




# Trabalhando com imagens no PHP

Palestra para o PHP Conference Brasil 2009

Fabio Fogliarini Brolesi

[fabio@freesandbox.net](mailto:fabio@freesandbox.net)

# Quem sou eu?

- Programador PHP
- PHPConf 2008 The logo for GeomeTriKo, featuring the text 'GeomeTriKo' in a colorful, sans-serif font. The 'G' is red, 'e' is green, 'o' is blue, 'm' is yellow, 'e' is green, 'T' is red, 'r' is blue, 'i' is green, 'K' is red, and 'o' is blue. The text is set against a yellow, triangular background.
- Campus Party 2009
- Formado na UNICAMP

# PHP

- Linguagem interpretada, lembra C, Java, Perl
- Livre (open source)
- Largamente utilizado
- Suporte a uma vasta gama de banco de dados
- Focado nos scripts do lado do servidor, mas pode-se fazer muito mais com ele.
- PHP5: forte introdução de OOP

# Trabalhar com imagens...

- O que precisa (dependências)?
  - GD (formatos png, gif, jpeg)
  - Free Type (bibliotecas de fontes)
- Instalação
  - `--with-gd[=DIR] , --with-jpeg-dir=DIR --with-freetype-dir=DIR`

# Como fazer o básico?

- Criando o canvas
- `$img = imagecreate($l, $a);`
- `$img = imagecreatetruecolor($l, $a);`
  
- Criando a paleta de cores
- `imagecolorallocate ($img, $red, $green, $blue);`

# Texto sobre imagem

```
1. <?php
2. $img = imagecreate(100, 100);
3. $branco = imagecolorallocate($img, 128,
   128, 128);
4. $corTexto = imagecolorallocate($img, 255,
   255, 255);
5. imagestring($img, 5, 0, 45, 'Hello
   world!', $corTexto);
6. header('Content-type: image/png');
7. imagepng($img);
8. imagedestroy($img);
9. ?>
```

# Texto sobre imagem



# Texto sobre imagem

```
1. <?php
2. $im = imagecreatetruecolor(300, 100);
3. $red = imagecolorallocate($im, 0xFF, 0x00,
   0x00);
4. $black = imagecolorallocate($im, 0x00,
   0x00, 0x00);
5. $fontFile = './arial.ttf';
6. imagefttext($im, 24, 0, 80, 55, $red,
   $fontFile, 'Hello World');
7. imagepng($im, 'hello.png');
8. imagedestroy($im);
9. ?>
```



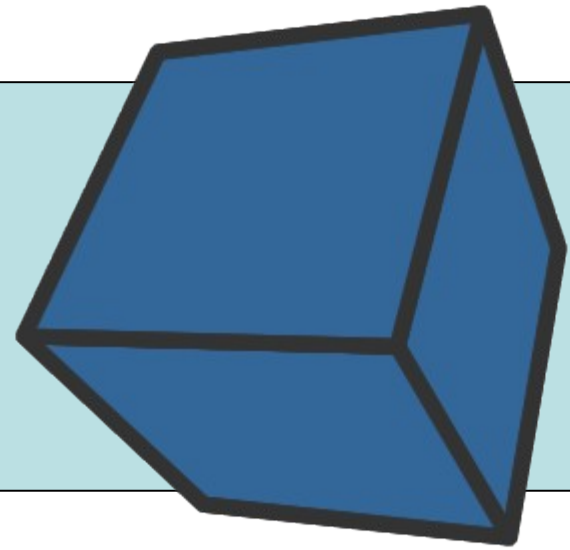
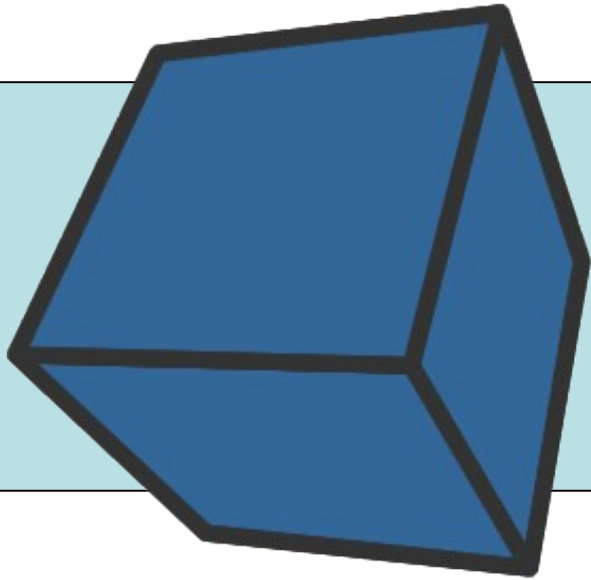
# Texto sobre imagem

Hello World

# Imagens sobre imagens

1. `<?php`
2. `$im =  
imagecreatefrompng('cube.png');`
3. `imagealphablending($im, true);`
4. `imagesavealpha($im, true);`
5. `header('Content-type: image/png');`
6. `imagepng($im);`
7. `imagedestroy($im);`
8. `?>`

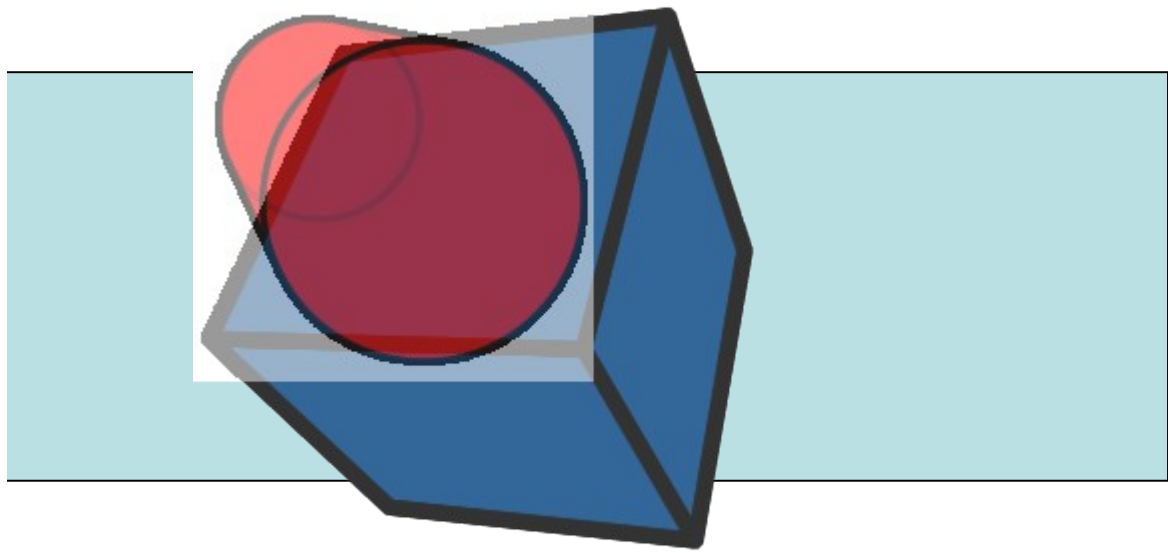
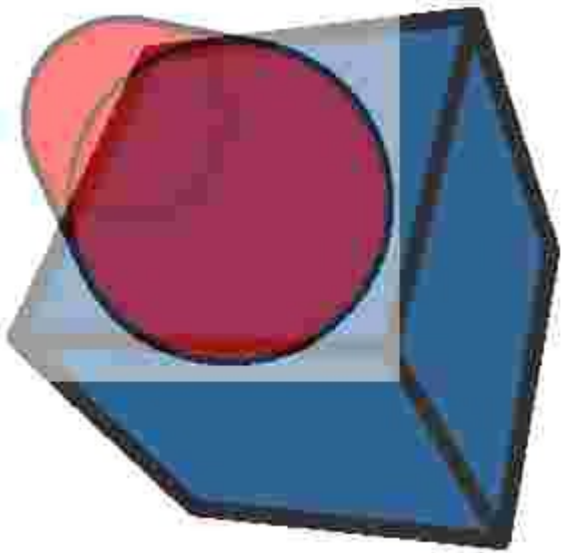
# Imagem sobre imagem



# Imagem sobre imagem

```
1.  <?php
2.  $stamp = imagecreatefrompng('cylinder.png');
3.  $im = imagecreatefrompng('cube.png');
4.  imagealphablending($im, true);
5.  imagesavealpha($im, true);
6.  imagealphablending($stamp, true);
7.  imagesavealpha($stamp, true);
8.  $w = imagesy($stamp);
9.  $h = imagesx($stamp);
10. imagecopymerge($im, $stamp, 0, 0, 0, 0, $h, $w, 50);
11. header('Content-type: image/png');
12. imagepng($im);
13. imagedestroy($im);
14. ?>
```

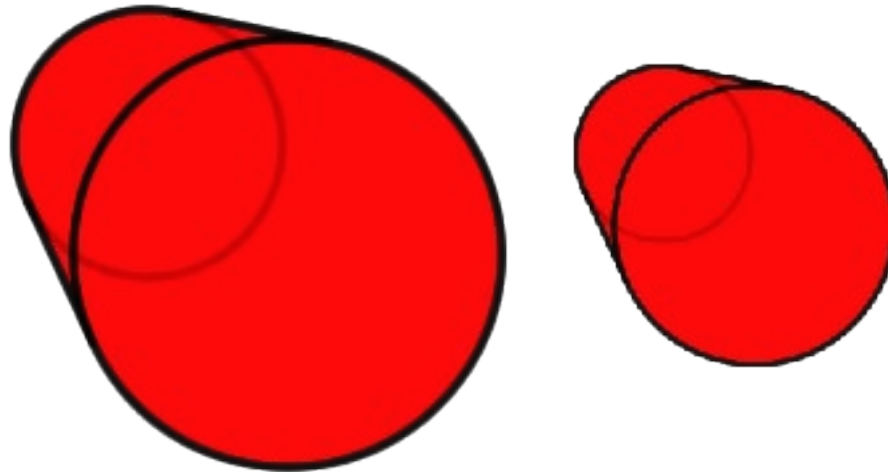
# Imagem sobre imagem



# Thumbnails

```
1.  <?php
2.  $ratio = 2/3;
3.  $original = 'cylinder.png';
4.  list($width, $height) = getimagesize($original);
5.  $nWidth = $width * $ratio;
6.  $nHeight = $height * $ratio;
7.  $thumb = imagecreate($nWidth, $nHeight);
8.  imagesavealpha($thumb, true);
9.  imagecolorexactalpha($thumb, 0, 0, 0, 1);
10. $image = imagecreatefrompng($original);
11. imagecopyresampled($thumb, $image, 0, 0, 0, 0,
    $nWidth, $nHeight, $width, $height);
12. header('Content-type: image/png');
13. imagepng($thumb);
14. imagedestroy($thumb)
15. ?>
```

# Thumbnails



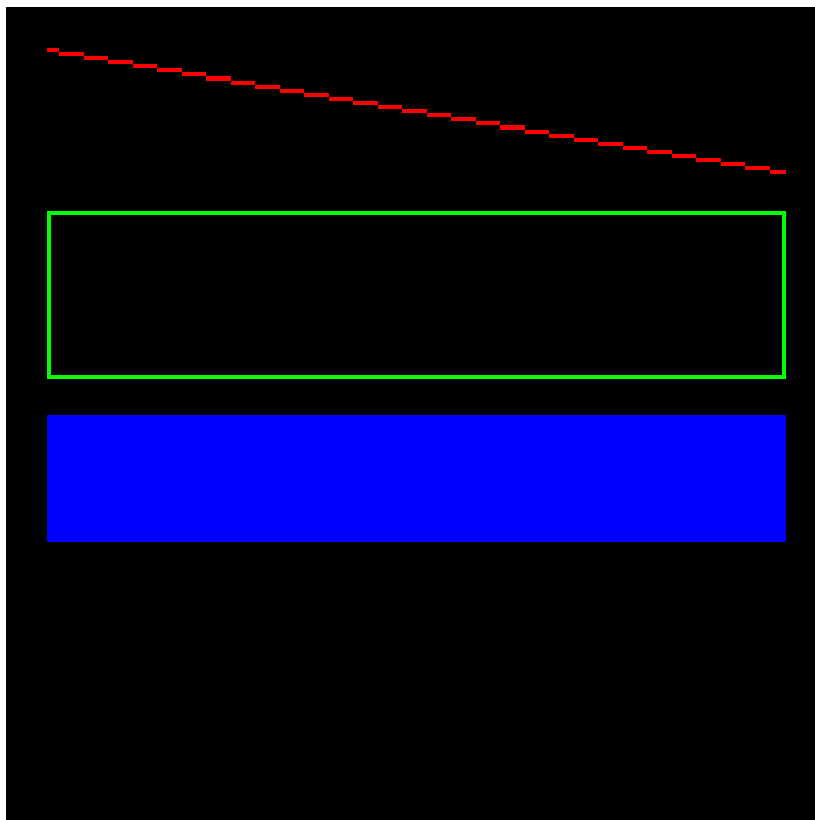
# Linhas, retângulos

- `bool imageline(resource $image, int $x1, int $y1, int $x2, int $y2, int $color);`
- `bool imagerectangle(resource $image, int $x1, int $y1, int $x2, int $y2, int $color)`
- `bool imagefilledrectangle(resource $image, int $x1, int $y1, int $x2, int $y2, int $color)`

# Linhas e retângulos

```
1. <?php
2. $img = imagecreate(200, 200);
3. $bl = imagecolorallocate($img, 0, 0, 0);
4. $wh = imagecolorallocate($img, 0xFF, 0xFF, 0xFF);
5. $r  = imagecolorallocate($img, 0xFF, 0, 0);
6. $g  = imagecolorallocate($img, 0, 0xFF, 0);
7. $b  = imagecolorallocate($img, 0, 0, 0xFF);
8. imageline($img, 10, 10, 190, 40, $r);
9. imagerectangle($img, 10, 50, 190, 90, $g);
10. imagefilledrectangle($img, 10, 100, 190, 130, $b);
11. header('Content-type: image/png');
12. imagepng($img);
13. imagedestroy($img);
14. ?>
```

# Linhas e retângulos



# Linhas, retângulos

```
1. function line($img, $x0, $y0, $x1, $y1, $cor, $box = false, $filled =  
false)  
2. {  
3.     if($box == false)  
4.     {  
5.         imageline($img, $x0, $y0, $x1, $y1, $cor);  
6.     }  
7.     else  
8.     {  
9.         if($filled == false)  
10.        {  
11.            imagerectangle($img, $x0, $y0, $x1, $y1, $cor);  
12.        }  
13.        else  
14.        {  
15.            imagefilledrectangle($img, $x0, $y0, $x1, $y1, $cor);  
16.        }  
17.    }  
18. }
```

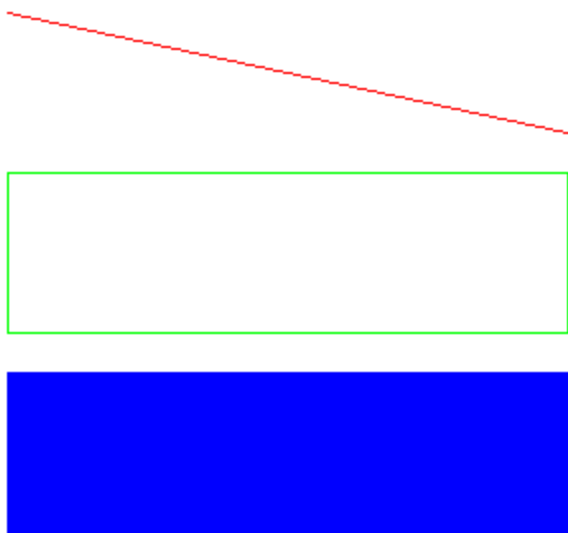
# Linhas, retângulos

```
1. $i = imagecreate(300, 300);  
2. $w = imagecolorallocate($i, 255, 255, 255);  
3. $r = imagecolorallocate($i, 255, 0, 0);  
4. $g = imagecolorallocate($i, 0, 255, 0);  
5. $b = imagecolorallocate($i, 0, 0, 255);
```

```
1. line($i, 10, 20, 290, 80, $r);  
2. line($i, 10, 100, 290, 180, $g, true);  
3. line($i, 10, 200, 290, 280, $b, true, true);
```

```
1. header('content-type:image/png');  
2. imagepng($i);
```

# Linhas, retângulos



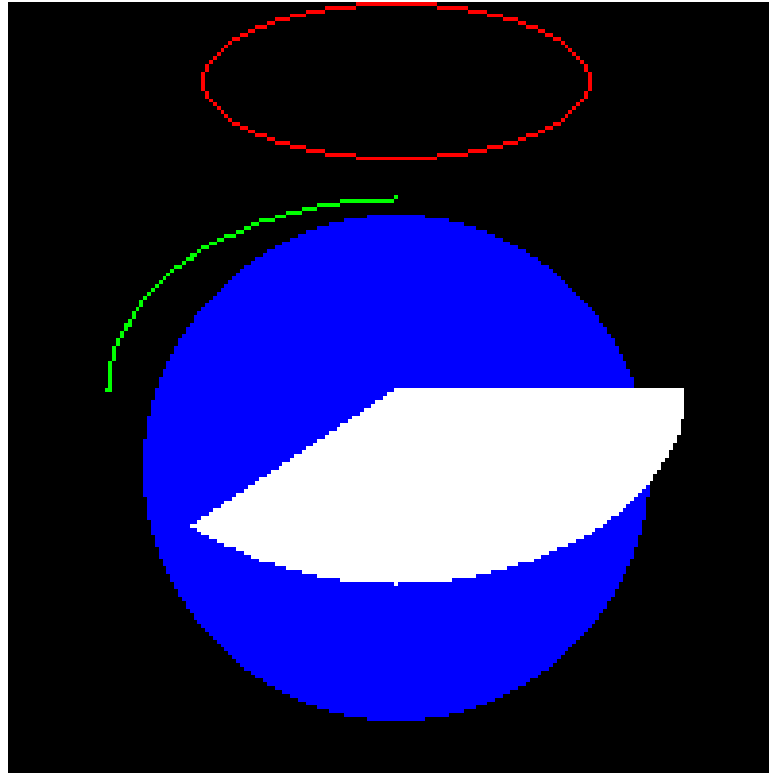
# Arcos, elipses

- `bool imagearc(resource $image, int $cx, int $cy, int $width, int $height, int $start, int $end, int $color )`
- `bool imagefilledarc(resource $image, int $cx, int $cy, int $width, int $height, int $start, int $end, int $color, int $style)`

# Arcos, elipses

```
1. <?php
2. $img = imagecreate(200, 200);
3. $bl = imagecolorallocate($img, 0, 0, 0);
4. $wh = imagecolorallocate($img, 0xFF, 0xFF, 0xFF);
5. $r  = imagecolorallocate($img, 0xFF, 0, 0);
6. $g  = imagecolorallocate($img, 0, 0xFF, 0);
7. $b  = imagecolorallocate($img, 0, 0, 0xFF);
8. imageellipse($img, 100, 20, 100, 40, $r);
9. imagefilledellipse($img, 100, 120, 130, 130, $b);
10. imagearc($img, 100, 100, 150, 100, 180, 270, $g);
11. imagefilledarc($img, 100, 100, 150, 100, 0, 135, $wh,
    IMG_ARC_EDGED);
12. header('Content-type: image/png');
13. imagepng($img);
14. imagedestroy($img);
15. ?>
```

# Arcos, elipses



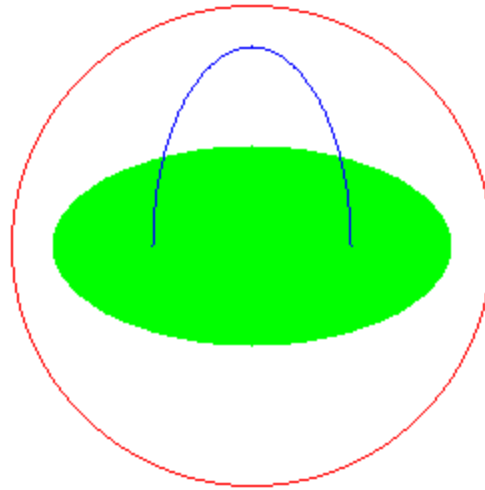
# Arcos, elipses

```
1. function circle($img, $x, $y, $r, $cor, $p = false, $e = 1, $r0 = 0, $rf = 360)
2. {
3.     $rx = $r;
4.     $ry = $r;
5.     if($e < 1)
6.     {
7.         $ry = $r * $e;
8.     }
9.     else
10.    {
11.        $rx = $r / $e;
12.    }
13.    if($p == false)
14.    {
15.        imagearc($img, $x, $y, $rx * 2, $ry * 2, -$rf, -$r0, $cor);
16.    }
17.    if($p == true)
18.    {
19.        imagefilledarc($img, $x, $y, $rx * 2, $ry * 2, -$rf, -$r0, $cor,
20. IMG_ARC_EDGED);
21.    }
```

# Arcos, elipses

```
1. $i = imagecreate(300, 300);
2. $w = imagecolorallocate($i, 255, 255, 255);
3. $r = imagecolorallocate($i, 255, 0, 0);
4. $g = imagecolorallocate($i, 0, 255, 0);
5. $b = imagecolorallocate($i, 0, 0, 255);
6. circle($i, 150, 150, 120, $r);
7. circle($i, 150, 150, 100, $g, true, .5);
8. circle($i, 150, 150, 100, $b, false, 2, 0, 180);
9. header('content-type:image/png');
10. imagepng($i);
```

# Arcos, elipses



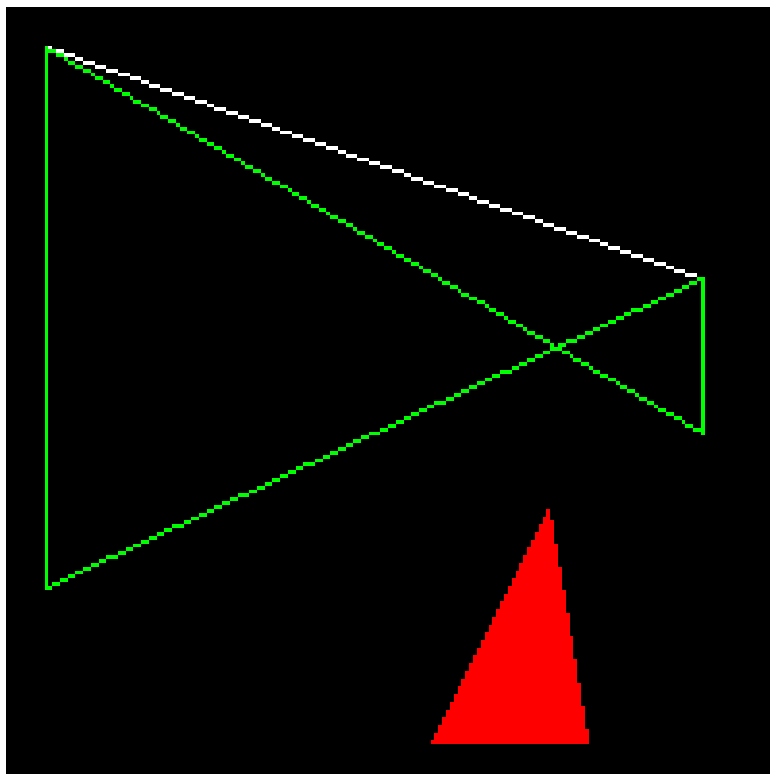
# Polígonos regulares

- `bool imagefilledpolygon ( resource $image, array $points, int $num_points, int $color )`
- `bool imagepolygon ( resource $image, array $points, int $num_points, int $color )`

# Polígonos regulares

```
1. <?php
2. $img = imagecreate(200, 200);
3. $bl = imagecolorallocate($img, 0, 0, 0);
4. $wh = imagecolorallocate($img, 0xFF, 0xFF, 0xFF);
5. $r  = imagecolorallocate($img, 0xFF, 0, 0);
6. $g  = imagecolorallocate($img, 0, 0xFF, 0);
7. $b  = imagecolorallocate($img, 0, 0, 0xFF);
8. $t1 = array(10, 10, 10, 150, 180, 70, 180, 110);
9. $t2 = array(150, 190, 140, 130, 110, 190);
10. imagepolygon ($img, $t1, 3, $wh);
11. imagepolygon ($img, $t1, 4, $g);
12. imagefilledpolygon($img, $t2, 3, $r);
13. header('Content-type: image/png');
14. imagepng($img);
15. imagedestroy($img);
16. ?>
```

# Polígonos regulares



# Polígonos regulares

```
1. function regularPolygonI($img, $x0, $y0, $lados, $r, $cor, $beta = 0, $p = false)
2. {
3.     $theta = 360 / $lados;
4.     $alpha = (180 - $theta) / 2;
5.     $a      = $r * (sin(deg2rad($theta)) / sin(deg2rad($alpha)));

1.     for($i = 0; $i < $lados; $i++)
2.     {
3.         $gamma = $alpha - $beta;
4.         $ponto[$i]['x'] = $x0 + $r * cos(deg2rad($gamma));
5.         $ponto[$i]['y'] = $y0 + $r * sin(deg2rad($gamma));

1.         $pontos[] = $ponto[$i]['x'];
2.         $pontos[] = $ponto[$i]['y'];
3.
4.         $beta += $theta;
5.     }
6.     if (!$p)
7.     {
8.         imagepolygon($img, $pontos, $lados, $cor);
9.     }
10.    else
11.    {
12.        imagefilledpolygon($img, $pontos, $lados, $cor);
13.    }
14. }
```

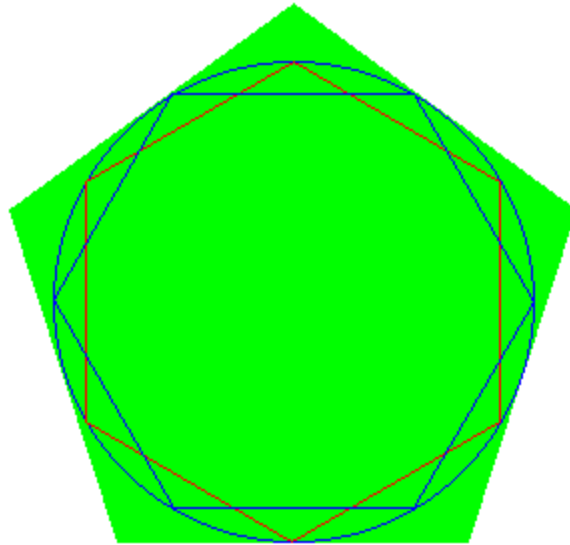
# Polígonos regulares

```
1. function regularPolygonC($img, $x0, $y0,  
   $lados, $r, $cor, $beta = 0, $p = false)  
2. {  
3.     $theta = 360 / $lados;  
4.     $alpha = (180 - $theta) / 2;  
5.     $r      = $r / sin(deg2rad($alpha));  
6.     regularPolygonI($img, $x0, $y0,  
   $lados, $r, $cor, $beta, $p);  
7. }
```

# Polígonos regulares

- `$i = imagecreate(300, 300);`
- `$w = imagecolorallocate($i, 255, 255, 255);`
- `$r = imagecolorallocate($i, 255, 0, 0);`
- `$g = imagecolorallocate($i, 0, 255, 0);`
- `$b = imagecolorallocate($i, 0, 0, 255);`
- `regularPolygonC($i, 150, 150, 5, 120, $g, 0, true);`
- `circle($i, 150, 150, 120, $b);`
- `regularPolygonI($i, 150, 150, 6, 120, $r, 30, false);`
- `regularPolygonI($i, 150, 150, 6, 120, $b, 0, false);`
- `header('content-type:image/png');`
- `imagepng($i);`

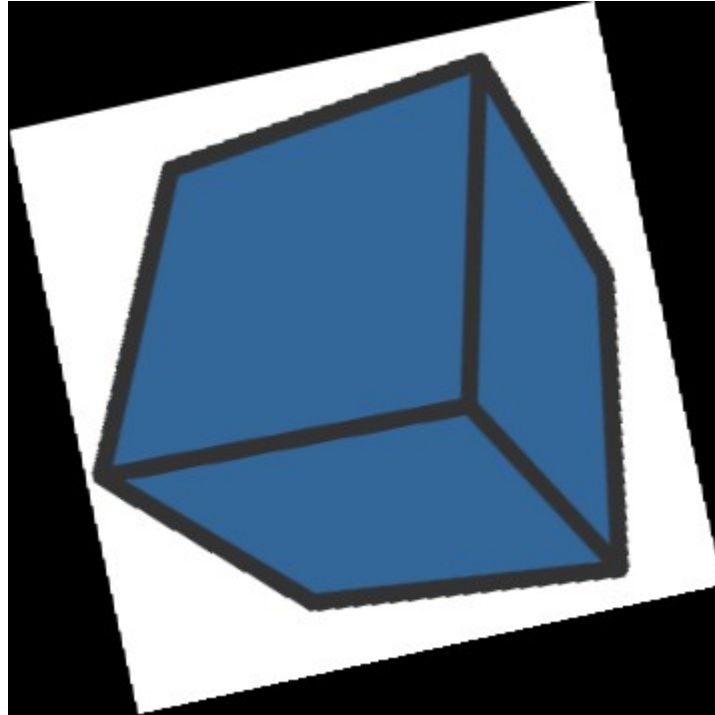
# Polígonos regulares



# Rotação

```
1. <?php
2. header('Content-type: image/gif');
3. $source =
   imagecreatefrompng('cube.png');
4. $rotate = imagerotate($source,
   12.5, 0);
5. imagepng($rotate);
6. ?>
```

# Rotação



# Filtros

- `bool imagefilter ( resource $image , int $filtertype [, int $arg1 [, int $arg2 [, int $arg3 [, int $arg4 ]]]) )`

# Filtros



# Filtros

- `imagefilter($img, IMG_FILTER_NEGATE);`



# Filtros

- `imagefilter($img, IMG_FILTER_GRAYSCALE);`



# Filtros

- `imagefilter($img, IMG_FILTER_BRIGHTNESS, 75);`



# Filtros

- `imagefilter($img, IMG_FILTER_CONTRAST, 25);`



# Filtros

- `imagefilter($img, IMG_FILTER_COLORIZE, 0xCC, 0x99, 0xCC, 0x00);`



# Filtros

- `imagefilter($img, IMG_FILTER_EDGEDETECT);`



# Filtros

- `imagefilter($img, IMG_FILTER_EMOSS);`



# Filtros

- `imagefilter($img, IMG_FILTER_GAUSSIAN_BLUR);`



# Filtros

- `imagefilter($img, IMG_FILTER_SELECTIVE_BLUR);`



# Filtros

- `imagefilter($img, IMG_FILTER_MEAN_REMOVAL);`



# Filtros

- `imagefilter($img, IMG_FILTER_SMOOTH);`



# Dúvidas



# Bibliografia

- Site oficial do PHP (<http://php.net>)
- PHP para quem conhece PHP - Juliano Niederauer
- Image fun with PHP - part 2 - <http://www.phpied.com/image-fun-with-php-part-2/>
- Image fun - <http://www.phpied.com/image-fun/>



# Agradecimentos

- Deus
- Meus pais
- Minha namorada
- Espectadores
- Organização do evento

# Just fun



## What is PHP?

PHP is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML. If you are new to PHP and want to get some idea of how it works, try the [introductory tutorial](#). After that, check out the online [manual](#), and the

exam  
some  
resol  
links

## PHP 5.3.1 Released!

*[19-Nov-2009]* The PHP development team would like to announce focuses on improving the stability of the PHP 5.3.x branch with o All users of PHP are encouraged to upgrade to this release.

### Security Enhancements and Fixes in PHP 5.3.1:

- Added "max\_file\_uploads" INI directive, which can be set to default, to prevent possible DOS via temporary file exhaust
- Added missing sanity checks around exif processing.
- Fixed a safe mode bypass in tempnam().



# Just fun



## What is PHP?

PHP is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML. If you are new to PHP and want to get some idea of how it works, try the [introductory tutorial](#). After that, check out the online [manual](#), and the example archive sites and some of the other resources available in the [links section](#).

## PHP 5.3.1 Released!

*[19-Nov-2009]* The PHP development team would like to announce focuses on improving the stability of the PHP 5.3.x branch with o All users of PHP are encouraged to upgrade to this release.

### **Security Enhancements and Fixes in PHP 5.3.1:**

- Added "max\_file\_uploads" INI directive, which can be set to default, to prevent possible DOS via temporary file exhaust
- Added missing sanity checks around exif processing.
- Fixed a safe\_mode bypass in tempnam().
- Fixed a open\_basedir bypass in posix\_mkfifo().
- Fixed failing safe\_mode\_include\_dir.



# Trabalhando com imagens no PHP

Palestra para o PHP Conference Brasil 2009

Fabio Fogliarini Brolesi  
fabio@freesandbox.net