $error;
127 }

```