PHP: Създаване на изображения

Виолета Божикова

- С PHP може да създаване или променяте изображения, използвайки библиотеката GD.
- Библиотеката GD поддържа форматите JPEG, PNG и WBMP.
- Форматът GIF вече не се поддържа (по причини свързани с вида компресия, използван при GIF – вид, обект на патент).

Формати на изображения

- JPEG (Joint Photographic Experts Group) е името на група от стандарти. Когато говорим за JPEG файлове разбираме всъщност файловия формат JFIF (JPEG File Interchange Format), който отговаря на един от стандартите, включени в JPEG.
- JPEG файловете обикновено се използват за съхраняване на фотографски и други изображения с много цветове или градации на цвят. При този формат се използва компресия със загуби (намаляването на файловия размер обаче води до загуба на качеството на изображението). Този формат не е удачен за рисунки, съдържащи прави линии, текст или плътни блокове цвят.

Формати на изображения

PNG – Portable Network Graphics. Този формат се използва като заместител на GIF (Graphics Interchange Format). Това е формат за изображения с компресия без загуби (но файловият размер е по-голям от JPEG), като предлаганата компресия е по-добра от GIF, но анимация не се поддържа. PNG

Форматът PNG е приложим за изображения, съдържащи текст, прави линии и плътни блокове цвят.

Форматът **WBMP** означава Wireless Bitmap. Това е формат, специално проектиран за безжични устройства.

Как да създаваме изображения с PHP?

- **Изисквания:** И така, създаването и поддръжката на изображения в PHP става с библиотеката GD, тоест: GD библиотеката трябва да бъде инсталирана на сървъра (това е автоматично след PHP 4.3).
- **Процедура на създаване на изображения**:
  - Започваме със създаване на празно изображение, следва задаване на цвят на празното изображение.
  - Ще показваме малки парчета код и резултантните фигури. Към предходните парчета код ще добавяме нови, постепенно усложнявайки сложността на фигурите.

Пример 1. Създаване на квадрат 200x200

```
<?php
create_image();
print "<img src=image.png?".date("U").">";

function  create_image(){
$im = @imagecreate(200, 200) or die("Cannot Initialize new
    GD image stream");
$background_color = imagecolorallocate($im, 255, 255, 0);
// yellow, 000-черен фон.
      imagepng($im,"image.png");
      imagedestroy($im);
}
?>
```

Коментар на примера

- Примерът започва с извикване на потребителска функция **create_image()**, която създава изображение с жълт фонов цвят.

$im = @imagecreate(200, 200) or die("Cannot Initialize new GD image stream");

- "@“ - пред imagecreate() е с цел да се избегне извеждането на грешки.Параметрите в imagecreate(200,200) са размерите на новото изображение: ширина и височина в пиксели.

$background_color = imagecolorallocate($im, 255, 255, 0); // yellow фон.

- и го записва във файл **image.png**,

imagepng($im,"image.png");

Коментар на примера

Съхраненото изображение се извежда след това с php конструкция **print** в прозореца на браузъра (която извежда от своя страна html code, извеждащ изображението, наречено"image.png“):

print "<img src=image.png?".date("U").">";

- Добавили сме "**?**“ и стойността -date("U")- което ще позволи да се избегне показването на изображения от кеша на браузъра.
- date ("U") ще добави към изображението един номер (секунди след 1970/01/01 00:00:00). Тъй като този брой е различен всеки път когато се създава ново изображение, няма да се позволи браузърът да покаже отново изображението, съхранено в кеша; изображенията "? Image.png 1034691963" и "? Image.png 1034691964" ще се считат за различни от браузъра, и те ще бъдат изтеглени от сървъра.

За “Начините за създаване на празно изображение (изображение-платно)”

- И така, за да създадете или промените изображение в PHP, отначало трябва да създадете негов **идентификатор**.

Това може да стане по 2 основни начина:

- Първият начин е: да се създаде празно платно, чрез извикване на функцията **ImageCreate()**, както в пример1. Общ вид:

$im = imagecreate($width,$height);

Създаване на изображение чрез ImageCreate()

- Параметрите $width,$height в imagecreate() са размерите на новото изображение: ширина и височина в пиксели.
- При успех imagecreate() връща идентификатор на изображение (image identifier), представящ празно изображение със зададен размер или **FALSE** - при грешка.
- Когато се използва imagecreate() команда, първият дефиниран цвят с imagecolorallocate() се счита за background color на изображението.
- Може да се използва imagecreatetruecolor() команда вместо imagecreate команда, background color-а ще е black (но по нататък можем да нарисуваме квадрат с друг цвят и размери):

resource imagecreatetruecolor (int $width , int $height)

Създаване на изображение на база на съществуващо такова: ImageCreateFromPNG() и т.н.

- Вторият начин за създаване на изображение е да прочетете съществуващ файл с изображение, което след това може да се филтрира, да се променят размерите му или да се добавят нови елементи.
- Това може да стане с някоя от функциите ImageCreateFromPNG(), ImageCreateFromJIF(), в зависимост от файловия формат на изображението, което се чете. Например:

```
$im = imageCreateFromPNG('baseimage.png');
```

Избор на фонов цвят за изображението - Imagecolorallocate()

int **imagecolorallocate** (resource $image, int $red , int $green , int $blue).

Параметри

$image - *идентификатор на изображение, върнат от една от функциите, създаващи изображение*, например imagecreatetruecolor()

$red, $green, $blue – Стойности на червената, зелената и синята компонента.

Тези параметри са цели числа, между 0 и 255 или шестнадесетични числа, между 0x00 и 0xFF.

Връщана стойност: Връща идентификатор на цвят (color identifier), представляващ цвят на изображението, съставен от съответните RGB компоненти или FALSE (логическо FALSE или нелогическа стойност, която може да се изчисли като FALSE) – при неуспех.

В примера: Дефинираме фонов цвят на изображението и го съхраняваме в променлива $background_color.

$background_color = imagecolorallocate($im, 255, 255, 0); // yellow, 000-черен фон.

Когато се използва imagecreate() команда, първият дефиниран цвят се счита за background color на изображението. Ако се използва imagecreatetruecolor() команда вместо imagecreate команда, background color-а ще е black (но по нататък можем да нарисуваме квадрат с друг цвят и размери).

Функция Imagepng() — Извеждане или съхраняване на изображението в браузър или във файл

В пример1 имаше: imagepng($im,"image.png");

//изображението с идентификатор $im се съхранява в "image.png“.

С функция Imagepng (или сходните функции за другите поддържани формати imagegif(), imagewbmp(), imagejpeg() или imagetypes()), изображението може както да се съхрани във файл, така и да се изведе директно в браузъра . Тук сме използвали png заради неговата open source природа.

Още за Функция Imagepng() ...

Общият вид на imagepng() е следното:

bool **imagepng** (resource $image [, string $filename [, int $quality [, int $filters]]])

- Извежда изображението в браузъра или ако е подаден $filename - записва изображението във PNG файл. Функцията връща **TRUE** при успех, или **FALSE** – при неуспех. **Например:** imagepng($im,"image.png"); //изображението с идентификатор $im се съхранява в "image.png“.

Параметри:

$image – идентификатор на изображение, върнат от една от функциите, създаващи изображение, например imagecreatetruecolor().

$filename – път и име на файла в който се съхранява изображението. Ако не се зададе този параметър или е **NULL**, то потокът на изображението се извежда директно.

Забележка: **NULL** е невалидно, ако аргументи quality и filters не са зададени:quality – ниво на компресия: от 0 (няма компресия) до 9; filters– позволява да се редуцира размера на PNG файла (може да е всяка комбинация PNG_FILTER_XXX , както и **PNG_NO_FILTER** или **PNG_ALL_FILTERS** - за забрана или позволяване на всякакви филтри).

Imagedestroy() — Унищожаване на идентификатора на изображението

Общ вид:

bool **imagedestroy**(resource $image)

imagedestroy() освобождава свързаната с изображението памет на сървъра.

Параметри:

$image – идентификатор на изображение, върнат от една от функциите, създаващи изображение, например imagecreatetruecolor().

Връщана стойност:

Функцията връща **TRUE** при успех, или **FALSE** – при неуспех.

В пример1 имахме: imagedestroy($im);

- освобождава се свързаната с изображението (с идентификатор $im) памет на сървъра.

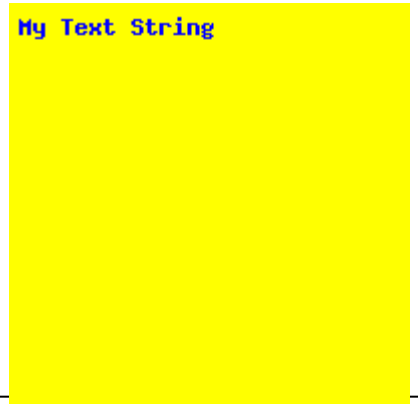
Още един метод за създаване на изображение

- Видяхме в пример 1, че изображението може да се запише във файл, към който може да се обърнем след това с команда print и обикновен img таг.
- Скриптът за създаване на изображението може също директно да се постави в <img> таг, например по следния начин:

****

При този метод изображението се включва inline в html, чрез таг за изображение. Вместо png, jpeg или gif, в scr тага включваме php скрипта, генериращ изображението.

Пример 1.2



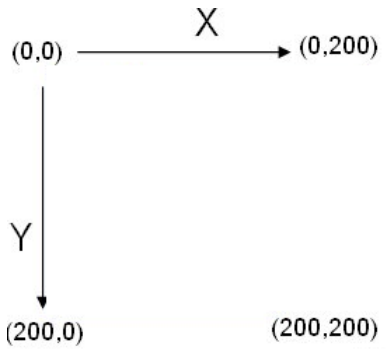
Index.php

```
<html>
   <head>
      <meta http-
     equiv="Content-Type"
     content="text/html;
     charset=UTF-8">
      <title></title>
   </head>
   <body>
   <img src="pic.php“>
   </body>
</html>
```

Pic.php

```
<?php
header("Content-type: image/png");
$im = @imagecreate(200, 200) or die("Cannot
     Initialize new GD image stream");
$background_color = imagecolorallocate($im,
     255, 255, 0);
    //цвят на фона – жълт
$blue = imagecolorallocate($im, 0, 0, 255);
     // цвят на текста - син
imagestring($im, 3, 5, 5,  "My Text String",
     $blue);
imagepng($im); //извеждане на
     изображението директно в браузъра
imagedestroy($im);
?>
```

Функция imagestring() – за добавяне на текст в изображението



- imagestring($im, 3, 5, 5, "My Text String", $blue);

Параметри:

- Идентификатор на изображението $im.
- int font – шрифт (тук 3), число от 1 – най-малък до 5 – най-голям, представляващо някой от множеството вградени шрифтове. Като алтернатива може да използвате TrueType или PostScript Type 1 шрифтове, които извеждат изгладен шрифт.
- Начална (X) и крайна координата (Y) за отпечатване (5,5)
- Самия текст: My Text String
- Цвят на текста: жълт

За “Извеждане на изображението директно в браузъра, т.е в движение”

Както казахме, с функция ImagePNG или сходна функция за другите поддържани формати изображението може да се съхрани във файл или **да се изведе директно в браузъра (както в разгледания пример)**.

Извеждането на изображението директно в браузъра (без съхраняване във файл) става на 2 етапа: първо: указваме на браузъра, че ще извеждаме изображение, а не текст или html файл. Това става с функция Header, с която определяме типа на изображението напр. png, и второ – извикване на функция imagepng(), подавайки идентификатора на изображението – напр. $im:

```
header('Content-Type: image/png');
imagepng ($im); //Изходният резултат се изпраща към браузъра в png формат
```

Забележка: Друго типично приложение на Header() е за HTTP пренасочване на браузъра към друга страница:

```
Header (Location: http://www.domain.com/new_home_page.html)
```

За “Извеждане на изображението директно в браузъра, т.е в движение”

- **header('Content-Type: image/png');**

Format	Content-Type
GIF	`image/gif`
JPEG	`image/jpeg`
PNG	`image/png`
WBMP	`image/vnd.wap.wbmp`

Още за Header()

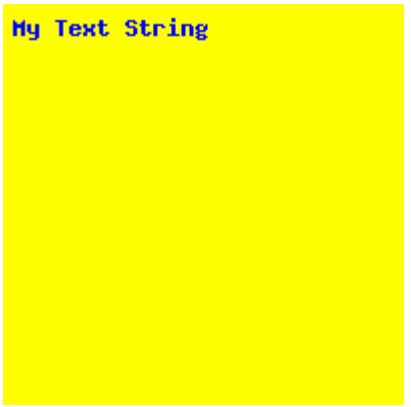
Важно е да се отбележи, че функция **Header()** **не може да бъде изпълнена**, ако за дадена страница е бил вече изпратен HTTP хедър (с echo например).

А такъв HTTP хедър автоматично се изпраща в момента, в който изведете нещо в браузъра.

Следователно, вие може да изпратите един или много HTTP хедъри за показваме на браузъра, че ще извеждаме изображение, но това трябва да стане преди изпращането на каквато и да е друга информация към браузъра.

Създаване и показване на изображения в движение (изображението не се съхранява на диска)

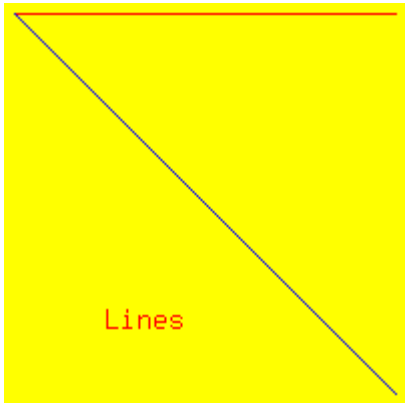
```
<?php
header("Content-type: image/png");
$im = @imagecreate(200, 200) or die("Cannot Initialize new GD
    image stream");
$background_color = imagecolorallocate($im, 255, 255, 0);    //
    yellow
$blue = imagecolorallocate($im, 0, 0, 255);   // blue
imagestring($im, 3, 5, 5,  "My Text String", $blue);
imagepng($im);
imagedestroy($im);
?>
```



- В този пример, първата информация, изпратена от сървъра към браузъра се показва в ред 2: информацията, изпращана от сървъра е PNG изображение.
- Създава се изображение 200x200
- Дефинират се използваните цветове в изображението
- Добавя се текст към изображението, изображението се извежда. Тъй като не е посочен файл, изображението е изведено ("в движение").

Пример 2: рисуване на линии – функция imageline()

```
<?php
    create_image();
    print "<img src=image.png?".date("U").">";
    function  create_image(){
    $im = @imagecreate(200, 200) or die("Cannot Initialize new GD image stream");
    $background_color = imagecolorallocate($im, 255, 255, 0);
// yellow за цвят на фона
// цветове, добавени към изображението
        $red = imagecolorallocate($im, 255, 0, 0);          // red
        $blue = imagecolorallocate($im, 0, 0, 255);          // blue
        imageline ($im,   5,  5, 195, 5, $red);
//хоризонтална линия с червен цвят
        imageline ($im,   5,  5, 195, 195, $blue);
//диагонал – със син.
imagestring($im,4,50,150,"Lines", $red); //Етикет
imagepng($im,"image.png");  imagedestroy($im);
    }
?>
```

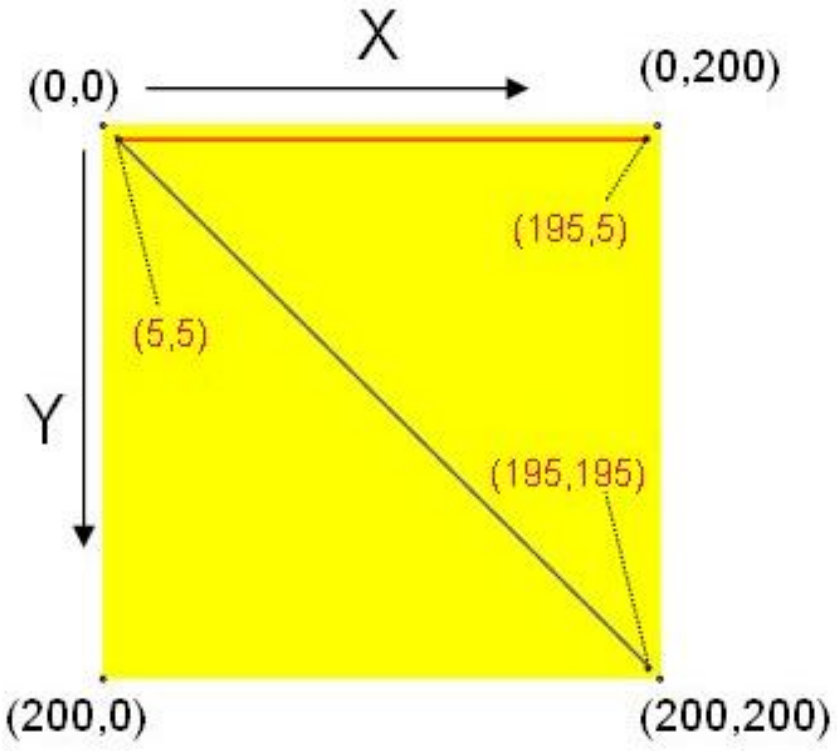


За да се рисува линия се използва команда imageline():

imageline ($im, X_1, Y_1, X_2, Y_2, $color);

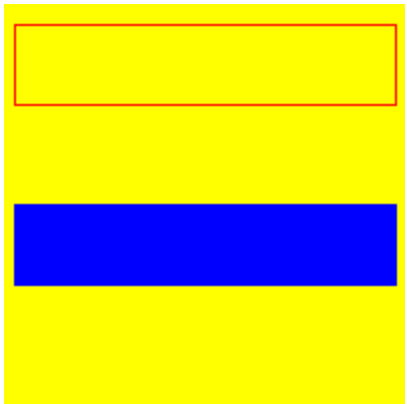
where X_1, Y_1, X_2 and Y_2 are the positions within the image.

For example, the red line starts at
X_1=5 and Y_1=5
and ends at
X_2=195 and Y_2=5



Пример 3: рисуване на правоъгълник – imagerectangle() и imagefilledrectangle()

```
<?php
    create_image();
    print "<img src=image.png?".date("U").">";

    function  create_image()
{ $im = @imagecreate(200, 200) or die("Cannot Initialize new GD image
    stream");
  $background_color = imagecolorallocate($im, 255, 255, 0);   // yellow
  $red = imagecolorallocate($im, 255, 0, 0);                  // red
  $blue = imagecolorallocate($im, 0, 0, 255);                  // blue
 imagerectangle ($im,   5,  10, 195, 50, $red);                //правоъгълника
 imagefilledrectangle ($im,   5,  100, 195, 140, $blue);
//запълнен правоъгълник
 imagedestroy($im);
    }
?>
```



За да се нарисува правоъгълник imagerectangle(), трябва да се зададат позициите на горния ляв и долния десен ъгли:

imagerectangle ($im, X_1, Y_1, X_2, Y_2, $color);

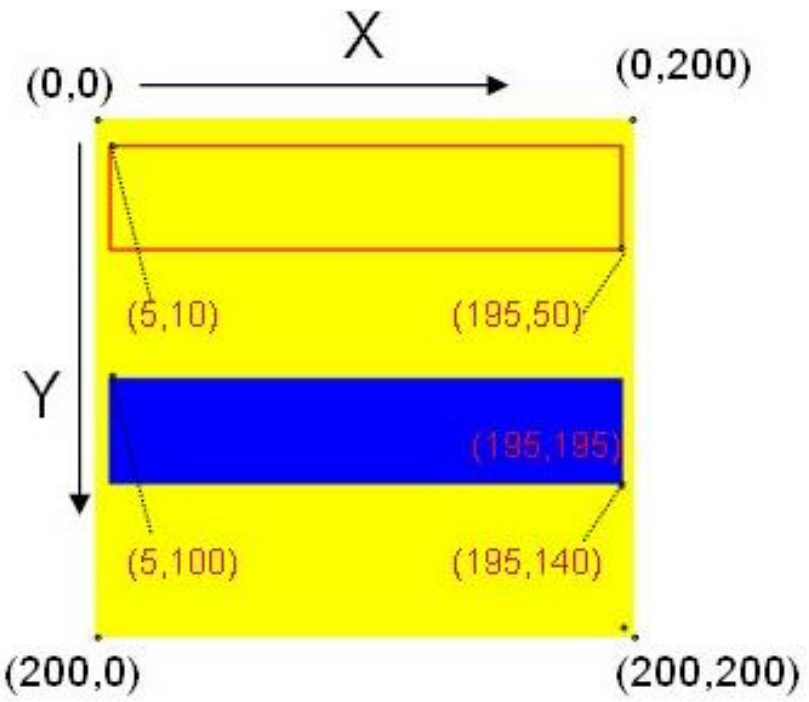
imagefilledectangle ($im, X_1, Y_1, X_2, Y_2, $color);

For example, the red rectangle starts at

X_1=5 and Y_1=10

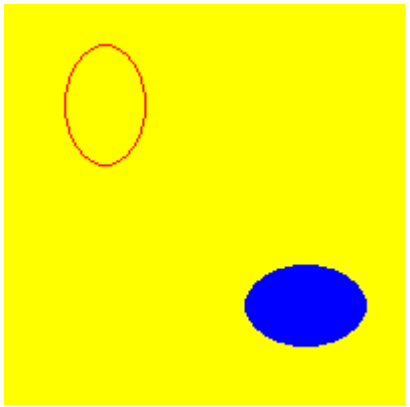
and ends at

X_2=195 and Y_2=50



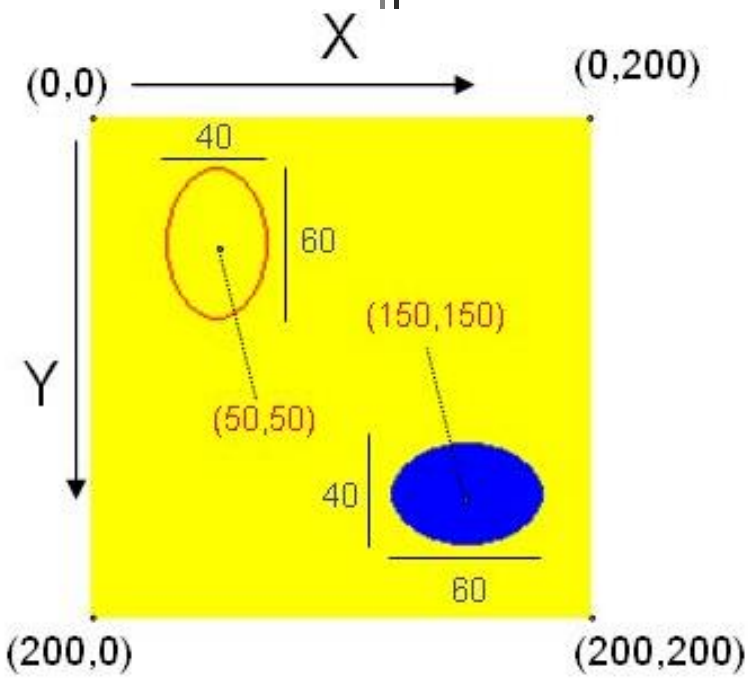
Пример 4: Рисуване на елипси imageellipse() и imagefilledellipse()

```
<?php
    create_image();
    print "<img src=image.png?".date("U").">";

    function  create_image(){
        $im = @imagecreate(200, 200) or die("Cannot Initialize new GD image
    stream");
        $background_color = imagecolorallocate($im, 255, 255, 0);   // yellow

        $red = imagecolorallocate($im, 255, 0, 0);              // red
        $blue = imagecolorallocate($im, 0, 0, 255);              // blue
        imageellipse($im, 50, 50, 40, 60, $red); //празна елипса
        imagefilledellipse($im, 150, 150, 60, 40, $blue); //запълнена елипса

        imagepng($im,"image.png");
        imagedestroy($im);
    }
?>
```



- За да се нарисуват елипси се задават позициите на центровете им, както и техните ширина и височина:

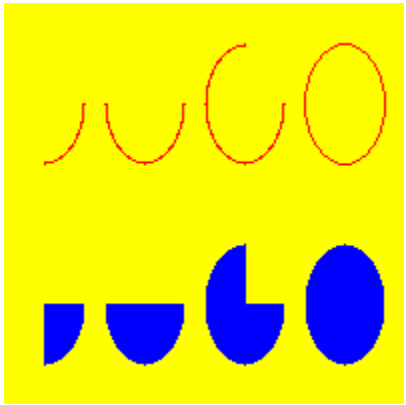
imageellipse($im, X, Y, width, height, $color);
imagefilledellipse($im, X, Y, width, height,$color);
For example, the center of the red ellipse is located at
X=50
Y=50
and the dimensions are
width = 40
height= 60



- За да се нарисува кръг - начертайте елипса със ширина=височина.

Пример 5: Рисуване на дъги (arcs) – imagearc() и imagefilledarc()

```
<?php
    create_image();
    print "<img src=image.png?".date("U").">";

    function  create_image(){
        $im = @imagecreate(200, 200) or die("Cannot Initialize new GD image stream");
        $background_color = imagecolorallocate($im, 255, 255, 0);   // yellow

        $red = imagecolorallocate($im, 255, 0, 0);              // red
        $blue = imagecolorallocate($im, 0, 0, 255);              // blue
        //a red arc is drawn (only the outer line)..
        imagearc($im, 20, 50, 40, 60, 0, 90, $red);
        imagearc($im, 70, 50, 40, 60, 0, 180, $red);
        imagearc($im, 120, 50, 40, 60, 0, 270, $red);
        imagearc($im, 170, 50, 40, 60, 0, 360, $red);
         //a blue arc is drawn (a filled one).
        imagefilledarc($im, 20, 150, 40, 60, 0, 90, $blue, IMG_ARC_PIE);
        imagefilledarc($im, 70, 150, 40, 60, 0, 180, $blue, IMG_ARC_PIE);
        imagefilledarc($im, 120, 150, 40, 60, 0, 270, $blue, IMG_ARC_PIE);
        imagefilledarc($im, 170, 150, 40, 60, 0, 360, $blue, IMG_ARC_PIE);

        imagepng($im,"image.png");
        imagedestroy($im);
    }
    ?>
```



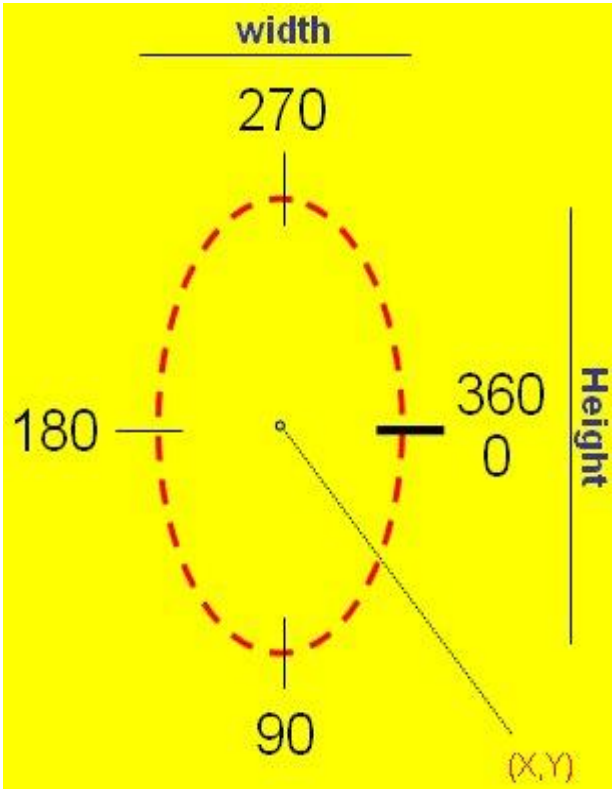
Коментар

- Базовите команди:
- **imagearc** ($im, X, Y, width, height, arc start, arc end, $color)
- **imagefilledarc** ($im, X, Y, width, height, arc start,

arc end, $color, flag)

X и Y дефинират позициите на центъра на дъгата в изображението
Width и height са размерите на пълен кръг, формиран от дъгата; **arc start** и **arc end** са началото и края на дъгата в градуси.

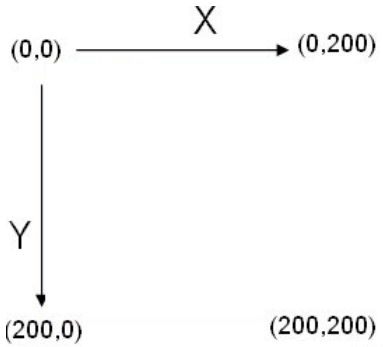
- Ако **arc start** е 0, както в примера, фигурата ще стартира в дясно.
Flags са специфични опции за тези команди, върху които няма да се спираме тук.
Фигурата в дясно ще ви помогне да разберете базовите команди за дъги:



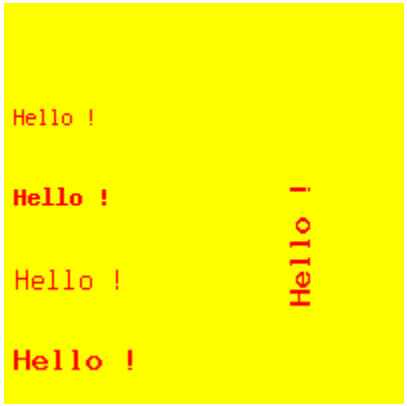
Още:

- Команди **imagearc** () и **imagefilledarc** () могат също да бъдат използвани за рисуване на кръгове (width=height, arc start=0 и arc end=360).
- Други фигури, които могат да се рисуват:
  imagepolygon () -- Рисува polygon
  imagefilledpolygon () -- Рисува запълнен polygon

Пример 6: Още...за добавяне на текст към изображението




```
<?php
    create_image();
    print "<img src=image.png?".date("U").">";

    function  create_image(){
        $im = @imagecreate(200, 200)or die("Cannot Initialize new GD image stream");
        $background_color = imagecolorallocate($im, 255, 255, 0);  // yellow
        $red = imagecolorallocate($im, 255, 0, 0);      // red
    // adds text to image  - text is written in horizontal
        imagestring($im, 2, 5,  50, "Hello !", $red);
        imagestring($im, 3, 5,  90, "Hello !", $red);
        imagestring($im, 4, 5, 130, "Hello !", $red);
        imagestring($im, 5, 5, 170, "Hello !", $red);
    // text is written in vertical
        imagestringup($im, 5, 140, 150, "Hello !", $red);

        imagepng($im,"image.png");
        imagedestroy($im);
    }
?>
```

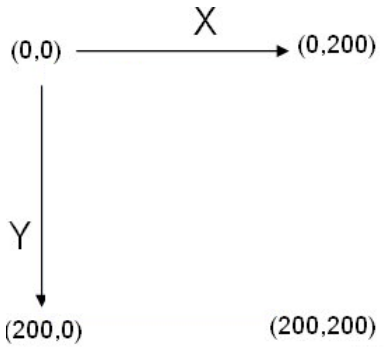


Коментар:

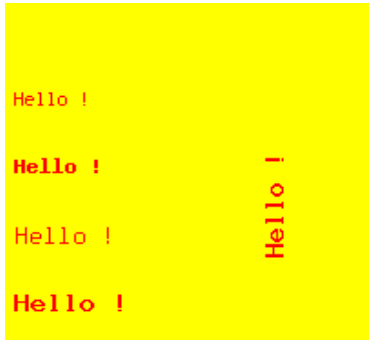
- Новото в този пример са използваните вградени шрифтове (built-in fonts) за извеждане на вертикален текст:

- imagestringup ($im, size, X, Y, text, $red);

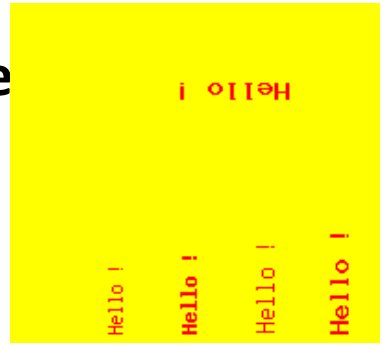
- Размерът (size) е от 1 до 5.
  X и Y са позициите на текста (там от където стартира).



Ротация на изображение




```
<?php
    $im = imagecreatefrompng("image.png");
// the information about the image is stored in variable $im.
    $yellow = imagecolorallocate($im, 255, 255, 0);
// a new color is used to  be use in next line
    $rotate = imagerotate($im, 90,$yellow);
/* the information about the image in $im is modified
    with imagerotate () and stored in a new variable: $rotate   */
imagepng($rotate,"image_rotated.png");
    imagedestroy($im);
    print "<img src=image.png> - <img src=image_rotate
    //визуализират се 2-те изображения
?>
```



Коментар

- Информацията за едно изображение е съхранена в променливата $im. В този случай, тази информация е получена от едно съществуващо png изображение, чрез командата imagecreatefrompng().
- Ако изображението е с друго разширение, то се използват други команди:
  - imagecreatefromgif -- Create a new image from file or URL
  - imagecreatefromjpeg -- Create a new image from file or URL
  - imagecreatefrompng -- Create a new image from file or URL
  - imagecreatefromwbmp -- Create a new image from file or URL
  - imagecreatefromxbm -- Create a new image from file or URL
  - imagecreatefromxpm -- Create a new image from file or URL

Имаме: $rotate = imagerotate($im, 90,$yellow);

Информацията за изображението в $im е модифицирана с imagerotate () и съхранена в една нова променлива $rotate:

Общ вид на imagerotate :

imagerotate ($im, degrees, $color);

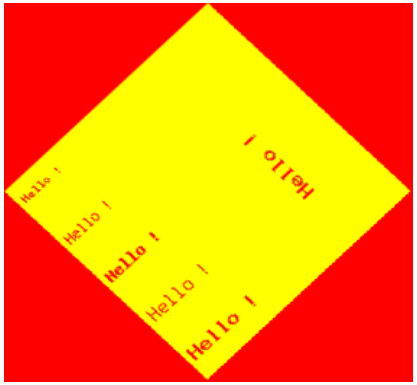
degrees – дефинират ъгъла на ротация. Стойностите са положителни (90, 180...) или отрицателни (-90,-180...).

$color – цвета, използван като background, когато е необходимо (това е когато ъгълът на ротация е различен от 90, 180 и 270). Например, когато ъгълът на ротация е 45 градуса, размерите на фигурата ще бъдат променени, както е показано по-долу. В резултат на това се изисква фонов цвят за да запълни новата фигура.

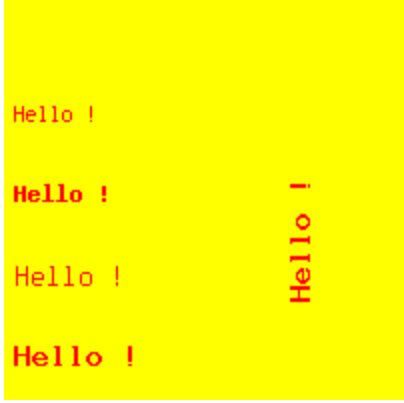
В примера долу, ние дефинираме фонов цвят=red:

Изображението се съхранява във файл и се освобождава паметта.

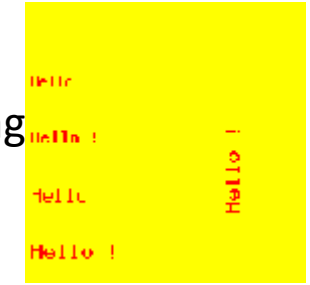
И двете изображения: оригиналът и новото са показани в резултат.



Пре-оразмеряване (Resize) на изображение




```
<?php
    $original_image = imagecreatefrompng("image.png");
    // създаване на изображение от съществуващо
    $image_info = getimagesize("image.png");
    // създава се масив $image_info съдържащ информация за размера на изображението
    // print_r($image_info)
    $width  = $image_info[0]; // ширина на изображенето
    $height = $image_info[1];    // височина на изображението
    // ще редуцираме размера до 70%, така че новите размери трябва да се изчислят
    $new_width  = round ($width*0.7);
    $new_height = round ($height*0.7);
    $new_image = imagecreate($new_width, $new_height);
    imagecopyresized($new_image, $original_image, 0, 0, 0, 0, $new_width, $new_height, $width, $height);
    imagepng($new_image,"resized_image.png");
    imagedestroy($new_image);
    print "<img src=image.png> <br>Resized image<BR> <img src=resized_imag
    ?>
```

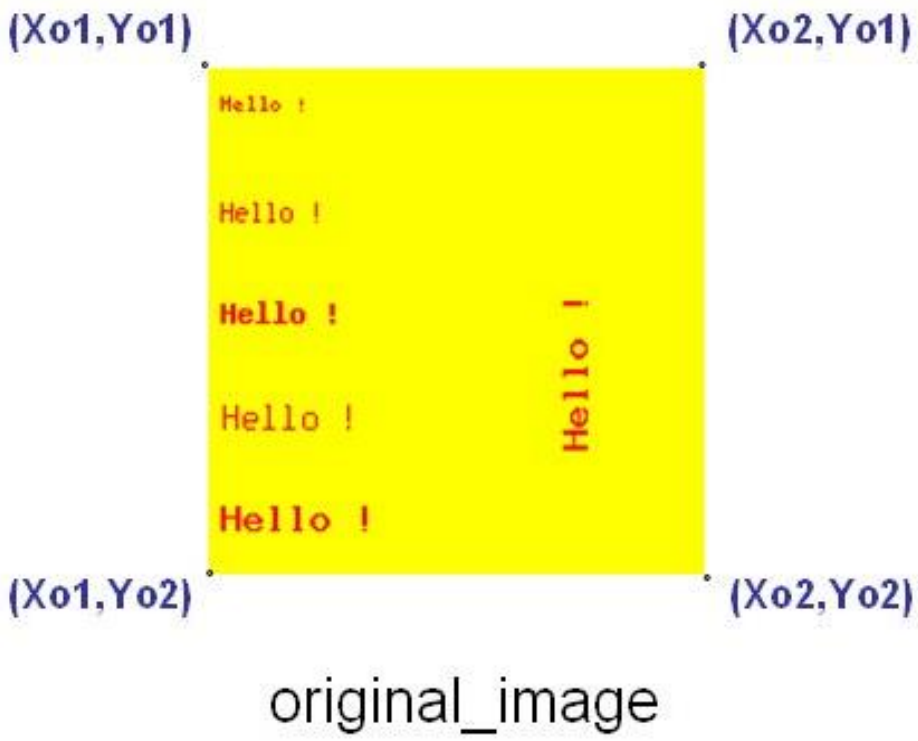


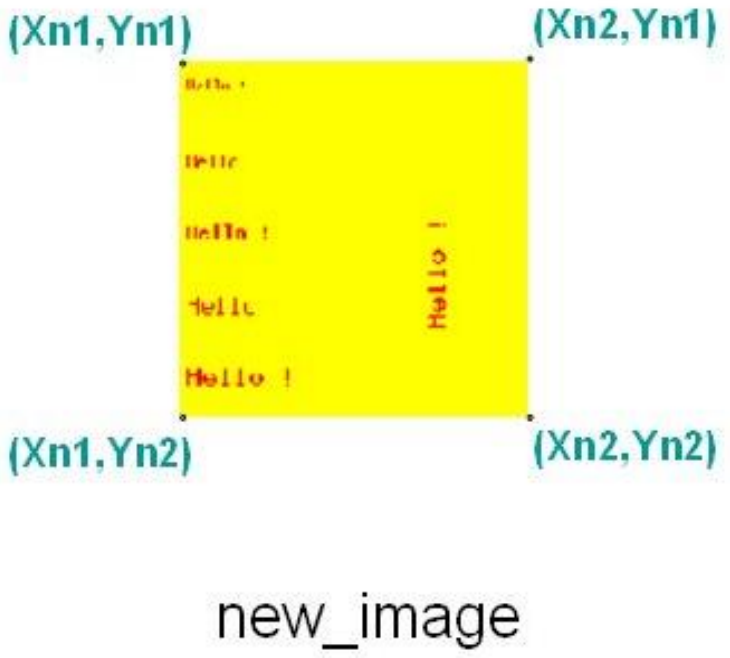
- Информацията за оригиналното изображение се съхранява във променлива $original_image.
- Размерите (size) на оригиналното изображение се извличат: в един масив $image_info .
- width и height на изображението се извличат от масива $image_info и се съхраняват в променливи $width и $height.
- В този пример, изображението се редуцира до 70%, така, че нови width и height се изчисляват. Използва се round за закръгляване.
- Стартира се създаването на новото изображение: Команда imagecreate() ще дефинира размерите на новото изображение и променлива $new_image съхранява цялата изформация за него.
- imagecopyresized - Това е ключовата команда в този скрипт.

imagecopyresized($new_image, $original_image, 0, 0, 0, 0, $new_width, $new_height, $width, $height);

- Следващата картинка е опит да се изясни смисъла на командата:

```
imagecopyresized ($new_image, $original_image, Xn1, Yn1, Xo1, Yo1, Xn2, Yn2, Xo2, Yo2);
```





Тук:

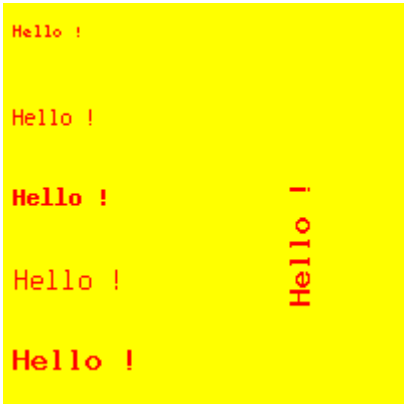
Xo1, Yo1, Xo2 и Yo2 са димeнсиите на оригиналното изображение

Xn1, Yn1, Xn2 и Yn2 са димeнсиите на новото изображение

Следва съхраняване на изображението (новото) и освобождаване на заетата от него памет.

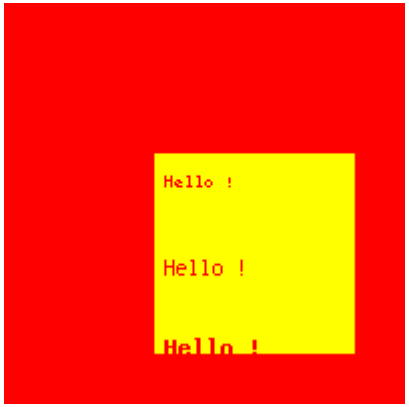
Двете изображения се визуализират (оригиналното и новото).

Пример 9: Вземане на част от изображение



- <?php

```
// get the original image
$original_image = imagecreatefrompng("image.png");

// create new image
$new_image = imagecreate(200, 200);

// define red color (it will be the background of the new image)
$red = imagecolorallocate($new_image, 255, 0, 0);

imagecopyresized($new_image, $original_image, 75, 75, 0, 0, 100, 100, 100, 100);

imagepng($new_image,"new_image.png");
imagedestroy($new_image);

print "<img src=image.png> <br>New image<BR> <img src=new_image.png>";

?>
```

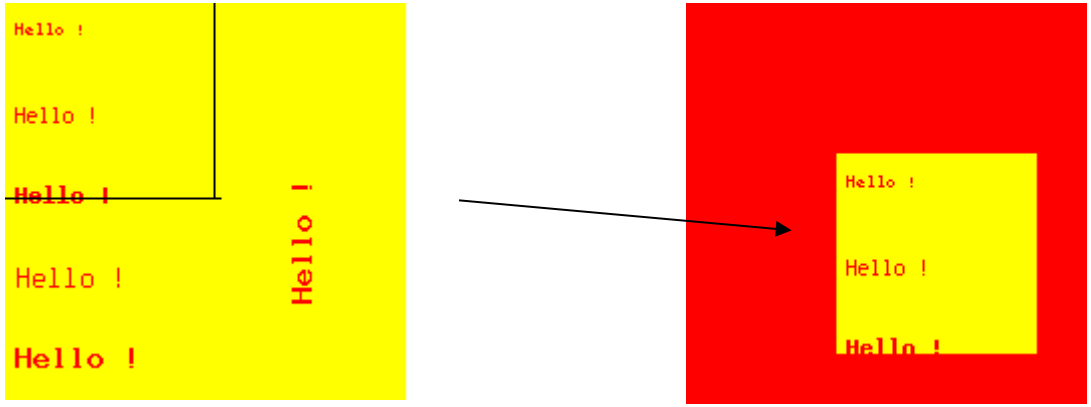


Коментар

- Информацията за оригиналното изображение се съхранява в променлива $original_image.
- Ново 200x200 pixels изображение се създава.
- Дефинира се цвят за новото изображение. Той ще бъде също и background color (както в пример 7)
- Ключова команда в скрипта:

imagecopyresized($new_image, $original_image, 75, 75, 0, 0, 100, 100, 100, 100);

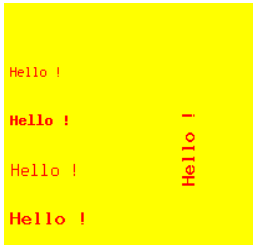
- Квадрат (част) от оригиналното изображение се селектира. За този пример селектираната част е с позиции: (0,0) и (100,100).

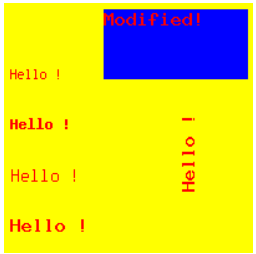


- Тази селектирана част се копира на позиции (75,75) на новото изображение и размерите на частта се поддържат да са (100x100).
- Новото изображение се съхранява в един файл и се освобождава заетата от него памет.
- Извеждат се и двете изображение: оригиналното и новото.

Пример 10: модифициране на изображение. Един правоъгълник и текст ще бъдат добавени към едно вече съществуващо изображение

```
<?php
    // get the original image
    $im = imagecreatefrompng("image.png");
    // new color
    $blue = imagecolorallocate($im, 0, 0, 255);  // blue
    $red = imagecolorallocate($im, 255, 0, 0);     // red
    imagefilledrectangle ($im,   80,  5, 195, 60, $blue);
    imagestring($im, 5, 80, 5, "Modified!", $red);
    imagepng($im,"modified_image.png");
    imagedestroy($im);
print "<img src=image.png> <br>Modified image<BR> <img
    src=modified_image.png>“;
?>
```





Коментар

- Ново изображение, използвайки едно съществуващо такова:

$im = imagecreatefrompng("image.png");

- Нови цветове се дефинират
- Един правоъгълник се добавя към изображението (както при пример 3)
- Текст се добавя към изображение (както при пример 6)
- Новото изображение се съхранява в един файл и се освобождава заетата от него памет (както се описва в пример 1).
- Двете изображения се визуализират (оригиналното и новото).

Литература

- http://phptutorial.info/learn/create_images/#code1
- http://www.php.net/manual/en/function.imagejpeg.php
- https://www.ibm.com/developerworks/library/os-objorient/
- http://php.happycodings.com/Graphics/
- http://www.alphadevx.com/a/56-Basic-Shapes-in-PHP-GD
- http://docstore.mik.ua/orelly/webprog/php/ch09_01.htm