

Лекция 10

Создание и работа с
изображениями в РНР

Общая информация

Язык PHP достаточно разнообразен и его использование не ограничивается только созданием скриптов, выполняющим различные манипуляции с данными. Он также позволяет работать с различными графическими элементами, в частности с изображениями разных форматов. Для этих целей написана специальная библиотека под названием GD, которая расширяет его стандартный функционал.

Для этого необходимо иметь установленную библиотеку функций изображений GD. GD и PHP может также требовать подключение других библиотек, в зависимости от того, с какими форматами изображений предполагается работать.

Библиотека GD

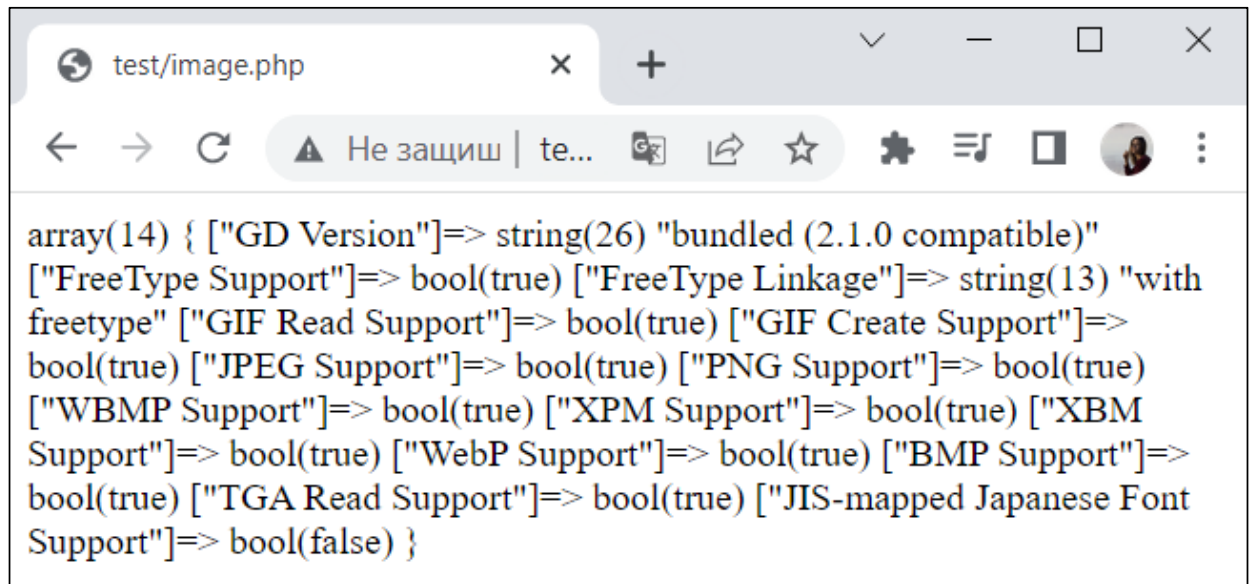
Библиотека gd сама по себе не является графической программой или программой рисования, а также не представляет собой автономное приложение или графический интерфейс пользователя. Вместо этого библиотека gd предоставляет функции, которые могут быть вызваны на выполнение в любой программе для осуществления требуемых манипуляций с изображениями.

Библиотека gd позволяет вызывать функции для создания исходных изображений (первоначально пустых, напоминающих чистый лист бумаги), чертить и рисовать внутри этих исходных изображений с применением различных способов, и в конечном итоге преобразовывать изображение из внутреннего формата изображения gd в стандартный формат изображения, а затем отправлять в окончательное место назначения (выводить в окно браузера либо сохранять в файле или в базе данных).

Проверка версии установленной графической библиотеки GD

Инсталляция библиотеки gd должна начинаться с нулевого этапа — с проверки того, не была ли эта библиотека уже установлена.

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

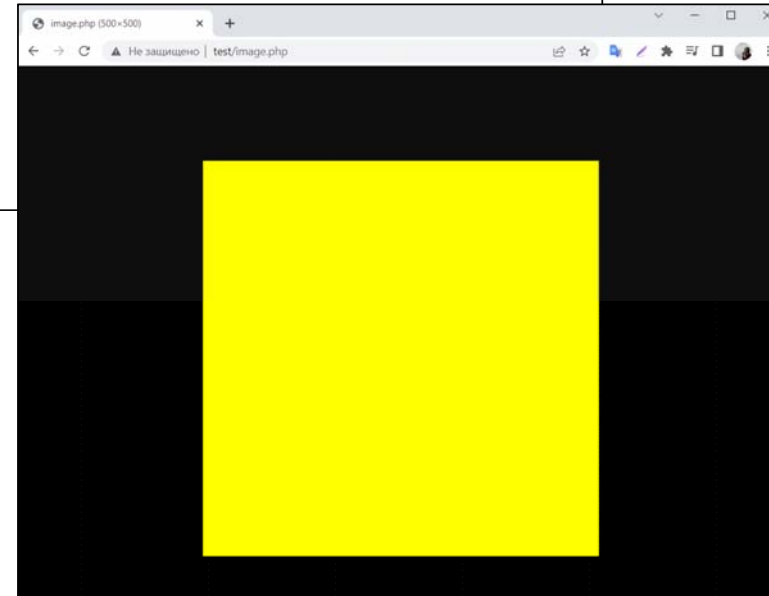


The screenshot shows a web browser window with the address bar displaying 'test/image.php'. The browser's address bar also shows a warning icon and the text 'Не защищ | te...'. The main content area of the browser displays the output of the PHP script as a JSON-like array:

```
array(14) { ["GD Version"]=> string(26) "bundled (2.1.0 compatible)"  
["FreeType Support"]=> bool(true) ["FreeType Linkage"]=> string(13) "with  
freetype" ["GIF Read Support"]=> bool(true) ["GIF Create Support"]=>  
bool(true) ["JPEG Support"]=> bool(true) ["PNG Support"]=> bool(true)  
["WBMP Support"]=> bool(true) ["XPM Support"]=> bool(true) ["XBM  
Support"]=> bool(true) ["WebP Support"]=> bool(true) ["BMP Support"]=>  
bool(true) ["TGA Read Support"]=> bool(true) ["JIS-mapped Japanese Font  
Support"]=> bool(false) }
```

Генерация нового изображения

```
<?php
    $i = imageCreate(500, 500);
    $color = imageColorAllocate($i, 255, 255, 0);
    imageFilledRectangle($i, 0, 0, imageSX($i), imageSY($i), $color);
    Header("Content-type: image/jpeg");
    imageJpeg($i);
    imageDestroy($i);
?>
```



Генерация изображений

imageCreate(int \$width, int \$height) - эта функция возвращает идентификатор изображения шириной **width** и высотой **height**. Если объяснить более понятным языком, то Вы этим действием создаёте "чистый холст для рисования".

Также есть функция *imagecreatetruecolor()* с аналогичными параметрами. Она обрабатывает изображения с максимально возможным качеством.

Генерация изображений

imageColorAllocate(resource \$image, int \$red, int \$green, int \$blue) - функция возвращает идентификатор цвета со следующими составляющими: красной (red), зелёной (green), синей (blue).

`imagecolorallocate()` должна вызываться для создания каждого цвета, который будет использоваться в изображении `image`. Первый вызов `imagecolorallocate()` задает цвет фона в палитровых изображениях - изображениях, созданных функцией `imagecreate()`.

Генерация изображений

imageFilledRectangle(resource \$image, int \$x1, int \$y1, int \$x2, int \$y2, int \$color) - эта функция рисует закрашенный прямоугольник на холсте image с координатами левого верхнего угла - x1 и y1, и координатами правого нижнего угла - x2 и y2, и цветом color.

imageSX(resource \$image) - возвращает ширину изображения image.

imageSY(resource \$image) - возвращает высоту изображения image.

header(\$string) - функция, которая посылает заголовок серверу. В данном случае мы сообщили, что наш контент имеет тип "image/jpeg". Это очень важная строка, и из-за её отсутствия очень часто возникают ошибки.

Генерация изображений

imageJpeg(resource \$image) - эта функция "выбрасывает" изображение **image** на экран. То есть в предыдущей строке мы сообщили серверу, что сейчас будет отправлено изображение, а уже этой строкой отправили само изображение.

imageDestroy(resource \$image) - «уничтожение» изображения **image**. Всегда надо использовать эту функцию, чтобы освободить память на сервере.

Генерация изображений

Imagecreatetruecolor

Создание нового полноцветного изображения (используется аналогично функции `imagecreate`). Возвращает идентификатор изображения, представляющий черное изображение заданного размера.

Параметры функции:

width – ширина изображения.

Height - высота изображения.

Генерация изображений

Imagecolorallocatealpha

Создание цвета для изображения. Работает аналогично функции `imagecolorallocate()`, но еще добавляет к цвету параметр `alpha`, отвечающий за прозрачность.

```
int imagecolorallocatealpha ( resource $image , int $red , int $green ,  
int $blue , int $alpha )
```

Генерация изображений

Imagecolorallocatealpha

Параметры:

image - изображения, полученный одной из функций создания изображений, например, такой как `imagecreatetruecolor()`.

red - значение красного компонента цвета.

green - значение зеленого компонента цвета.

blue - значение синего компонента цвета.

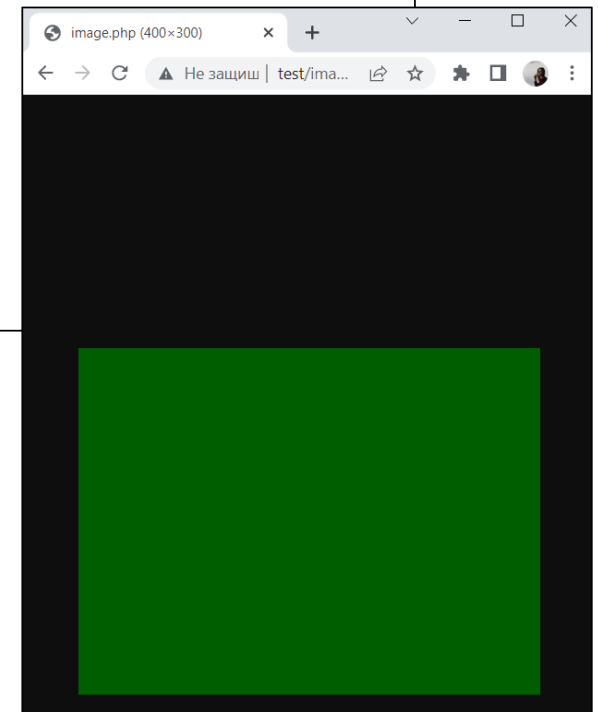
alpha - значение в диапазоне от 0 до 127. 0 означает непрозрачный цвет, 127 означает полную прозрачность. Параметры *red*, *green* и *blue* могут быть либо целочисленными в диапазоне от 0 до 255 либо шестнадцатеричными в диапазоне от 0x00 до 0xFF.

Генерация изображений

```
<?php
$i = imagecreatetruecolor(400, 300);
$color = imagecolorallocatealpha($i, 0, 255, 0, 80);
imagefilledrectangle($i, 0, 0, imageSX($i), imageSY($i), $color);
Header("Content-type: image/jpeg");

imageJpeg($i);
imageDestroy($i);

?>
```



Рисование пикселей

imageSetPixel

Рисует пиксель. Выводит один пиксель цвета color в изображении im, расположенный в точке (x, y).

```
int imageSetPixel(int im, int x, int y, int color)
```

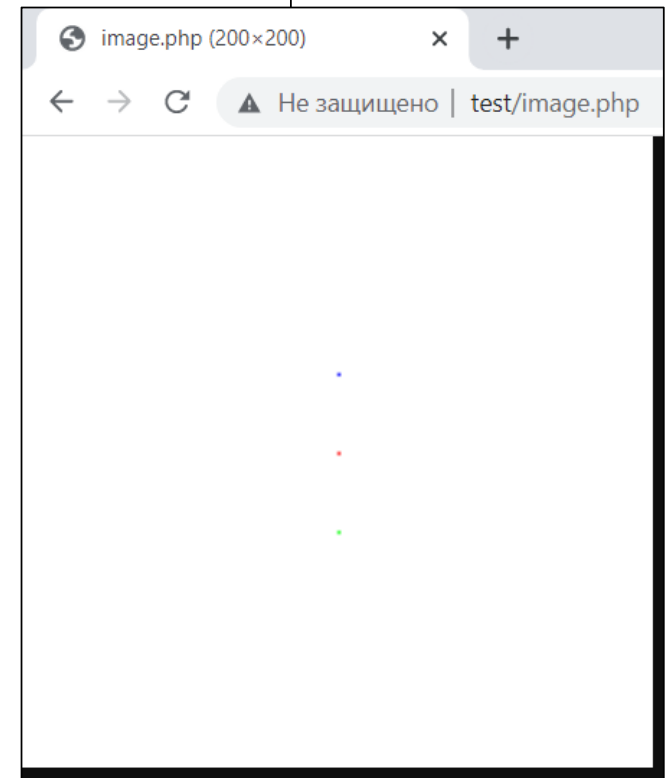
imageSetPixel

```
<?php
$i = imagecreatetruecolor(200, 200);
$color = imagecolorallocate($i, 255,255,255);
imagefilledrectangle($i, 0, 0, imageSX($i), imageSY($i), $color);
// Красная точка
$color = imagecolorallocate($i, 255, 0, 0);
imagesetpixel($i, 100, 100, $color);

// Зелёная точка
$color = imagecolorallocate($i, 0, 255, 0);
imagesetpixel($i, 100, 125, $color);

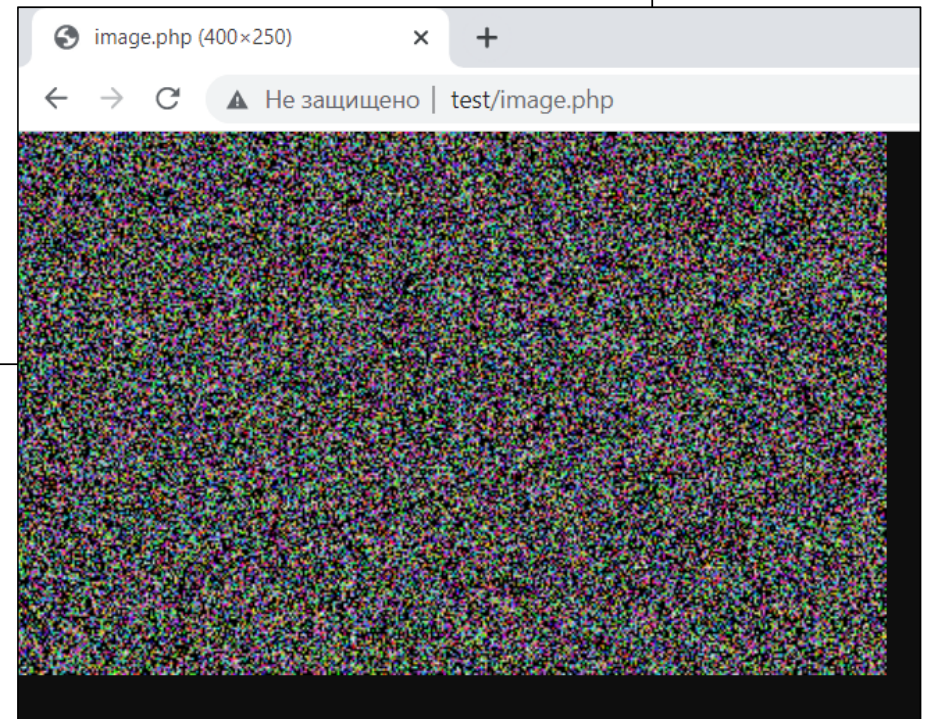
// Синяя точка
$color = imagecolorallocate($i, 0, 0, 255);
imagesetpixel($i, 100, 75, $color);

header('Content-Type: image/png');
imagepng($i);
imageDestroy($i);
?>
```



imageSetPixel

```
<?php
function set_rgb_noise($im, $w, $h, $num){
    for ($i = 0; $i < $num; $i++) {
        $color = imageColorAllocate($im, mt_rand(0, 255), mt_rand(0,
255), mt_rand(0, 255));
        imageSetPixel($im, mt_rand(0, $w), mt_rand(0, $h), $color);
    }
}
$im = imagecreatetruecolor(400, 250);
set_rgb_noise($im, 400, 250, 400 * 250);
header('Content-Type: image/png');
imagepng($im);
imageDestroy($im);
?>
```



Графические примитивы

imageLine

Рисует сплошную тонкую линию. Эта функция рисует сплошную тонкую линию в изображении `im`, проходящую через точки $(x1, y1)$ и $(x2, y2)$, цветом `color`.

```
int imageLine(int im, int x1, int y1, int x2, int y2, int color)
```

Графические примитивы

imageDashedLine

Рисует пунктирную линию. Эта функция работает почти так же, как и `imageLine()`, только рисует не сплошную, а пунктирную линию. К сожалению, ни размер, ни шаг штрихов задавать нельзя, так что, если вам обязательно нужна пунктирная линия произвольной фактуры, придется заняться математическими расчетами и использовать `imageLine()`.

```
int imageDashedLine(int im, int x1, int y1, int x2, int y2, int color)
```

Графические примитивы

`imagesetthickness()` задает значение толщины линий для рисования отрезков, прямоугольников, многоугольников, эллипсов и т.п. в `thickness` пикселей.

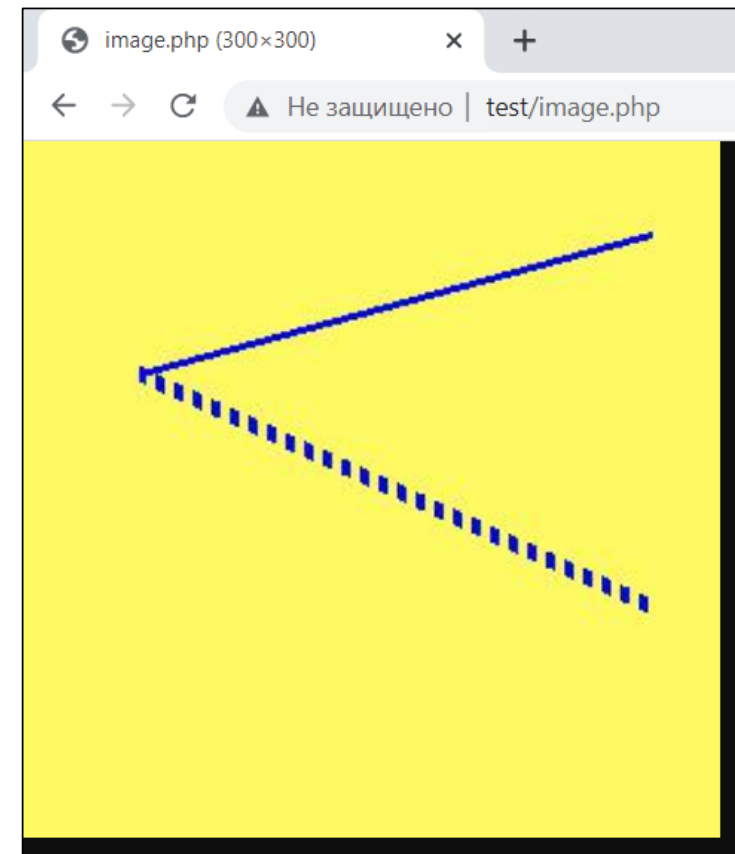
```
bool imagesetthickness ( resource $image , int $thickness )
```

Графические примитивы

```
<?php
$i = imageCreate(300, 300);
$color = imageColorAllocate($i, 255, 250, 100);
$color = imageColorAllocate($i, 0, 0, 255);

imageSetThickness($i, 3);
imageLine($i, 50, 100, 270, 40, $color);
imageDashedLine($i, 50, 100, 270, 200, $color);

Header("Content-type: image/jpeg");
imageJpeg($i);
imageDestroy($i);
?>
```



Графические примитивы

imagesetbrush

Установка изображения (кисти), посредством которого будут рисоваться линии. Задаёт изображение, которое будет использоваться функциями для рисования линий (такими как `imageline()` и `imagepolygon()`) при использовании цветов `IMG_COLOR_BRUSHED` или `IMG_COLOR_STYLEDBRUSHED`.

```
bool imagesetbrush ( resource $image , resource $brush )
```

Параметры:

`image` - ресурс изображения, полученный одной из функций создания изображений, например, такой как `imagecreatetruecolor()`.

`brush` - ресурс изображения.

Графические примитивы

imagesetstyle

устанавливает стиль рисования линий. `imagesetstyle()` задает стиль, который будет использоваться функциями рисования линий (такими как `imageline()` и `imagepolygon()`) при задании специального цвета `IMG_COLOR_STYLED` или `IMG_COLOR_STYLEDBRUSHED`.

```
bool imagesetstyle ( resource $image , array $style )
```

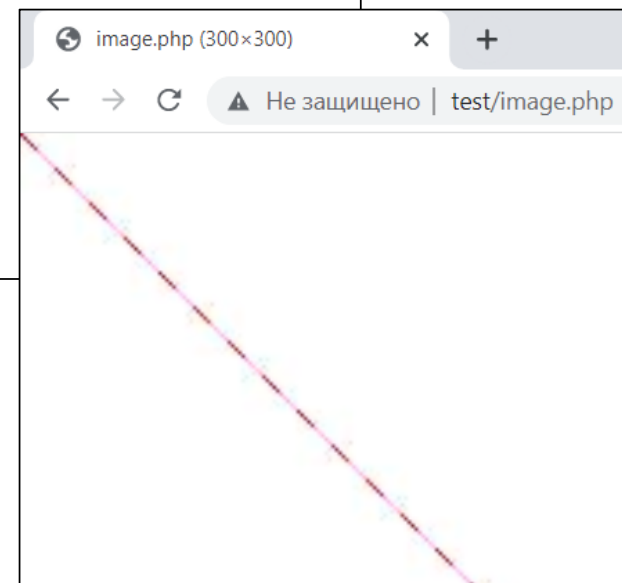
Параметры:

image - ресурс изображения, полученный одной из функций создания изображений, например, такой как `imagecreatetruecolor()`.

style - массив с цветами пикселей. Можно использовать константу `IMG_COLOR_TRANSPARENT` для добавления прозрачной точки. Обратите внимание, что `style` не должен быть пустым массивом.

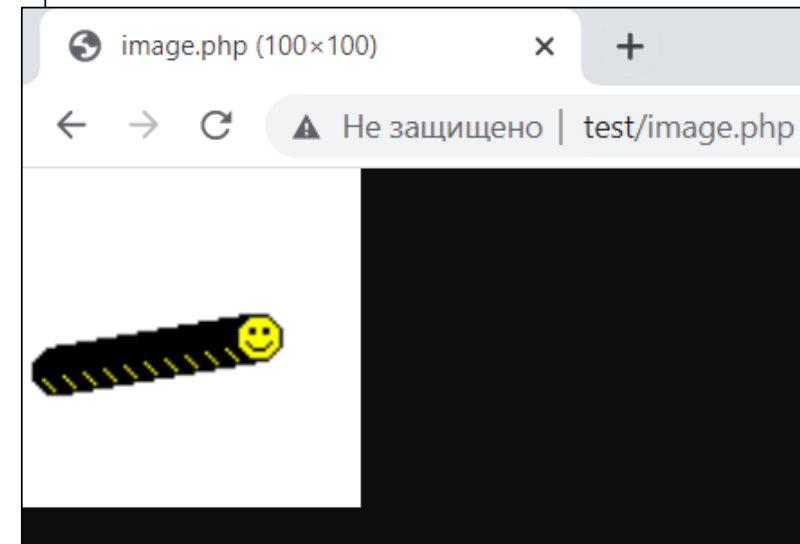
imagesetstyle

```
<?php
header("Content-type: image/jpeg");
$im = imagecreatetruecolor(300, 300);
$w = imagecolorallocate($im, 255, 170, 255);
$red = imagecolorallocate($im, 255, 0, 0);
$color = imagecolorallocate($im, 255, 255, 255);
imageFilledRectangle($im, 0, 0, imageSX($im), imageSY($im), $color);
/* Рисование пунктирной линии, 5 красных точек, 5 белых */
$style = array($red, $red, $red, $red, $red, $w, $w, $w, $w, $w);
imagesetstyle($im, $style);
imageline($im, 0, 0, 300, 300, IMG_COLOR_STYLED);
imagejpeg($im);
imagedestroy($im);
?>
```



imagesetbrush

```
<?php
// Загрузка картинки для кисти
$php = imagecreatefrompng("smile.happy.png");
// Создание основного изображения 100x100
$im = imagecreatetruecolor(100, 100);
// Заливка фона белым цветом
$white = imagecolorallocate($im, 255, 255, 255);
imagefilledrectangle($im, 0, 0, 100, 100, $white);
// Установка кисти
imagesetbrush($im, $php);
// Рисование линии из изображений кисти
imageline($im, 70, 50, 10, 60, IMG_COLOR_BRUSHED);
// Вывод и очистка памяти
header('Content-type: image/png');
imagepng($im);
imagedestroy($im);
imagedestroy($php);
?>
```



Графические примитивы

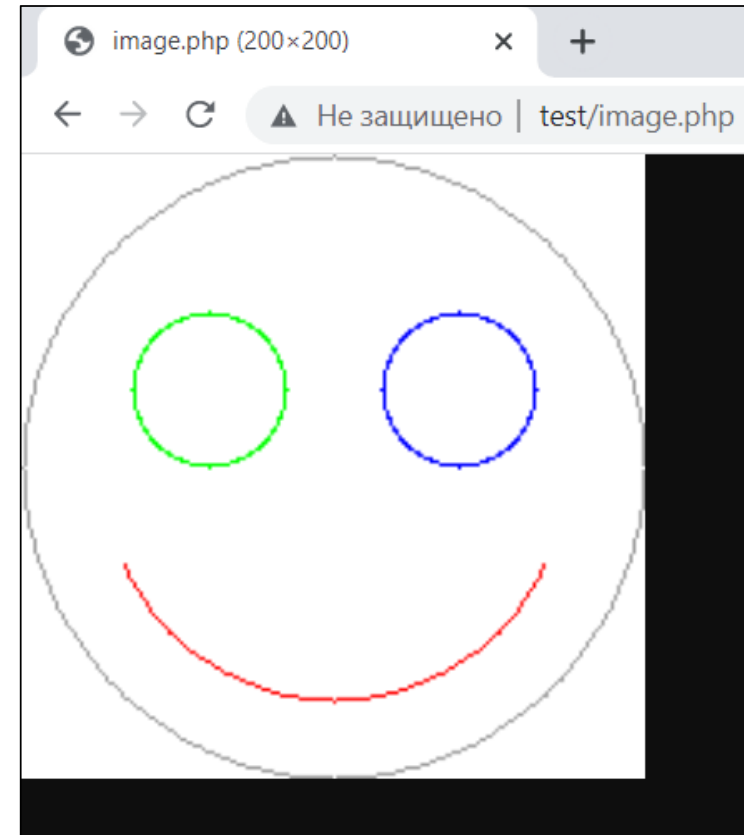
imageArc

Эта функция рисует в изображении *im* дугу сектора эллипса от угла *s* до *e* (углы указываются в градусах против часовой стрелки, отсчитываемых от горизонтали). Эллипс рисуется такого размера, чтобы вписываться в прямоугольник (*w*, *h*), где *w* и *h* задают его ширину и высоту. *cx* и *cy* - координаты центра эллипса. Сама фигура не закрашивается, обводится только ее контур, для чего используется цвет *color*.

```
int imageArc(int im, int cx, int cy, int w, int h, int s, int e,  
int color)
```

imageArc

```
<?php
$img = imagecreatetruecolor(200, 200);
// создаем несколько цветов
$white = imagecolorallocate($img, 255, 255, 255);
$color = imagecolorallocate($img, 155, 155, 155);
$red = imagecolorallocate($img, 255, 0, 0);
$green = imagecolorallocate($img, 0, 255, 0);
$blue = imagecolorallocate($img, 0, 0, 255);
imageFilledRectangle($img, 0, 0, imageSX($img),
imageSY($img), $white);
// рисуем голову
imagearc($img, 100, 100, 200, 200, 0, 360, $color);
// рот
imagearc($img, 100, 100, 150, 150, 25, 155, $red);
// глаза
imagearc($img, 60, 75, 50, 50, 0, 360, $green);
imagearc($img, 140, 75, 50, 50, 0, 360, $blue);
header("Content-type: image/png");
imagepng($img);
imagedestroy($img);
?>
```



Графические примитивы

imageFill

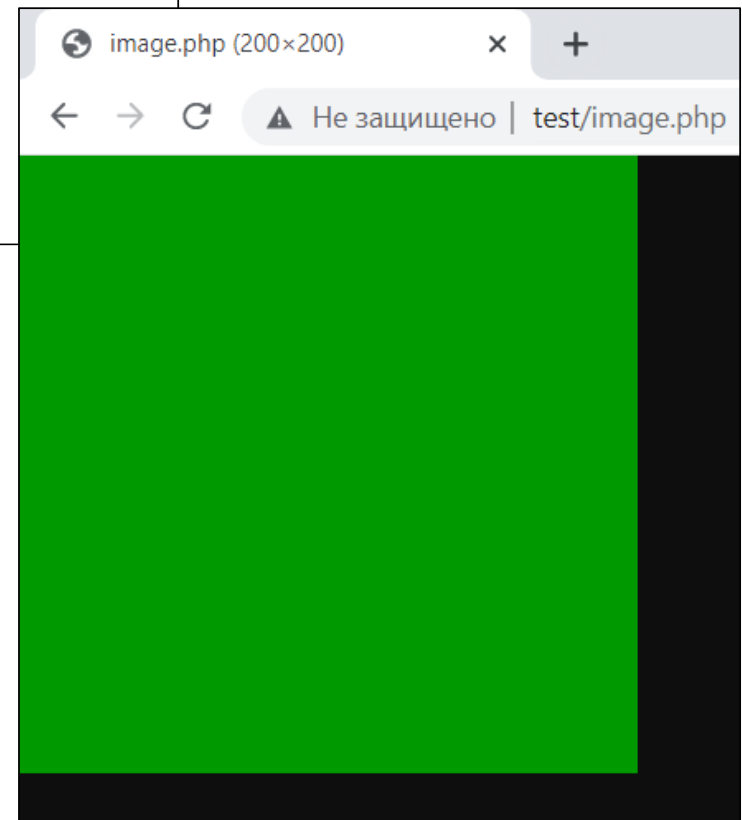
Эта функция выполняет сплошную заливку одноцветной области, содержащей точку с координатами (x, y) цветом color. Нужно заметить, что современные алгоритмы заполнения работают довольно эффективно, так что не стоит особо заботиться о скорости ее работы. Будут закрашены только те точки, к которым можно проложить "одноцветный сильно связанный путь" из точки x, y.

Две точки называются связанными сильно, если у них совпадает, по крайней мере, одна координата, а по другой координате они отличаются не более, чем на 1 в любую сторону.

```
int imageFill(int im, int x, int y, int color)
```

imageFill

```
<?php
$image = imagecreatetruecolor(500, 400);
$green = imagecolorallocate($image, 0, 153, 0);
imagefill($image, 0, 0, $green);
header('Content-type: image/png');
imagepng($image);
imagedestroy($image);
?>
```



Графические примитивы

imageFillToBorder

Эта функция очень похожа на `imageFill()`, только она выполняет закрашку цветом `color` не одноцветных точек, а любых, но до тех пор, пока не будет достигнута граница цвета `border`.

```
int imageFillToBorder(int im, int x, int y, int border, int color)
```

imageFillToBorder

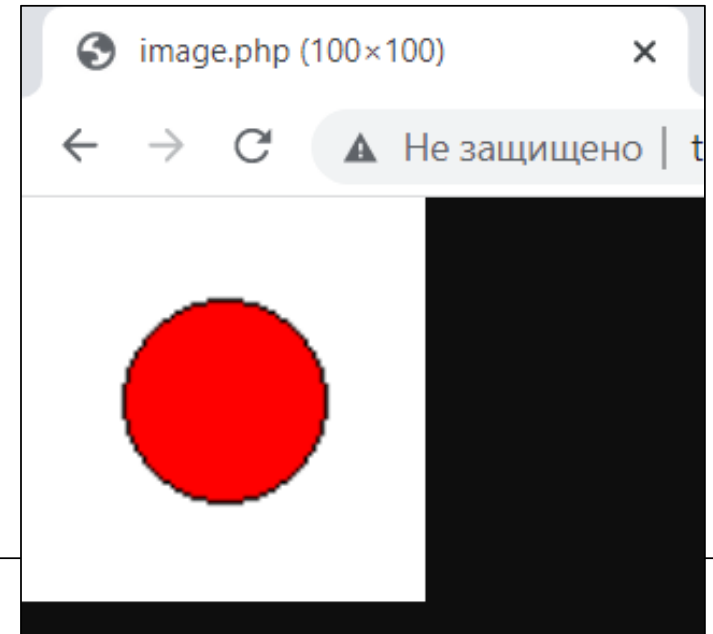
```
<?php
// создание изображения, установка белого фона
$im = imagecreatetruecolor(100, 100);
imagefilledrectangle($im, 0, 0, 100, 100, imagecolorallocate($im, 255, 255, 255));

// рисование эллипса, закрашенного черным цветом
imageellipse($im, 50, 50, 50, 50, imagecolorallocate($im, 0, 0, 0));

// установка цвета границы заливки
$border = imagecolorallocate($im, 0, 0, 0);
$fill = imagecolorallocate($im, 255, 0, 0);

// заливка области
imagefilltoborder($im, 50, 50, $border, $fill);

header('Content-type: image/png');
imagepng($im);
imagedestroy($im);
?>
```



Графические примитивы

imageellipse

Рисует эллипс с центром в заданных координатах.

```
bool imageellipse ( resource $image , int $cx , int $cy , int $width ,  
int $height , int $color )
```

image – ресурс изображения, полученный одной из функций создания изображений, например, такой как `imagecreatetruecolor()`.

cx - x-координата центра.

cy - y-координата центра.

width - ширина эллипса.

height - высота эллипса.

color - цвет заливки. Идентификатор цвета, созданный функцией `imagecolorallocate()`.

Графические примитивы

imagefilledellipse

Рисует закрашенный эллипс с центром в заданных координатах.

```
bool imagefilledellipse ( resource $image , int $cx , int $cy ,  
int $width , int $height , int $color )
```

image – ресурс изображения, полученный одной из функций создания изображений, например, такой как `imagecreatetruecolor()`.

cx - x-координата центра.

cy - y-координата центра.

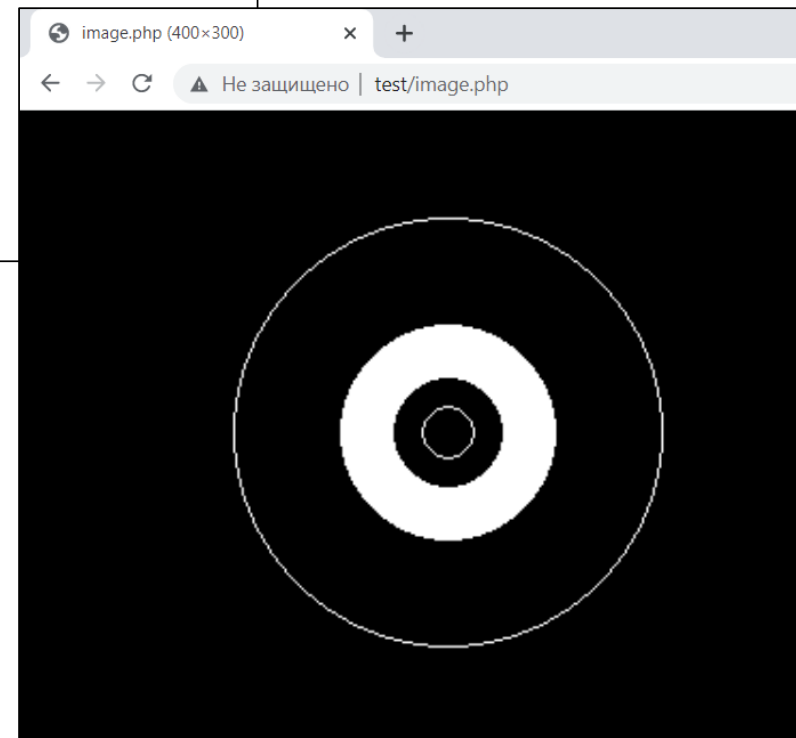
width - ширина эллипса.

height - высота эллипса.

color - цвет заливки. Идентификатор цвета, созданный функцией `imagecolorallocate()`.

imageellipse u imagefilledellipse

```
<?php
$image = imagecreatetruecolor(400, 300);
$bg = imagecolorallocate($image, 0, 0, 0);
$white = imagecolorallocate($image, 255, 255, 255);
$black = imagecolorallocate($image, 0, 0, 0);
imageellipse($image, 200, 150, 200, 200, $white);
imagefilledellipse($image, 200, 150, 100, 100, $white);
imagefilledellipse($image, 200, 150, 50, 50, $black);
imageellipse($image, 200, 150, 25, 25, $white);
header("Content-type: image/png");
imagepng($image);
?>
```



Imageellipse, imagefilledellipse, imageArc

```
<?php
$i = imageCreate(450, 250);
$color = imageColorAllocate($i, 255, 250, 100);

imageSetThickness($i, 3);
$red = imagecolorallocate($i, 255, 0, 0);
imagearc($i, 380, 100, 80, 80, 180, 360, $red);
imagearc($i, 50, 100, 100, 100, 0, 360, $red);

$blue=imagecolorallocate($i, 0, 0, 255);
imageellipse($i, 220, 100, 150, 150, $blue);

$green=imagecolorallocate($i, 0, 255, 0);
imagefilledellipse($i, 220, 100, 90, 90, $green);

Header("Content-type: image/jpeg");
imageJpeg($i);
imageDestroy($i);
?>
```



Графические примитивы

imagePolygon

Эта функция рисует в изображении *im* многоугольник, заданный своими вершинами. Координаты углов передаются в массиве *points*, причем *\$points[0]=x0*, *\$points[1]=y0*, *\$points[2]=x1*, *\$points[3]=y1* и т.д.

Параметр *num_points* указывает общее число вершин - на тот случай, если в массиве их больше, чем нужно нарисовать. Многоугольник не закрашивается - только рисуется его граница цветом *color*.

```
int imagePolygon(int im, array points, int num_points, int color)
```

Графические примитивы

imageFilledPolygon

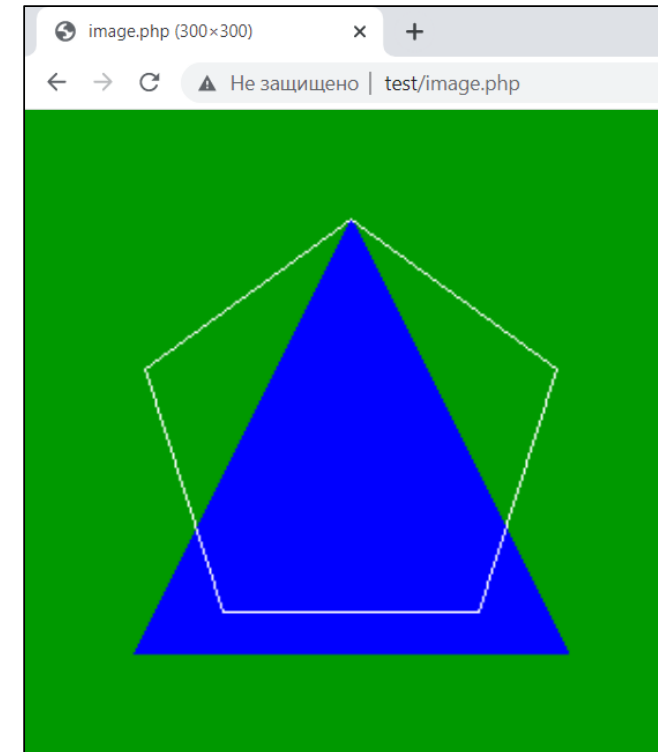
Эта функция делает практически то же самое, что и `imagePolygon()`, за исключением одного очень важного свойства: полученный многоугольник целиком заливается цветом `color`.

```
int imageFilledPolygon(int im, array points, int num_points, int color)
```

imagePolygon, imageFilledPolygon

```
<?php
$values = array(
    150, 50,  // Point 1 (x, y)
    50,  250, // Point 2 (x, y)
    250, 250  // Point 3 (x, y)
);
$values2 = array(
150, 50,  55, 119, 91, 231, 209, 231,  245, 119 );
$image = imagecreatetruecolor(300, 300);
$background_color = imagecolorallocate($image, 0, 153, 0);

imagefill($image, 0, 0, $background_color);
$white = imagecolorallocate($image, 255, 255, 255);
$blue = imagecolorallocate($image, 0, 0, 255);
imageFilledPolygon($image, $values, 3, $blue);
imagepolygon($image, $values2, 5, $white);
header('Content-type: image/png');
imagepng($image);
imagedestroy($image);
?>
```



Графические примитивы

imagechar

`imagechar()` рисует первый символ аргумента на изображении с идентификатором `image` на координатах `x,y` (координаты отсчитываются с левого верхнего угла) цветом `color`.

Параметры:

image - ресурс изображения, полученный одной из функций создания изображений, например, такой как `imagecreatetruecolor()`.

font - может принимать значения 1, 2, 3, 4, 5 для встроенных шрифтов в кодировке latin2 (более высокое число соответствует большему шрифту) или любому из ваших собственных идентификаторов шрифтов, зарегистрированных с помощью `imageloadfont()`.

x - x-координата начала рисования.

y - y-координата начала рисования.

c - символ для рисования.

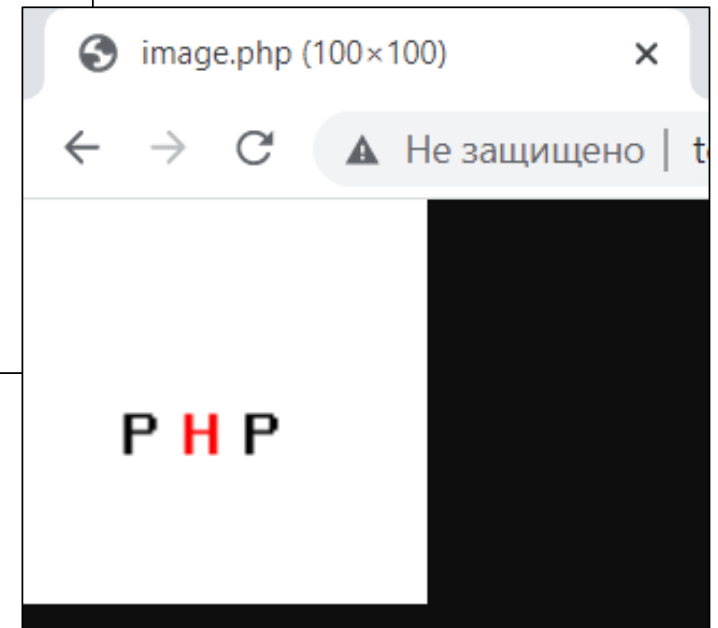
color - идентификатор цвета, созданный функцией `imagecolorallocate()`.

```
bool imagechar ( resource $image , int $font , int $x , int $y ,  
string $c , int $color )
```

imagechar

```
<?php
$im = imagecreate(100, 100);
$string = 'P';
$bg = imagecolorallocate($im, 255, 255, 255);
$black = imagecolorallocate($im, 0, 0, 0);
$red = imagecolorallocate($im, 255, 0, 0);
// печатает черный символ "P" в левом верхнем углу
imagechar($im, 5, 25, 50, 'P', $black);
imagechar($im, 5, 40, 50, 'H', $red);
imagechar($im, 5, 55, 50, 'P', $black);

header('Content-type: image/png');
imagepng($im);
?>
```



Графические примитивы

imagecharup

Рисует символ вертикально на заданных координатах изображения `image`.

Параметры:

image - ресурс изображения, полученный одной из функций создания изображений, например, такой как `imagecreatetruecolor()`.

font - может принимать значения 1, 2, 3, 4, 5 для встроенных шрифтов в кодировке latin2 (более высокое число соответствует большему шрифту) или любому из ваших собственных идентификаторов шрифтов, зарегистрированных с помощью `imageloadfont()`.

x - x-координата начала рисования.

y - y-координата начала рисования.

c - символ для рисования.

color - идентификатор цвета созданных функцией `imagecolorallocate()`.

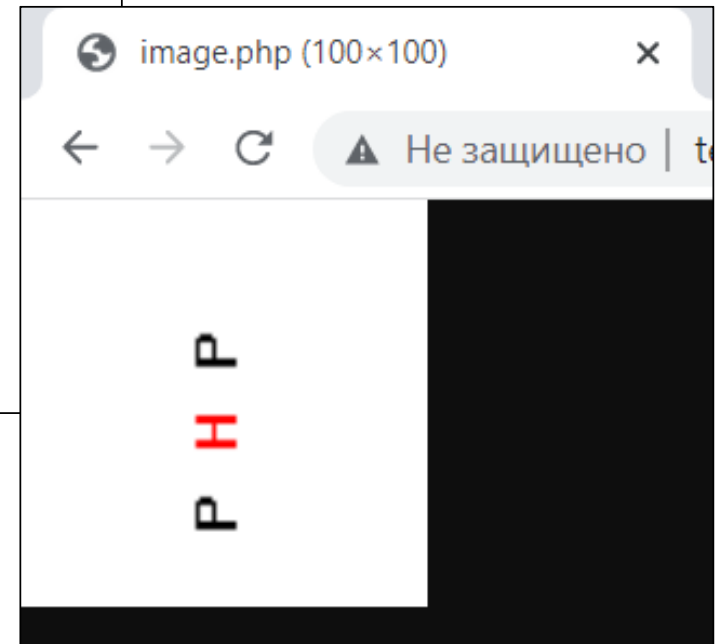
```
bool imagecharup ( resource $image , int $font , int $x , int $y ,  
string $c , int $color )
```


imagecharup

```
<?php
$im = imagecreate(100, 100);
$string = 'P';
$bg = imagecolorallocate($im, 255, 255, 255);
$black = imagecolorallocate($im, 0, 0, 0);
$red = imagecolorallocate($im, 255, 0, 0);

imagecharup($im, 5, 40, 80, 'P', $black);
imagecharup($im, 5, 40, 60, 'H', $red);
imagecharup($im, 5, 40, 40, 'P', $black);

header('Content-type: image/png');
imagepng($im);
?>
```



Графические примитивы

imagestring

Рисует текст string на заданных координатах.

Список параметров:

image - ресурс изображения, полученный одной из функций создания изображений, например, такой как imagecreatetruecolor().

font - может принимать значения 1, 2, 3, 4, 5 для встроенных шрифтов в кодировке latin2 (более высокое число соответствует большему шрифту) или любому из ваших собственных идентификаторов шрифтов, зарегистрированных с помощью imageloadfont().

x - x-координата верхнего левого угла.

y - y-координата верхнего левого угла.

string - строка текста.

color - идентификатор цвета, создаваемый функцией imagecolorallocate().

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

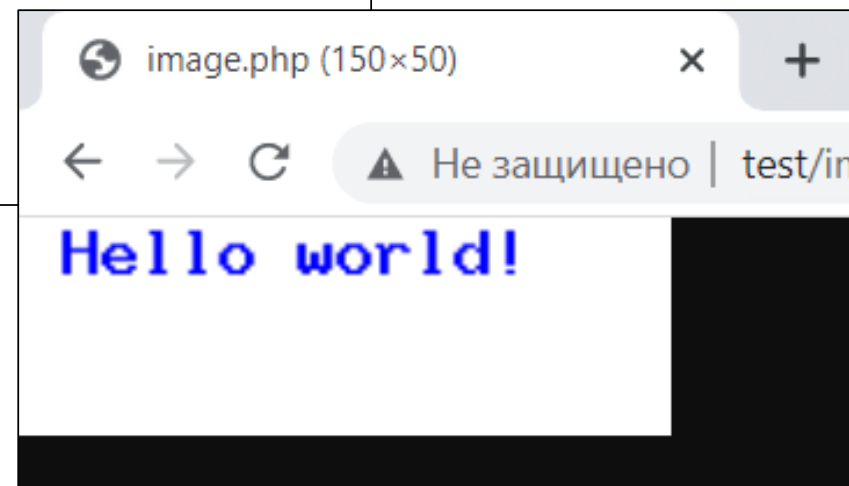
imagestring

```
<?php
$im = imagecreate(150, 50);

$bg = imagecolorallocate($im, 255, 255, 255);
$textcolor = imagecolorallocate($im, 0, 0, 255);

imagestring($im, 5, 10, 0, 'Hello world!', $textcolor);

header('Content-type: image/png');
imagepng($im);
imagedestroy($im);
?>
```



Графические примитивы

imagestringup

Рисует текст `string` вертикально на заданных координатах. Список параметров:

image - ресурс изображения, полученный одной из функций создания изображений, например, такой как `imagecreatetruecolor()`.

font - может принимать значения 1, 2, 3, 4, 5 для встроенных шрифтов в кодировке latin2 (более высокое число соответствует большему шрифту) или любому из ваших собственных идентификаторов шрифтов, зарегистрированных с помощью `imageloadfont()`.

x - x-координата нижнего левого угла.

y - y-координата нижнего левого угла.

string - текст надписи.


color - идентификатор цвета, созданный функцией `imagecolorallocate()`.

```
bool imagestringup ( resource $image , int $font , int $x , int $y ,
string $string , int $color )
```

imagestringup

```
<?php
$im = imagecreatetruecolor(150, 150);
$textcolor = imagecolorallocate($im, 0, 0, 255);
imagestringup($im, 5, 70, 120, 'Hello world!', $textcolor);
imagepng($im, './stringup.png');
imagedestroy($im);
?>
```

Изображение будет сохранено в текущем каталоге.



Hello world!

Графические примитивы

Imagettftext

Рисование текста на изображении шрифтом TrueType.

```
imagettftext(  
    GdImage $image,  
    float $size,  
    float $angle,  
    int $x,  
    int $y,  
    int $color,  
    string $font_filename,  
    string $text,  
    array $options = []  
): array|false
```

Графические примитивы

imaggittftext

Список параметров

image

Объект GdImage, возвращаемый одной из функций создания изображений, например, такой как imagecreatetruecolor().

size

Размер шрифта в типографских пунктах.

angle

Угол в градусах, 0 градусов означает расположение текста слева направо. Положительные значения означают поворот текста против часовой стрелки. Например, текст повернутый на 90 градусов нужно будет читать снизу-вверх.

Графические примитивы

`imagefttext`

`x`

Координаты `x` и `y` определяют отправную точку для первого символа текста (конкретно, левый нижний угол символа). Здесь есть отличие от функции `imagestring()`, в которой `x` и `y` определяют верхний левый угол первого символа. Например, "верхний левый" имеет координаты 0,0.

`y`

`y`-координата. Это позиция базовой линии шрифта, в общем случае она не совпадает с низшей точкой в символе.

`color`

Индекс цвета. Использование отрицательных индексов создаёт эффект выключенного сглаживания. Смотрите `imagecolorallocate()`.

Графические примитивы

`image`
`ttf`
`text`

`fontfile`

Путь к шрифту TrueType, который вы хотите использовать.

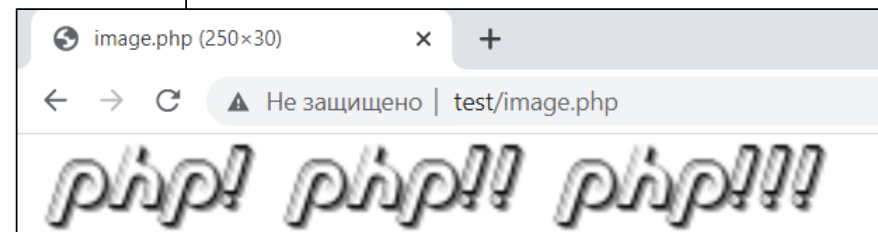
`text`

Текстовая строка в кодировке UTF-8.

Для доступа к символам после 127-го можно использовать числовые ссылки (в формате: `€`). Также поддерживается шестнадцатеричный формат (`©`). Строки в кодировке UTF-8 можно передавать напрямую.

imageTTFtext

```
<?php
header('Content-Type: image/png');
$im = imagecreatetruecolor(250, 30);
// Создание цветов
$white = imagecolorallocate($im, 255, 255, 255);
$grey = imagecolorallocate($im, 128, 128, 128);
$black = imagecolorallocate($im, 0, 0, 0);
imagefilledrectangle($im, 0, 0, 399, 29, $white);
// Текст надписи
$text = 'PHP! PHP!! PHP!!!';
// Замена пути к шрифту на пользовательский
$font = 'metroplex.ttf';
// Тень
imageTTFtext($im, 20, 0, 11, 21, $grey, $font, $text);
// Текст
imageTTFtext($im, 20, 0, 10, 20, $black, $font, $text);
imagepng($im);
imagedestroy($im);
?>
```



Операции с изображениями. Поворот

```
resource imagerotate ( resource $image , float $angle , int $bgd_color  
[ , int $ignore_transparent = 0 ] )
```

Поворот изображения **image** на заданный угол **angle** в градусах.

Центром поворота является центр изображения. Поворачиваемое изображение может отличаться размером от оригинала.

Список параметров:

image

Ресурс изображения, полученный одной из функций создания изображений.

angle

Угол поворота в градусах против часовой стрелки.

bgd_color

Цвет фона свободной зоны после поворота.

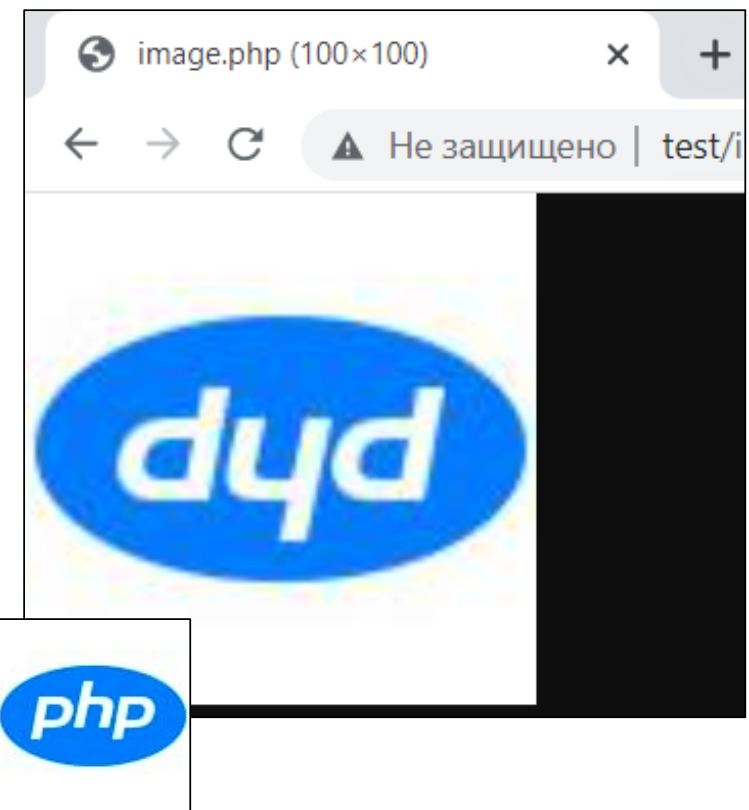
ignore_transparent

Если установлено и не равно нулю, прозрачность игнорируются (иначе сохраняется).

Операции с изображениями. Поворот

imagerotate

```
<?php
$filename = 'clouds.jpg';
$degrees = 180;
header('Content-type: image/jpeg');
$source = imagecreatefromjpeg($filename);
// Поворот
$rotate = imagerotate($source, $degrees, 0);
imagejpeg($rotate);
imagedestroy($source);
imagedestroy($rotate);
?>
```



Операции с изображениями. Масштабирование

`imagescale()` — масштабирует изображение по заданной ширине и высоте. В отличие от многих функций по работе с изображениями, `imagescale()` не изменяет переданный параметр `image`; вместо него будет возвращено новое изображение.

```
imagescale(  
    GdImage $image,  
    int $width,  
    int $height = -1,  
    int $mode = IMG_BILINEAR_FIXED  
): GdImage|false
```

Операции с изображениями. Масштабирование

imagescale

Список параметров:

image

Объект GdImage, возвращаемый одной из функций создания изображений, например, такой как `imagecreatetruecolor()`.

width

Ширина для масштабирования.

height

Высота для масштабирования изображения. Если этот параметр опущен или отрицателен, соотношение сторон будет сохранено.

mode

Одна из констант `IMG_NEAREST_NEIGHBOUR`, `IMG_BILINEAR_FIXED`, `IMG_BICUBIC`, `IMG_BICUBIC_FIXED` или что-либо ещё (будет использовано два прохода).

Операции с изображениями. Масштабирование

`imagescale`

```
<?php
$image_name = 'clouds.jpg';
$image = imagecreatefromjpeg($image_name);
$img = imagescale( $image, 500, 200 );
header("Content-type: image/png");
imagepng($img);
?>
```



Операции с изображениями. Обрезка

```
imagecrop(GdImage $image, array $rectangle): GdImage|false
```

Обрезает изображение до заданной прямоугольной области и возвращает полученное изображение. Заданный параметр `image` не изменяется.

Список параметров:

image

Объект `GdImage`, возвращаемый одной из функций создания изображений, например, такой как `imagecreatetruecolor()`.

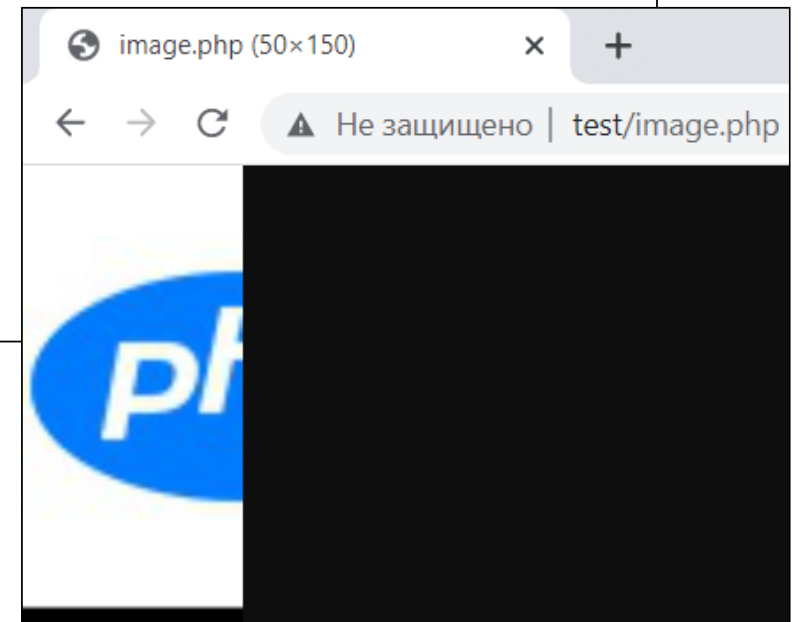
rectangle

Обрезанный прямоугольник в виде массива (`array`) с ключами `x`, `y`, `width` и `height`.

Операции с изображениями. Обрезка

imagecrop

```
<?php
$im = imagecreatefromjpeg('clouds.jpg');
$size = min(imagesx($im), imagesy($im));
$im2 = imagecrop($im, ['x' => 0, 'y' => 0, 'width' => 50, 'height' => 150]);
if ($im2 !== FALSE) {
    header("Content-type: image/png");
    imagepng($im2);
    imagedestroy($im2);
}
imagedestroy($im);
?>
```



Операции с изображениями. Переворот

```
imageflip(GdImage $image, int $mode): bool
```

Переворачивает изображение **image**, используя выбранный режим **mode**.

Список параметров:

image

Объект GdImage, возвращаемый одной из функций создания изображений, например, такой как imagecreatetruecolor().

mode

Режим переворота - одна из констант **IMG_FLIP_***:

IMG_FLIP_HORIZONTAL - переворачивает изображение по горизонтали.

IMG_FLIP_VERTICAL - переворачивает изображение по вертикали.

IMG_FLIP_BOTH - переворачивает изображение и по горизонтали и по вертикали.

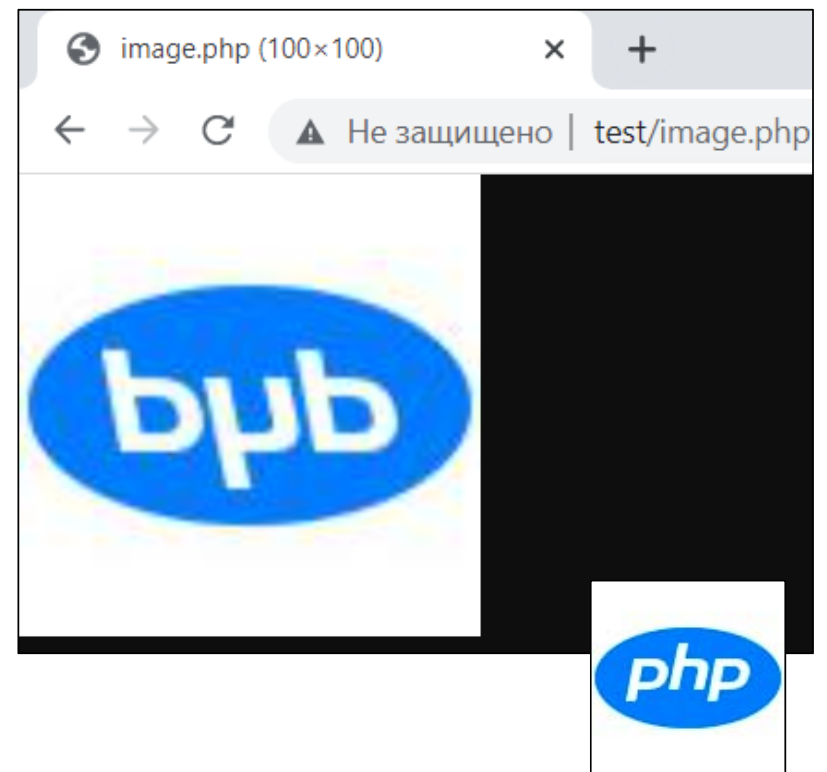
Операции с изображениями. Переворот

imageflip

```
<?php
$filename = 'clouds.jpg';
header('Content-type: image/png');
$im = imagecreatefromjpeg($filename);

// Переворачиваем по вертикали
imageflip($im, IMG_FLIP_VERTICAL);

imagejpeg($im);
imagedestroy($im);
?>
```



Применение фильтров к изображению

```
imagefilter(GdImage $image, int $filter, array|int|float|bool ...$args): bool
```

imagefilter() применяет заданный фильтр **filter** к изображению **image**.

Список параметров:

image

Объект GdImage, возвращаемый одной из функций создания изображений, например, такой как imagecreatetruecolor().

filter

filter может быть одним из следующих:

IMG_FILTER_NEGATE: Инвертирует все цвета изображения.

IMG_FILTER_GRAYSCALE: Преобразует цвета изображения в градации серого путём преобразования компонент красного, зелёного и синего в их сумму с учётом весов, таких же как при вычислении яркости (Y') по REC.601. Альфа-канал сохраняется. Для изображений, использующих палитру, результат может отличаться в виду ограничений, накладываемых палитрой.

Применение фильтров к изображению

```
imagefilter(GdImage $image, int $filter, array|int|float|bool ...$args): bool
```

IMG_FILTER_BRIGHTNESS: Изменяет яркость изображения. Используйте аргумент args для задания уровня яркости. Диапазон яркостей от -255 до 255.

IMG_FILTER_CONTRAST: Изменяет контрастность изображения. Используйте аргумент args для задания уровня контрастности.

IMG_FILTER_COLORIZE: То же, что и IMG_FILTER_GRAYSCALE, за исключением того, что можно задать цвет. Используйте аргументы args, arg2 и arg3 для указания каналов red, green, blue, а arg4 для alpha канала. Диапазон для каждого канала цвета от 0 до 255.

IMG_FILTER_EDGEDETECT: Использует определение границ для их подсветки.

IMG_FILTER_EMBOSS: Добавляет рельеф.

IMG_FILTER_GAUSSIAN_BLUR: Размывает изображение по методу Гаусса.

IMG_FILTER_SELECTIVE_BLUR: Размывает изображение.

IMG_FILTER_MEAN_REMOVAL: Использует усреднение для достижения эффекта "эскиза".

IMG_FILTER_SMOOTH: Делает границы более плавными, а изображение менее чётким. Используйте аргумент args для задания уровня гладкости.

Применение фильтров к изображению

```
imagefilter(GdImage $image, int $filter, array|int|float|bool ...$args): bool
```

IMG_FILTER_PIXELATE: Применяет эффект пикселирования. Используйте аргумент `args` для задания размера блока и аргумент `arg2` для задания режима эффекта пикселирования.

IMG_FILTER_SCATTER: Применяет эффект рассеивания к изображению, используйте `args` и `arg2` для определения силы эффекта и дополнительно `arg3` для применения только к выбранным цветам пикселей.

`args`

IMG_FILTER_BRIGHTNESS: Уровень яркости.

IMG_FILTER_CONTRAST: Уровень контрастности.

IMG_FILTER_COLORIZE: Значение красного компонента цвета.

IMG_FILTER_SMOOTH: Уровень сглаживания.

IMG_FILTER_PIXELATE: Размер блока в пикселах.

IMG_FILTER_SCATTER: Уровень уменьшения эффекта. Не должен быть больше или равен уровню добавления эффекта, установленному с помощью `arg2`

Применение фильтров к изображению

```
imagefilter(GdImage $image, int $filter, array|int|float|bool ...$args): bool
```

arg2

IMG_FILTER_COLORIZE: Значение зелёного компонента цвета.

IMG_FILTER_PIXELATE: Использовать усовершенствованный эффект пикселирования или нет (по умолчанию false).

IMG_FILTER_SCATTER: Уровень добавления эффекта.

arg3

IMG_FILTER_COLORIZE: Значение синего компонента цвета.

IMG_FILTER_SCATTER: Необязательный массив значений индексации цвета для применения эффекта.

arg4

IMG_FILTER_COLORIZE: Альфа канал, значение между 0 и 127. 0 означает непрозрачность, 127 соответствует абсолютной прозрачности.

Применение фильтров к изображению

imagefilter

```
<?php
$im = imagecreatefromjpeg('clouds.jpg');
header('Content-type: image/png');
if($im && imagefilter($im,
IMG_FILTER_GRAYSCALE))
{
    imagejpeg($im);
}
imagedestroy($im);
?>
```



Применение фильтров к изображению

imagefilter

```
<?php
$im = imagecreatefromjpeg('clouds.jpg');
header('Content-type: image/png');
if($im && imagefilter($im,
IMG_FILTER_COLORIZE, 0, 255, 0))
{
    imagejpeg($im);
}
imagedestroy($im);
?>
```



Применение фильтров к изображению

imagefilter

```
<?php
$im = imagecreatefromjpeg('clouds.jpg');
header('Content-type: image/png');
if($im && imagefilter($im,
IMG_FILTER_PIXELATE, 3))
{
    imagejpeg($im);
}
imagedestroy($im);
?>
```

