



В данном примере, рисуя **блок с закругленными уголками** мы задаем еще и фоновое изображение. Фон состоит из двух изображений. Верхнее выравнивается по левому верхнему краю, нижнее по левому нижнему.

Формирование закруглений так же как и в [предыдущем примере](#) происходит за счет выстроенных вертикально div-ов высотой в 1 пиксель. Они имеют разный отступ от краев прямоугольника, в который обернуты. Однако пришлось отказаться от сглаживания, т.к. рассчитать цвет в конкретной точке на стороне сервера уже не представляется возможным, либо становится нецелесообразным ввиду своей трудоемкости.

На выходе формируется таблица стилей и сам div блока. Таблицу стилей можно использовать повторно, если блоки на странице однотипные.

Подобный зафekt легко реализовать при использовании html5 и css3, но ввиду распространенности старых браузеров, данный способ формирования блоков с закругленными углами остается весьма актуальным.

Необходимо учитывать тот факт, что из-за переменной высоты блока может стать заметна граница между контентом и "подвалом" блока. Поэтому следует тщательно подбирать фоновые изображения, а при отсутствии нижнего изображения, фоновый цвет, что избежать подобного эффекта. На рисунке "подвал" блока имеет синюю гамму, поэтому граница отчетлива видна.

<?php

//преобразует шестнадцатеричное число в массив rgb

```
function parseColor( $color ){
    return array(
        "r" => ($color & 0xFF0000) >> 16,
        "g" => ($color & 0xFF00) >> 8,
        "b" => $color & 0xFF
    );
}
```

//преобразует числа rgb в шестнадцатеричное число

```
function RGB( $r, $g, $b )
{
    return $r << 16 | $g << 8 | $b;
}
```

//hex цвет в строку вида 'rgb(xxx,xxx,xxx)' в соответствии с интенсивностью \${0,1}

```
function ci( $bgcolor, $f = 1 ){
    $rgb_color = parseColor( $bgcolor );
    foreach( $rgb_color as &$c ) $c = round( $c + ( 255 - $c ) * ( 1 - $f ) );
    return ' rgb('.$rgb_color['r'].','.$rgb_color['g'].','.$rgb_color['b'].') ';
}
```

//рисует блок

```
function showblock(
    $html = "", //inner html
    $bgcolor = 0xDDD0D0, //background color
    $width = 'auto',
    $height = 'auto',
    $radius = 16,
    $scale = 1,
    $padding = '10px',
    $fon_image1 = 'fon1.png',
    $fon_image2 = 'fon2.png'
)
{
    $id = 's'.rand(0,9999).time();
```

```
    echo '<style type="text/css">';
```

```
    for( $i = 1; $i <= $radius; $i++ ) echo ' '.$id.'.r'.$i.( $i != $radius ? ', ' : " );
```

```
    echo '{display:block;height:1px;}';
```

```
    $previous_margin = $radius * $scale / 2;
```

```
    for( $i = 1; $i <= $radius; $i++ ){
        $a = $radius - $i + 1;
```

```
$d = sqrt( $radius * $radius - $a * $a ); //теорема пифагора
$now_margin = ( $radius - $d ) * $scale;
$d_margin = abs( $previous_margin - $now_margin );
$border = floor( ( $d_margin / $scale ) / 2 ); //толщина рамки (используется для
сглаживания краев)
if( $border == 0 ) $border = 1;
$f = $d - floor( $d ); //интенсивность цвета рамки (используется для сглаживания краев)
if( $i == 1 ) $f = 0.5;

echo ' '.$id.'.r'.$i.' { height:'.$scale.'px;background-color:'.ci($bgcolor).';margin: 0 ' . (($radius -
floor($d))*$scale).'px;
background-image:url("'.$fon_image1.'");
background-position:-'.(($radius - floor($d))*$scale).'px -'.$i.'px;
//font-size:0;
//margin-top:-1px;
}';

$previous_margin = $now_margin;
}
//////////
for( $i = 1; $i <= $radius; $i++ ) echo ' '.$id.'.r'.$i.'.b'.( $i != $radius ? ', ' : " );

echo '{display:block;height:1px;}';

$previous_margin = $radius * $scale / 2;
for( $i = 1; $i <= $radius; $i++ ){
$a = $radius - $i + 1;
$d = sqrt( $radius * $radius - $a * $a ); //теорема пифагора
$now_margin = ( $radius - $d ) * $scale;
$d_margin = abs( $previous_margin - $now_margin );
$border = floor( ( $d_margin / $scale ) / 2 ); //толщина рамки (используется для
сглаживания краев)
if( $border == 0 ) $border = 1;
$f = $d - floor( $d ); //интенсивность цвета рамки (используется для сглаживания краев)
if( $i == 1 ) $f = 0.5;

echo ' '.$id.'.r'.$i.'.b { height:'.$scale.'px;background-color:'.ci($bgcolor).';margin: 0 ' . (($radius -
floor($d))*$scale).'px;
background-image:url("'.$fon_image2.'");
background-position:-'.(($radius - floor($d))*$scale).'px '.$i.'px;
//font-size:0;
//margin-top:-1px;
}';

$previous_margin = $now_margin;
}
```

```
//////////
```

```
echo '#'.$id.'rounded-box '.$id.'inner-box, #'.$id.'rounded-box div
{background-color:'.ci($bgcolor).'};
'.$id.'inner-box {padding:0 '.$padding.' 0 '.$padding.';
background-image:url("'.$fon_image1.'");
background-position:0 -'.$radius.'px;
}
</style>;
```

```
echo '<div id="'.$id.'rounded-box">;
```

```
for( $i = 2; $i <= $radius; $i++ ) echo '<div class="'.$id.'r'.$i.'"></div>;
```

```
echo '<div class="'.$id.'inner-box";background-color:'.ci($bgcolor).';">'.$html.'</div>;
```

```
for( $i = $radius; $i >= 2; $i-- ) echo '<div class="'.$id.'r'.$i.'b"></div>;
```

```
echo '</div>;
```

```
return;
```

```
}
```

```
?>
```

```
<div style="width:300px;">
```

```
<?php
```

```
for($i=0;$i<101;$i+=10){
```

```
showblock('Блок с радиусом закругления<br/><b
```

```
style="font-size:2em;">'.$i.'px</b><br/>Формирование закруглений происходит за счет
```

```
выстроенных вертикально div-ов высотой в 1 пиксель. Они имеют разный отступ от
```

```
краев прямоугольника, в который обернуты. Эффект сглаживания достигается путем
```

```
использования право- и левосторонней рамки с цветом меньшей интенсивности, чем
```

```
основной. Интенсивность цвета рамки и ее толщина рассчитываются программно.',
```

```
0xDDD0D0, 'auto', 'auto', $i);
```

```
echo '<br/><br/>;
```

```
}
```

```
?>
```

```
</div>
```