PHP - Advanced Tutorial

Gulev

December 6, 2002. Veracruz, Mexico

Rasmus Lerdorf <rasmus@php.net>

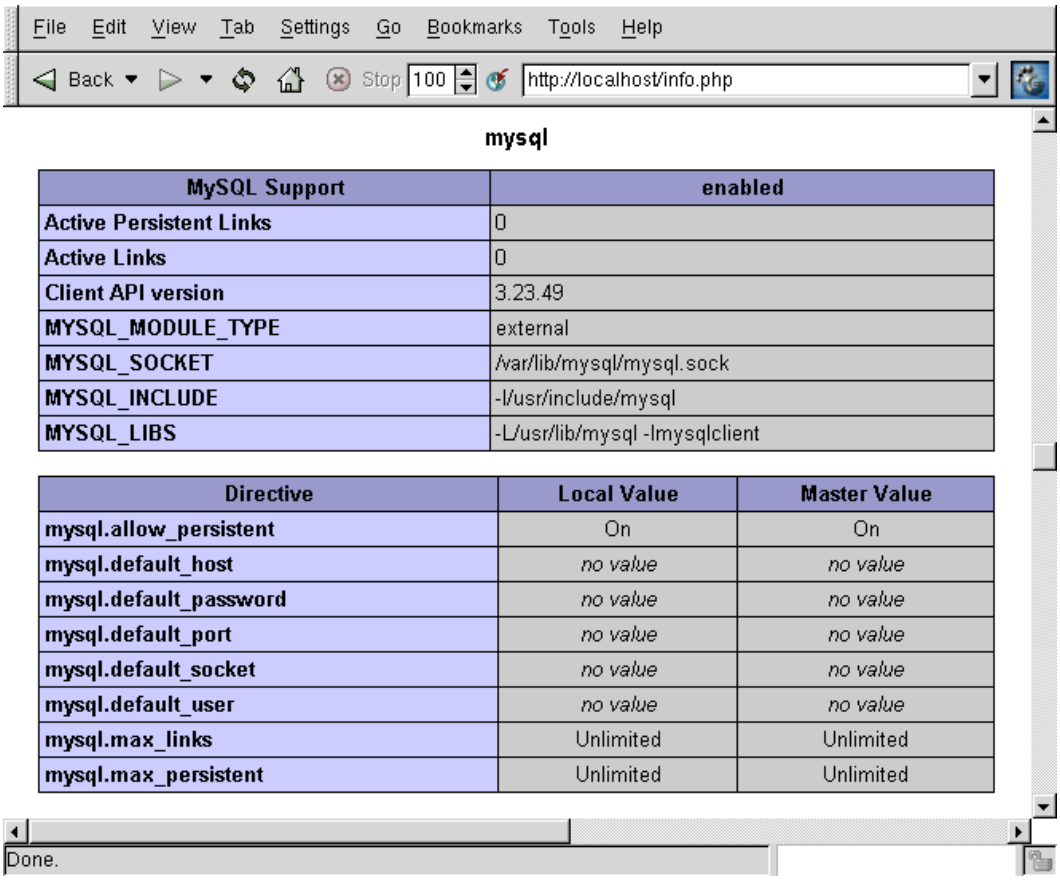
http://lerdorf.com/veracruz.pdf

Agenda

- PHP-MySQL
- Cookie Handling
- Dynamic Images
- PDF
- Flash
- Sessions
- Security
- Tips & Tricks
- Optimization
- Latest Developments
- Future

Check your PHP Setup for MySQL support

```
<? phpinfo() ?>
```



If not enabled

Very rare since a MySQL client library is distributed with PHP and built into PHP by default. However, it is possible to build PHP without MySQL support. Some possible fixes:

```
apt-get install php-mysql

rpm -Uvh php-mysql-4.2.2-1.i386.rpm



./configure --with-mysql=shared,/usr
cp modules/mysql.so /usr/local/lib/php

extension_dir=/usr/local/lib/php
extension=mysql.so
```

Sanity Check

Make sure MySQL is running

```
prompt:~> mysqlshow
+-----------+
| Databases |
+-----------+
| mysql     |
| test      |
+-----------+
```

Or with the latest PHP

```
<? echo mysql_stat() ?>
```

Output:

```
Uptime: 6717  Threads: 1  Questions: 1  Slow queries: 0  Opens: 6  Flush tables: 1
Open tables: 0 Queries per second avg: 0.000
```

Connecting to MySQL

The simple connection

```
<?
$conn = mysql_connect('localhost');
echo $conn;
?>
```

Output:

```
Resource id #28
```

Other variations

```
<?
mysql_connect('db.domain.com:33306','rasmus','foobar');
mysql_connect('localhost:/tmp/mysql.sock');
mysql_connect('localhost','rasmus','foobar',
              true,MYSQL_CLIENT_SSL|MYSQL_CLIENT_COMPRESS);
?>
```

Persistent Connections

The simple connection

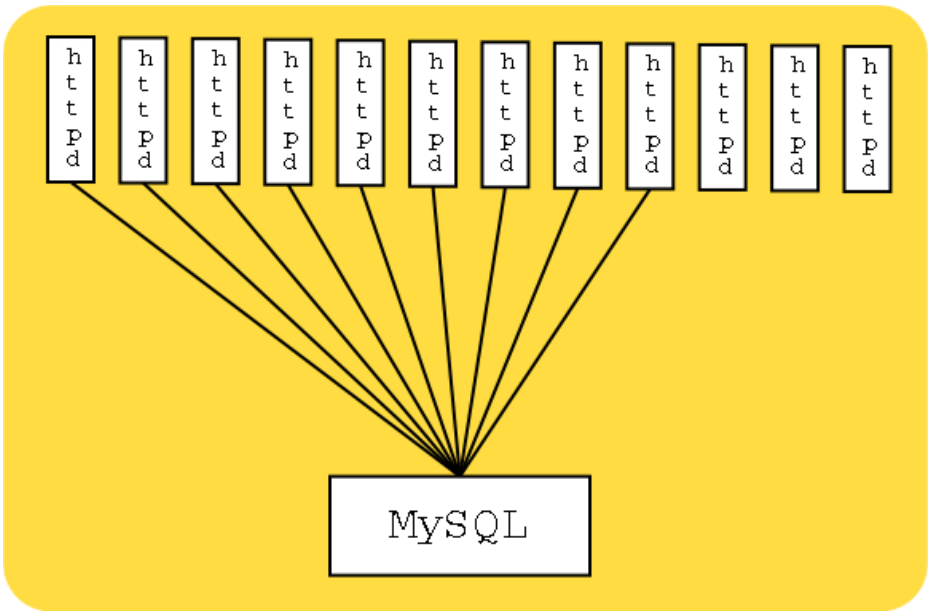
```
<?
$conn = mysql_pconnect('localhost');
echo $conn;
?>
```

Output:

```
Resource id #31
```

Caveats

- Watch out for multi-credential connections
- Make sure you match up max_connections and MaxClients



Creating a Database

Create a DB

```
<?
  mysql_connect('localhost');
  if(mysql_query("CREATE DATABASE foo")) {
    echo "Database foo created";
  } else {
    echo mysql_error();
  }
?>
```

Output:

```
Database foo created
```

Create a Table

```
<?
  mysql_select_db('foo');
  $result = mysql_query("CREATE TABLE users (
    id varchar(16) binary NOT NULL default '',
    Password varchar(16) NOT NULL default '',
    Name varchar(64) default NULL,
    email varchar(64) default NULL,
    ts timestamp(14) NOT NULL,
    PRIMARY KEY (id)
  )");
  if($result) {
    echo "Table created";
  } else {
    echo mysql_error();
  }
?>
```

Output:

```
Table created
```

Inserting Data

INSERT Query

```
<?php
function add_user($id, $pass, $name, $email) {
  $result=mysql_query("insert into users values
                      ('$id',ENCRYPT('$pass'),'$name','$email',NULL)");
  if($result) {
    echo "Row inserted<br />";
  } else {
    echo mysql_error()."<br />";
  }
}

mysql_connect('localhost');
mysql_select_db('foo');

add_user('rasmus','foobar','Rasmus Lerdorf','rasmus@php.net');
add_user('carl','carlspass','Carl AlexandeR Lerdorf','carl@lerdorf.com');
?>
```

Output:

```
Row insertedRow inserted
```

Selecting Data

SELECT Query

```
<?
  mysql_connect('localhost');
  mysql_select_db('foo');
  $result = mysql_query("select * from users");
  if(!$result) echo mysql_error();
  else {
    while($row = mysql_fetch_row($result)) {
        echo "$row[0] - $row[1] - $row[2] - $row[3] - $row[4]<br />\n";
    }
  }
?>
```

Output:

```
rasmus - aVOtbUF2LODnw - Rasmus Lerdorf - rasmus@php.net - 20021206142646
carl - aVOnaDJh48k7o - Carl AlexandeR Lerdorf - carl@lerdorf.com - 20021206142646
```

mysql_fetch_array()

```
<?
  $result = mysql_query("select * from users order by id");
  if(!$result) echo mysql_error();
  else {
    while($row = mysql_fetch_array($result,MYSQL_ASSOC)) {
        echo "$row[id] - $row[Password] - $row[Name] -
              $row[email] - $row[ts]<br />\n";
    }
  }
?>
```

Output:

```
carl - aVOnaDJh48k7o - Carl AlexandeR Lerdorf -
              carl@lerdorf.com - 20021206142646
rasmus - aVOtbUF2LODnw - Rasmus Lerdorf -
              rasmus@php.net - 20021206142646
```

Dealing with timestamps

Using DATE_FORMAT

```
<?
mysql_connect('localhost');
mysql_select_db('foo');
$result = mysql_query(
  "select id, email,
   date_format(ts,'%W %M %D, %Y %r') as d
   from users order by ts");
if($result) {
  while($row = mysql_fetch_assoc($result)) {
    echo "$row[id] - $row[email] - $row[d]<br />\n";
  }
} else {
  echo mysql_error();
}
?>
```

Output:

```
rasmus - rasmus@php.net - Friday December 6th, 2002 02:26:46 PM
carl - carl@lerdorf.com - Friday December 6th, 2002 02:26:46 PM
```

Changing Existing Rows

Using UPDATE

```
<?
  mysql_connect('localhost');
  mysql_select_db('foo');
  $result = mysql_query(
    "update users set email = 'babycarl@lerdorf.com'
     where id = 'carl'");
  if($result) {
    echo mysql_affected_rows();
  } else {
    echo mysql_error();
  }
?>
```

Output:

```
1
```

REPLACE INTO

You can also use REPLACE INTO to update a row if it exists and insert it if it doesn't.

Magic Quotes

Escaping troublesome characters

When you are inserting data into a MySQL database, certain characters have a special meaning and must therefore be escaped if you wish to insert these characters literally.

By default, PHP will escape these characters for you in any data coming from the user in GET, Post or Cookie data. This magic escaping is known as Magic Quotes and can be configured in your php.ini file by setting the magic_quotes_gpc directive.

The characters affected are \ ' " and NUL (char 0). If these characters appear in user-supplied data they will be escaped with a \ (backslash).

Some people prefer to turn this feature off and handle escaping data manually using the addslashes() function. There is a converse function, stripslashes(), which removes the backslash characters in an escaped string.

Guestbook Example

A very simple guestbook example to illustrate basic file handling.

```
<html><head><title>My Guestbook</title></head>
<body>
<h1>Welcome to my Guestbook</h1>
<h2>Please write me a little note below</h2>
<form action="<?="$PHP_SELF#results"?>" method="POST">
<textarea cols=40 rows=5 name=note wrap=virtual></textarea>
<input type=submit value=" Send it ">
</form>
<?if(isset($note)) {
    $fp = fopen("/tmp/notes.txt","a");
    fputs($fp,nl2br($note).'<br>');
    fclose($fp);
  }
?><h2>The entries so far:</h2>
<? @ReadFile("/tmp/notes.txt") ?>
</body></html>
```

Output:

```
My Guestbook

Welcome to my Guestbook
Please write me a little note below



The entries so far:
```

SQL'izing the Guestbook Example

We are going to convert this into an SQL-driven guestbook by first creating a database, then a schema for the table where we will store the data and then we will modify the code.

Create a database

```
mysqladmin create mydb
```

Create a Schema

```
CREATE TABLE comments (
  id int(8) DEFAULT '0' NOT NULL auto_increment,
  comment text,
  ts datetime,
  PRIMARY KEY (id)
);
```

SQL'izing the Guestbook Example

Here we add the necessary code to store our guestbook comments in an SQL database

```
<html><head><title>My Guestbook</title></head>
<body>
<h1>Welcome to my Guestbook</h1>
<h2>Please write me a little note below</h2>
<form action="<? echo "$PHP_SELF#results"?>" method="POST">
<textarea cols=40 rows=5 name="note" wrap=virtual></textarea>
<input type="submit" value=" Send it ">
</form>
<?
  mysql_connect('localhost');
  mysql_select_db('mydb');
  if(isset($note)) {
     $ts = date("Y-m-d H:i:s");
     mysql_query("insert into comments values
                  (0,'$note','$ts')");
  }
?>
<h2>The entries so far:</h2>
<? $result = mysql_query("select * from comments");
     while($row=mysql_fetch_row($result)) {
       echo $row[0] ." " . $row[1] . " " . $row[2] . "<br>\n";
     }  ?>
</body></html>
```

Output:

```
My Guestbook

Welcome to my Guestbook
Please write me a little note below



The entries so far:
```

DB Abstraction

A database abstraction layer is bundled with PHP 4. In the example below, the only thing you would need to change to use a different database is the odbc word on the third line.

```
<?php
  require_once 'DB.php';
  $db = DB::connect('odbc://user:pw@host/mydb');
  $stmt = $db->prepare('SELECT * FROM comments');
  $result = $db->execute($stmt);
  while($row = $db->fetchrow($result)) {
     while($row as $field => $value ) {
        echo "$field: $value<br>\n";
     }
  }
  $db->disconnect();
?>
```

HTTP Headers

You can add headers to the HTTP response in PHP using the Header() function. Since the response headers are sent before any of the actual response data, you have to send these headers before outputting any data. So, put any such header calls at the top of your script.

Redirection

```
<?header('Location: http://www.php.net')?>
```

Setting a Last-Modified Header

```
<?header('Last-Modified: '.
      gmdate('D, d M Y H:i:s',getlastmod()).' GMT')?>
```

Avoid all Caching

```
<?php
Header('Cache-Control: no-cache, must-revalidate');
Header('Pragma: no-cache');
Header('Expires: Mon,26 Jul 1980 05:00:00 GMT');
?>
```

Cookies

Setting a Session Cookie

```
<? SetCookie('Cookie_Name','value'); ?>
```

Setting a Persistent Cookie

```
<? SetCookie('Cookie_Name','value',
             mktime(12,0,0,22,11,2002) ); ?>
```

Reading a Cookie

```
<? echo $Cookie_Name; ?>

<? echo $HTTP_COOKIE_VARS['Cookie_Name']; ?>
```

Deleting the Cookies

```
<? SetCookie('Cookie_Name',''); ?>

<? SetCookie('Cookie_Name','',
             mktime(12,0,0,22,11,1970) ); ?>
```

Other Optional Paramters

Path, Domain, and Secure parameters can also be set to restrict a cookie to a certain path, domain or in the case of the Secure parameter, limit the cookie to only be set if the request came in over an SSL connection.

Cookie Expiry

Problem

Short expiry cookies depend on users having their system clocks set correctly.

Solution

Don't depend on the users having their clocks set right. Embed the timeout based on your server's clock in the cookie.

```
<?php
  $value = time()+3600 . ':' . $variable;
  SetCookie('Cookie_Name',$value);
?>
```

Then when you receive the cookie, decode it and determine if it is still valid.

```
<?php
list($ts,$variable) = explode(':',$Cookie_Name,2);
if($ts < time()) {
   ...
} else {
    SetCookie('Cookie_Name','');
}
?>
```

Creating a PNG with a TrueType font

```
<?
Header("Content-type: image/png");
$im = ImageCreate(630,80);
$blue = ImageColorAllocate($im,0x5B,0x69,0xA6);
$white = ImageColorAllocate($im,255,255,255);
$black = ImageColorAllocate($im,0,0,0);
ImageTTFText($im, 45, 0, 10, 57, $black, "CANDY", $text);
ImageTTFText($im, 45, 0, 6, 54, $white, "CANDY", $text);
ImagePNG($im);
?>

<IMG src="txt.php?text=<?echo urlencode($text)?>">

!
```

Color Handling

For images with an 8-bit indexed palette it can be tricky to manage colors.

```
<?
$im  = ImageCreate(300,256);
for($r=0; $r<256; $r++) {
    $col = ImageColorAllocate($im,$r,0,0);
    ImageLine($im, 0,$r, 100, $r, $col);
}
for($g=0; $g<256; $g++) {
    $col = ImageColorAllocate($im,0,$g,0);
    ImageLine($im, 100,255-$g, 200, 255-$g, $col);
}
for($b=0; $b<256; $b++) {
    $col = ImageColorAllocate($im,0,0,$b);
    ImageLine($im, 200,$b, 300, $b, $col);
}
Header('Content-Type: image/png');
ImagePNG($im);
?>
```

Output:

For paletted images the following functions can be useful:

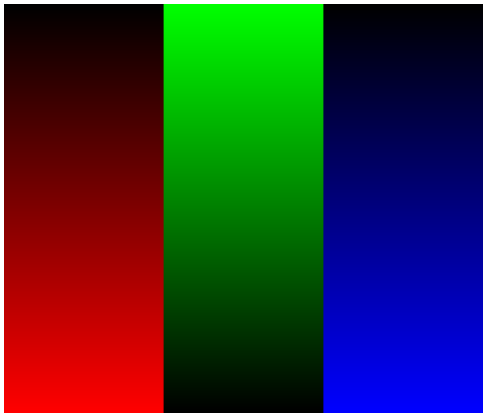
- ImageColorClosest
- ImageColorExact
- ImageColorDeallocate

Colour Handling

For Truecolor images we have no such issues.

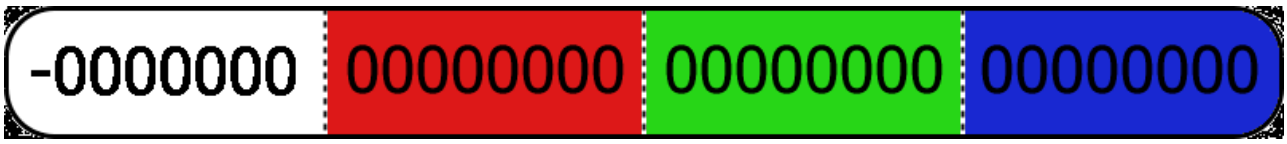
```
<?
$im  = ImageCreateTruecolor(300,256);
for($r=0; $r<256; $r++) {
    $col = ImageColorAllocate($im,$r,0,0);
    ImageLine($im, 0,$r, 100, $r, $col);
}
for($g=0; $g<256; $g++) {
    $col = ImageColorAllocate($im,0,$g,0);
    ImageLine($im, 100,255-$g, 200, 255-$g, $col);
}
for($b=0; $b<256; $b++) {
    $col = ImageColorAllocate($im,0,0,$b);
    ImageLine($im, 200,$b, 300, $b, $col);
}
Header('Content-Type: image/png');
ImagePNG($im);
?>
```

Output:



Truecolor color handling

For Truecolor images the colors are actually simple 31-bit longs. Or, think of them as being composed of 4 bytes arranged like this:

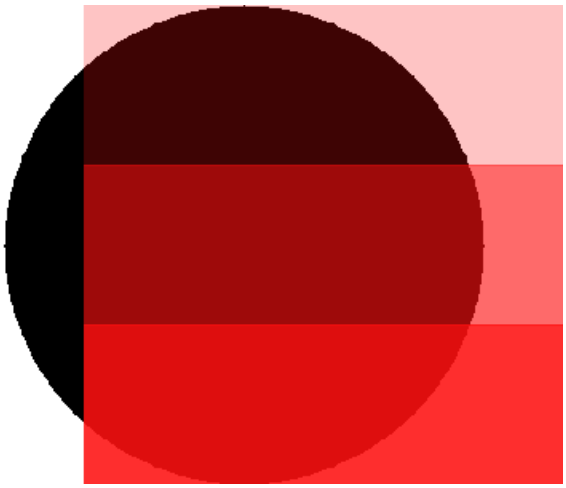


The highest or leftmost bit in the alpha channel is not used which means the alpha channel can only have values from 0 to 127. You can use the ImageColorAllocate() as with paletted images, but you can also specify the color directly.

For example:

```
<?
$im  = ImageCreateTruecolor(400,300);
ImageFilledRectangle($im,0,0,399,299,0x00ffffff);
ImageFilledEllipse($im,200,150,300,300,0x00000000);
ImageAlphaBlending($im,true);
ImageFilledRectangle($im,100,0,400,100,0x60ff1111);
ImageFilledRectangle($im,100,100,400,200,0x30ff1111);
ImageFilledRectangle($im,100,200,400,300,0x10ff1111);
Header('Content-Type: image/png');
ImagePNG($im);
?>
```

Output:



This example could also be written like this:

```
<?php
$im  = ImageCreateTruecolor(400,300);
$white = ImageColorAllocate($im,255,255,255);
ImageFilledRectangle($im,0,0,399,299,$white);
$black = ImageColorAllocate($im,0,0,0);
ImageFilledEllipse($im,200,150,300,300,$black);
ImageAlphaBlending($im,true);
$col = ImageColorResolveAlpha($im,0xff,0x11,0x11,0x60);
ImageFilledRectangle($im,100,0,400,100,$col);
$col = ImageColorResolveAlpha($im,0xff,0x11,0x11,0x30);
ImageFilledRectangle($im,100,100,400,200,$col);
$col = ImageColorResolveAlpha($im,0xff,0x11,0x11,0x10);
```

```
ImageFilledRectangle($im,100,200,400,300,$col);
Header('Content-Type: image/png');
ImagePNG($im);
?>
```

Truecolor Color Handling

Given the nature of the way truecolor colors are constructed, we can rewrite our color testing strip using PHP's bitshift operator:

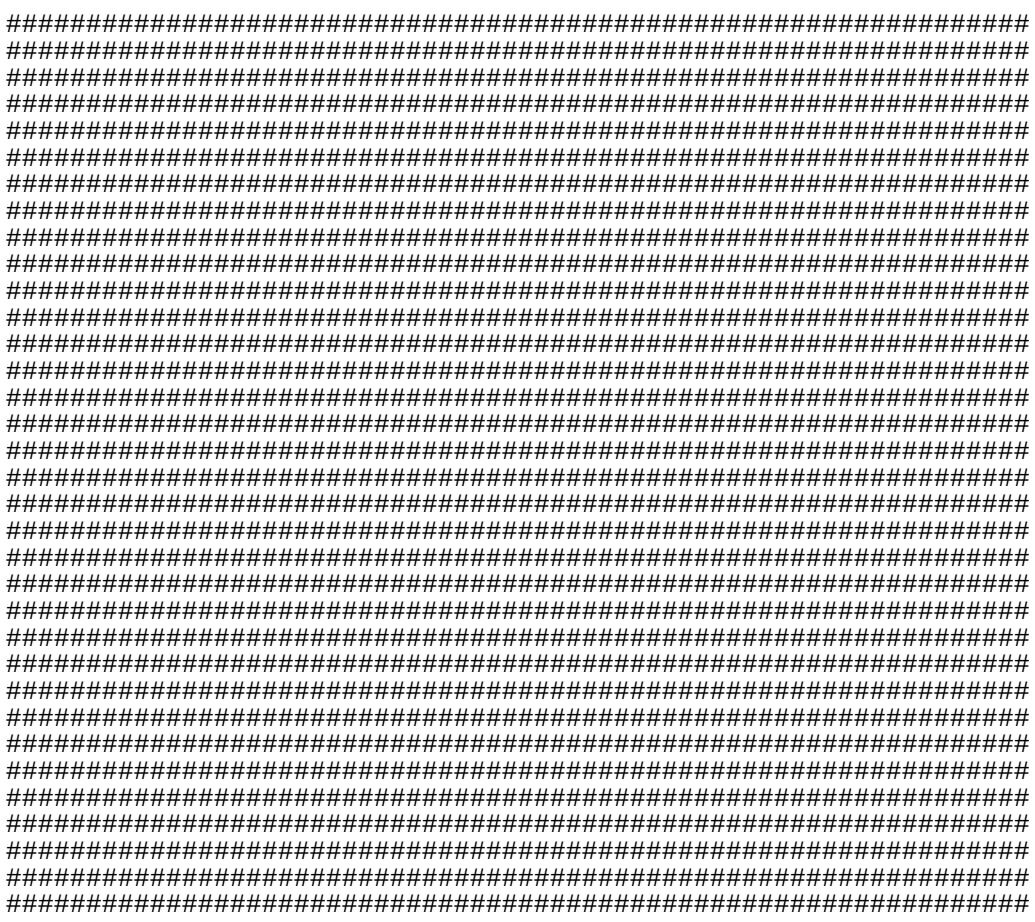
```
<?
$im = ImageCreateTrueColor(256,60);
for($x=0; $x<256; $x++) {
    ImageLine($im, $x, 0, $x, 19, $x);
    ImageLine($im, 255-$x, 20, 255-$x, 39, $x<<8);
    ImageLine($im, $x, 40, $x, 59, $x<<16);
}
Header('Content-Type: image/png');
ImagePNG($im);
?>
```

Output:

Using ImageColorAt()

```
<? $image = "presentations/slides/intro/php-tiny.jpg"; ?>
<img src="<?=$image?>"><br />
<?
    $im = imagecreatefromjpeg($image);
    $dx = imagesx($im);
    $dy = imagesy($im);
    for($y = 0; $y < $dy; $y++) {
        for($x=0; $x < $dx; $x++) {
            $col = imagecolorat($im, $x, $y);
            $rgb = imagecolorsforindex($im,$col);
            printf("<font color=#%02x%02x%02x>#</font>",
              $rgb['red'],$rgb['green'],$rgb['blue']);
        }
        echo "<br />\n";
    }
    imagedestroy($im);
?>
```

Output:



CreateFrom and Bounding Box Math

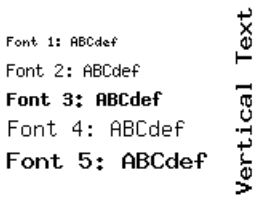
```
<?
Header("Content-type: image/png");
$font = 'phpi';
if(!$si) $si = 66;
$im = ImageCreateFromPNG('php-blank.png');
$tsize = imagettfbbox($si,0,$font,$text);
$dx = abs($tsize[2]-$tsize[0]);
$dy = abs($tsize[5]-$tsize[3]);
$x = ( imagesx($im) - $dx ) / 2;
$y = ( imagesy($im) - $dy ) / 2 + 3*$dy/4;
$blue = ImageColorAllocate($im,0x5B,0x69,0xA6);
$white = ImageColorAllocate($im,255,255,255);
$black = ImageColorAllocate($im,0,0,0);
ImageAlphaBlending($im,true);
ImageTTFText($im, $si, 0, $x, $y, $white, $font, $text);
ImageTTFText($im, $si, 0, $x+2, $y, $white, $font, $text);
ImageTTFText($im, $si, 0, $x, $y+2, $white, $font, $text);
ImageTTFText($im, $si, 0, $x+2, $y+2, $white, $font, $text);
ImageTTFText($im, $si, 0, $x+1, $y+1, $black, $font, $text);
ImagePNG($im);
?>

<IMG src="txt2.php?text=<?echo urlencode($text)?>&si=<?echo $si?>">

Text:  Size:  !
```

Built-in Fonts

GD comes with 5 built-in fonts. They aren't all that useful.

```
<?
$im  = ImageCreate(175,125);
$white = ImageColorAllocate($im,255,255,255);
$black = ImageColorAllocate($im,0,0,0);
ImageString($im,1,10,20,"Font 1: ABCdef",$black);
ImageString($im,2,10,35,"Font 2: ABCdef",$black);
ImageString($im,3,10,53,"Font 3: ABCdef",$black);
ImageString($im,4,10,70,"Font 4: ABCdef",$black);
ImageString($im,5,10,90,"Font 5: ABCdef",$black);
ImageStringUp($im,5,150,118,"Vertical Text",$black);
Header('Content-Type: image/png');
ImagePNG($im);
?>
```

Output:



TrueType Fonts

You can use any TrueType Font that includes a Unicode mapping table. Fonts such as Wingdings will not work.

```
<?
$im  = ImageCreate(600,7150);
$white = ImageColorAllocate($im,255,255,255);
$black = ImageColorAllocate($im,0,0,0);
$dir = opendir('/usr/share/fonts/truetype');
$y=30;
while($file = readdir($dir)) {
    if(substr($file,strrpos($file,'.'))=='.ttf') {
        ImageString($im,5,5,$y-20,substr($file,0,-4),$black);
        ImageTTFText($im,30,0,100,$y,$black, substr($file,0,-4),"ABCdéf123");
        $y+=40;
    }
}
Header('Content-Type: image/png');
ImagePNG($im);
?>
```

Output:

abalc 

a_d_mono 

Adresack 

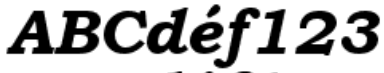
aggstock 

ariblk 

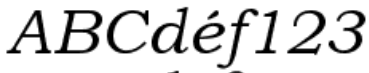
arnari 

arnar 

bkant 

Bookosbi 

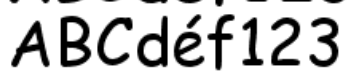
bookosb 

Bookosi 

Bookos 

calist 

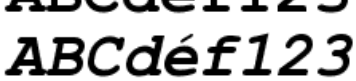
comicbd 

comic 

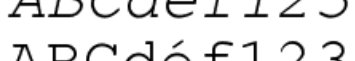
coprgtb 

coprgtl 

courbd 

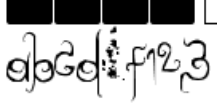
courbi 

couri 

cour 

dc_sans 

dc_serif 

dilate 

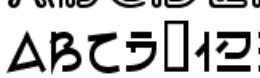
dirtydoz 

Dotmatrx 

dronecat 

earth 

Eklekti0 

electroh 

Eroded2020 

Reading EXIF Headers from a JPEG

```
<?php
$data = exif_read_data('presentations/slides/intro/img_resize.jpg');
foreach($data as $key=>$val) {
    if(is_array($val)) {
        foreach($val as $k=>$v) {
            echo $key."[$k]: $v<br />\n";
        }
    } else
        echo "$key: ".@substr($val,0,40)."<br />\n";
}
?>
```

Output:

```
FileName: img_resize.jpg
FileDateTime: 1027351588
FileSize: 669158
FileType: 2
MimeType: image/jpeg
SectionsFound: ANY_TAG, IFD0, THUMBNAIL, EXIF
COMPUTED[html]: width="1536" height="1024"
COMPUTED[Height]: 1024
COMPUTED[Width]: 1536
COMPUTED[IsColor]: 1
COMPUTED[ByteOrderMotorola]: 0
COMPUTED[ApertureFNumber]: f/4.0
COMPUTED[FocusDistance]: 1.07m
COMPUTED[Thumbnail.FileType]: 8
COMPUTED[Thumbnail.MimeType]: image/tiff
COMPUTED[Thumbnail.Height]: 64
COMPUTED[Thumbnail.Width]: 96
Make: Eastman Kodak Company
Model: KODAK DC265 ZOOM DIGITAL CAMERA (V01.00)
Orientation: 1
XResolution: 150/1
YResolution: 150/1
ResolutionUnit: 2
YCbCrPositioning: 1
Exif_IFD_Pointer: 190
THUMBNAIL[ImageWidth]: 96
THUMBNAIL[ImageLength]: 64
THUMBNAIL[BitsPerSample]: Array
THUMBNAIL[Compression]: 1
THUMBNAIL[PhotometricInterpretation]: 2
THUMBNAIL[StripOffsets]: 1748
THUMBNAIL[Orientation]: 1
THUMBNAIL[SamplesPerPixel]: 3
THUMBNAIL[RowsPerStrip]: 64
THUMBNAIL[StripByteCounts]: 18432
THUMBNAIL[XResolution]: 72/1
THUMBNAIL[YResolution]: 72/1
THUMBNAIL[PlanarConfiguration]: 1
THUMBNAIL[ResolutionUnit]: 2
ExposureTime: 1/250
FNumber: 400/100
ExifVersion: 0200
DateTimeOriginal: 1999:01:31 04:17:59
ComponentsConfiguration:
```

Fetching an embedded thumbnail

```
<?
```

```
Header('Content-type: image/tiff');
echo exif_thumbnail('p0004557.jpg');
?>
```

A PDF Invoice

```
<?php
$pdf = pdf_new();
pdf_open_file($pdf);
pdf_set_info($pdf, "Author","Rasmus Lerdorf");
pdf_set_info($pdf, "Title","Sample Invoice");
pdf_set_info($pdf, "Creator", "See Author");
pdf_set_info($pdf, "Subject", "Sample Invoice");

$sizes = array('a4'=>'595x842', 'letter'=>'612x792', 'legal'=>'612x1008');

if(!isset($type)) $type='letter';
list($x,$y) = explode('x',$sizes[$type]);

$items = array(array('Our special low-cost widget that does everything','299.99'),
               array('Our special high-cost widget that does more','1899'),
               array('A blue widget','29.95'),
               array('And a red widget','49.95'),
               array('A yellow widget that makes noise','49.9'),
               array('And one that doesn\'t','999.95'),
              );

pdf_begin_page($pdf, $x, $y);

$im = pdf_open_jpeg($pdf, "php-big.jpg");
pdf_place_image($pdf, $im, 5, $y-72, 0.5);
pdf_close_image ($pdf,$im);

pdf_set_value($pdf, 'textrendering', 0); // fill

pdf_set_font($pdf, "Helvetica" , 12, winansi);
pdf_show_xy($pdf, 'Generic Evil Company Inc.',145,$y-20);
pdf_continue_text($pdf, '123 Main Street');
pdf_continue_text($pdf, 'Dark City, CA 98765');

pdf_set_font($pdf, "Helvetica" , 10, winansi);
pdf_show_xy($pdf, 'Helpless Customer Ltd.',20,$y-100);
pdf_continue_text($pdf, '2 Small Street');
pdf_continue_text($pdf, 'Little Town, ID 56789');

pdf_set_font($pdf, "Helvetica" , 10, winansi);
pdf_show_xy($pdf, 'Terms: Net 30',150,$y-100);
pdf_continue_text($pdf, 'PO #:  12345');

pdf_set_font($pdf, "Helvetica-Bold" , 30, winansi);
pdf_show_xy($pdf, "* I N V O I C E *",$x-250,$y-112);

pdf_setcolor($pdf,'fill','gray',0.9,0,0,0);
pdf_rect($pdf,20,80,$x-40,$y-212);
pdf_fill_stroke($pdf);

$offset = 184; $i=0;
while($y-$offset > 80) {
    pdf_setcolor($pdf,'fill','gray',($i%2)?0.8:1,0,0,0);
    pdf_setcolor($pdf,'stroke','gray',($i%2)?0.8:1,0,0,0);
    pdf_rect($pdf,21,$y-$offset,$x-42,24);
    pdf_fill_stroke($pdf);
    $i++; $offset+=24;
}

pdf_setcolor($pdf,'fill','gray',0,0,0,0);
pdf_setcolor($pdf,'stroke','gray',0,0,0,0);
pdf_moveto($pdf, 20,$y-160);
pdf_lineto($pdf, $x-20,$y-160);
```

```
pdf_stroke($pdf);

pdf_moveto($pdf, $x-140,$y-160);
pdf_lineto($pdf, $x-140,80);
pdf_stroke($pdf);

pdf_set_font($pdf, "Times-Bold" , 18, winansi);
pdf_show_xy($pdf, "Item",30,$y-150);
pdf_show_xy($pdf, "Price",$x-100,$y-150);


pdf_set_font($pdf, "Times-Italic" , 15, winansi);

$offset = 177;
foreach($items as $item) {
    pdf_show_xy($pdf, $item[0],30,$y-$offset);
    pdf_show_boxed($pdf, '$'.number_format($item[1],2), $x-55, $y-$offset, 0, 0,
'right');
    $offset+=24;
    $total += $item[1];
}

pdf_set_font($pdf, "Times-Bold" , 17, winansi);
$offset+=24;
pdf_show_xy($pdf, 'Total',30,$y-$offset);
pdf_show_boxed($pdf, '$'.number_format($total,2), $x-55, $y-$offset, 0, 0,
'right');

pdf_end_page($pdf);
pdf_close($pdf);

$data = pdf_get_buffer($pdf);
header('Content-type: application/pdf');
header("Content-disposition: inline; filename=invoice.pdf");
header("Content-length: " . strlen($data));
echo $data;
?>
```

Output:

See http://www.opaque.net/ming/

```
<?
    $s = new SWFShape();
    $fp = fopen('php-big.jpg','r');
    $jpg = new SWFBitmap($fp);
    $w = $jpg->getWidth(); $h = $jpg->getHeight();

    $f = $s->addFill($jpg);
    $f->moveTo(-$w/2, -$h/2);
    $s->setRightFill($f);

    $s->movePenTo(-$w/2, -$h/2);
    $s->drawLine($w, 0);
    $s->drawLine(0, $h);
    $s->drawLine(-$w, 0);
    $s->drawLine(0, -$h);

    $p = new SWFSprite();
    $i = $p->add($s);

    for($step=0; $step<360; $step+=2) {
        $p->nextFrame();
        $i->rotate(-2);
    }

    $m = new SWFMovie();
    $i = $m->add($p);
    $i->moveTo(230,120);
    $m->setRate(100);
    $m->setDimension($w*1.8, $h*1.8);

    header('Content-type: application/x-shockwave-flash');
    $m->output();
?>
```

Output:

Flash + RSS/XML

```
<?php
require 'XML/RSS.php';

$r =& new XML_RSS('slashdot.rdf');
$r->parse();

$allItems = $r->getItems();
$itemCount = count($allItems);
$width = 1000;
$m = new SWFMovie();
$m->setDimension($width, 70);
$m->setBackground(0xcf, 0xcf, 0xcf);

$f = new SWFFont("../../../fonts/Techno.fdb");

$hit = new SWFShape();
$hit->setRightFill($hit->addFill(0,0,0));
$hit->movePenTo(-($width/2), -30);
$hit->drawLine($width, 0);
$hit->drawLine(0, 60);
$hit->drawLine(-$width, 0);
$hit->drawLine(0, -60);
$x = 0;

// build the buttons
foreach($allItems as $Item) {

    $title = $Item['title'];
    $link = $Item['link'];

    // get the text
    $t = new SWFText();
    $t->setFont($f);
    $t->setHeight(50);
    $t->setColor(0,0,0);
    $t->moveTo(-$f->getWidth($title)/2, 25);
    $t->addString($title);

    // make a button
    $b[$x] = new SWFButton();
    $b[$x]->addShape($hit, SWFBUTTON_HIT);
    $b[$x]->addShape($t, SWFBUTTON_OVER | SWFBUTTON_UP | SWFBUTTON_DOWN);
    $b[$x++]->addAction(new SWFAction("getURL('$link','_new');"), SWFBUTTON_MOUSEUP);
}


// display them
for($x=0; $x<$itemCount; $x++) {

    $i = $m->add($b[$x]);
    $i->moveTo($width/2,30);

    for($j=0; $j<=30; ++$j) {
        $i->scaleTo(sqrt(sqrt($j/30)));
        $i->multColor(1.0, 1.0, 1.0, $j/30);
        $m->nextFrame();
    }

    for($j=0; $j<=30; ++$j) {
        $i->scaleTo(sqrt(sqrt(1+($j/30))));
        $i->multColor(1.0, 1.0, 1.0, (30-$j)/30);
        $m->nextFrame();
    }
```

```
    $m->remove($i);

}

header('Content-type: application/x-shockwave-flash');
$m->output();
?>
```

Output:

Super-cool Dynamic Image Generator

Want to be cooler than all your friends? Well here it is!

First, set up an ErrorDocument 404 handler for your images directory.

```
<Directory /home/doc_root/images>
    ErrorDocument 404 /images/generate.php
</Directory>')
```

Then generate.php looks like this:

```
<?php
$filename = basename($_SERVER['REDIRECT_URL']);
if(preg_match('/^([^_]*?)_([^_]*?)_([^_]*?)\.(.*?)$/',$filename, $reg)) {
    $type = $reg[1];
    $text = $reg[2];
    $rgb = $reg[3];
    $ext = $reg[4];
}

if(strlen($rgb)==6) {
    $r = hexdec(substr($rgb,0,2));
    $g = hexdec(substr($rgb,2,2));
    $b = hexdec(substr($rgb,4,2));
} else $r = $g = $b = 0;

    switch(strtolower($ext)) {
        case 'jpg':
            Header("Content-Type: image/jpg");
            break;
        case 'png':
        case 'gif': /* We don't do gif - send a png instead */
            Header("Content-Type: image/png");
            break;
        default:
            break;
    }

    switch($type) {
        case 'solid':
            $im = imagecreatetruecolor(80,80);
            $bg = imagecolorallocate($im, $r, $g, $b);
            imagefilledrectangle($im,0,0,80,80,$bg);
            break;
        case 'button':
            $si = 32; $font = "php";
            $im = imagecreatefrompng('blank_wood.png');
            $tsize = imagettfbbox($si,0,$font,$text);
            $dx = abs($tsize[2]-$tsize[0]);
            $dy = abs($tsize[5]-$tsize[3]);
            $x = ( imagesx($im) - $dx ) / 2;
            $y = ( imagesy($im) - $dy ) / 2 + $dy;
            $white = ImageColorAllocate($im,255,255,255);
            $black = ImageColorAllocate($im,$r,$g, $b);
            ImageTTFText($im, $si, 0, $x, $y, $white, $font, $text);
            ImageTTFText($im, $si, 0, $x+2, $y, $white, $font, $text);
            ImageTTFText($im, $si, 0, $x, $y+2, $white, $font, $text);
            ImageTTFText($im, $si, 0, $x+2, $y+2, $white, $font, $text);
            ImageTTFText($im, $si, 0, $x+1, $y+1, $black, $font, $text);
        break;
    }
    Header("HTTP/1.1 200 OK");
    $dest_file = dirname($_SERVER['SCRIPT_FILENAME']).'/'.$filename;
    switch(strtolower($ext)) {
```

```
            case 'png':
            case 'gif':
                @ImagePNG($im,$dest_file);
                ImagePNG($im);
                break;
            case 'jpg':
                @ImageJPEG($im,$dest_file);
                ImageJPEG($im);
                break;
        }
    ?>
```

The URL, http://localhost/images/button_test_000000.png produces this image:

Starting a Session

To start a session use session_start() and to register a variable in this session use the $_SESSION array.

```
<?php
  session_start();
  $_SESSION['my_var'] = 'Hello World';
?>
```

If register_globals is enabled then your session variables will be available as normal variables on subsequent pages. Otherwise they will only be in the $_SESSION array.

```
<?php
  session_start();
  echo $_SESSION['my_var'];
?>
```

Session Configuration

Default session settings are set in your php.ini file:

```
session.save_handler = files    ; Flat file backend
session.save_path=/tmp          ; where to store flat files
session.name = PHPSESSID        ; Name of session (cookie name)
session.auto_start = 0          ; init session on req startup
session.use_cookies = 1         ; whether cookies should be used
session.use_only_cookies = 0    ; force only cookies to be used
session.cookie_lifetime = 0     ; 0 = session cookie
session.cookie_path = /         ; path for which cookie is valid
session.cookie_domain =         ; the cookie domain
session.serialize_handler = php ; serialization handler (wddx|php)
session.gc_probability = 1      ; garbage collection prob.
session.gc_dividend    = 100    ; If 100, then above is in %
session.gc_maxlifetime = 1440   ; garbage collection max lifetime
session.referer_check =         ; filter out external URL\'s
session.entropy_length = 0      ; # of bytes from entropy source
session.entropy_file =          ; addtional entropy source
session.use_trans_sid = 1       ; use automatic url rewriting
url_rewriter.tags = "a=href,area=href,frame=src,input=src"
session.cache_limiter = nocache ; Set cache-control headers
session.cache_expire = 180      ; expiry for private/public caching
```

Cache-control is important when it comes to sessions. You have to be careful that end-user client caches aren't caching invalid pages and also that intermediary proxy-cache mechanisms don't sneak in and cache pages on you. When cache-limiter is set to the default, no-cache, PHP generates a set of response headers that look like this:

```
HTTP/1.1 200 OK
Date: Sat, 10 Feb 2001 10:21:59 GMT
Server: Apache/1.3.13-dev (Unix) PHP/4.0.5-dev
X-Powered-By: PHP/4.0.5-dev
Set-Cookie: PHPSESSID=9ce80c83b00a4aefb384ac4cd85c3daf; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Connection: close
Content-Type: text/html
```

For cache_limiter = private the cache related headers look like this:

```
Set-Cookie: PHPSESSID=b02087ce4225987870033eba2b6d78c3; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: private, max-age=10800, pre-check=10800
```

For cache_limiter = public they look like this:

```
Set-Cookie: PHPSESSID=37421e3d0283c667f75481745b25b9ad; path=/
Expires: Tue, 12 Feb 2001 13:57:16 GMT
Cache-Control: public, max-age=10800
```

Custom Backend

You can change the session backend datastore from a script using session_module_name().

```
<?php
  session_module_name("files");  // ASCII files

  session_module_name("mm");     // Shared memory

  session_module_name("user");   // Custom session backend
?>
```

You can also define your own custom session backend datastore using the session_set_save_handler() function.

```
<?php
  session_set_save_handler("myOpen", "myClose",
                           "myRead", "myWrite",
                           "myDestroy", "myGC");
?>
```

You would then write these 6 functions.

Custom Backend

Let's have a look at an actual custom session backend. This uses MySQL to store the session data. We could set these right in the script, but let's make use of Apache's httpd.conf file to set our custom save handler for a portion of our web site.

```
<Directory "/var/html/test">
    php_value session.save_handler user
    php_value session.save_path mydb
    php_value session.name sessions
</Directory>
```

The MySQL schema looks like this:

```
CREATE TABLE sessions (
    id char(32) NOT NULL,
    data text,
    ts timestamp,
    PRIMARY KEY (id)
)
```

We can now write our handler. It looks like this:

```
<?php
function open($db,$name) {
    global $table;
    mysql_connect('localhost');
    mysql_select_db($db);
    $table = $name;
    return true;
}

function close() {
    mysql_close();
    return true;
}

function read($id) {
    global $table;
    $result = mysql_query("select data from $table where id='$id'");
    if($result && mysql_num_rows($result)) {
        return mysql_result($result,0);
    } else {
        error_log("read: ".mysql_error()."\n",3,"/tmp/errors.log");
        return "";
    }
}

function write($id, $data) {
    global $table;
    $data = addslashes($data);
    mysql_query("replace into $table (id,data) values('$id','$data')")
        or error_log("write: ".mysql_error()."\n",3,"/tmp/errors.log");
    return true;
}

function destroy($id) {
    global $table;
    mysql_query("delete from $table where where id='$id'");
}

function gc($max_time) {
    global $table;
    mysql_query(
      "delete from $table where UNIX_TIMESTAMP(ts)<UNIX_TIMESTAMP()-$max_time")
```

```
      or error_log("gc: ".mysql_error()."\n",3,"/tmp/errors.log");
    return true;
}

session_set_save_handler('open','close','read','write','destroy','gc');
?>
```

Our PHP files under /var/html/test then simply need to look something like this:

```
<?php
  require 'handler.php';

  session_start();
  session_register('var');
  $var = "Hello World";
?>
```

register_globals

PHP automatically creates global variables containing data from a variety of external sources. This feature can be turned off by turning off the register_globals setting. With register_globals you can access this data via a number of special associative arrays listed below.

$_GET['foo']='bar'

```
http://www.php.net/index.php?foo=bar
```

$_POST['foo']='bar'

```
<form action="script.php" method="POST">
<input type="text" name="foo" value="bar">
</form>
```

$_COOKIE['foo']='bar'

```
<?php
    SetCookie('foo','bar');
?>
```

$_REQUEST['foo']='bar'

```
<?php
    SetCookie('foo','bar');
?>
```

$_SERVER

Special variables set by your web server. You can get a list of what is set by running this code on your server:

```
<?php
foreach($_SERVER as $key=>$val) {
    echo '$_SERVER['.$key."] = $val<br>\n";
}
?>
$_SERVER[DOCUMENT_ROOT] = /home/rasmus/phpweb
$_SERVER[HTTP_ACCEPT] = text/xml,application/xml,application/xht...
$_SERVER[HTTP_ACCEPT_CHARSET] = ISO-8859-1, utf-8;q=0.66, *;q=0.66
$_SERVER[HTTP_ACCEPT_ENCODING] = gzip, deflate, compress;q=0.9
$_SERVER[HTTP_ACCEPT_LANGUAGE] = en-us, en;q=0.50
$_SERVER[HTTP_CACHE_CONTROL] = max-age=0
$_SERVER[HTTP_CONNECTION] = keep-alive
$_SERVER[HTTP_COOKIE] = dims=1014_690; PHPSESSID=c1c486d22f970af...
$_SERVER[HTTP_HOST] = localhost
$_SERVER[HTTP_KEEP_ALIVE] = 300
$_SERVER[HTTP_REFERER] = http://localhost/pres2/index.php/PHP
$_SERVER[HTTP_USER_AGENT] = Mozilla/5.0 (X11; U; Linux i686; en-US; ...
$_SERVER[PATH] = /usr/local/bin:/bin:/usr/bin:/usr/sbin:/...
$_SERVER[REMOTE_ADDR] = 127.0.0.1
$_SERVER[REMOTE_PORT] = 1099
$_SERVER[SCRIPT_FILENAME] = /home/rasmus/phpweb/pres2/show.php
$_SERVER[SERVER_ADDR] = 127.0.0.1
$_SERVER[SERVER_ADMIN] = rasmus@lerdorf.com
$_SERVER[SERVER_NAME] = localhost
$_SERVER[SERVER_PORT] = 80
$_SERVER[SERVER_SIGNATURE] = Apache/1.3.28-dev Server at localhost Po...
$_SERVER[SERVER_SOFTWARE] = Apache/1.3.28-dev (Unix) PHP/4.4.0-dev
$_SERVER[GATEWAY_INTERFACE] = CGI/1.1
$_SERVER[SERVER_PROTOCOL] = HTTP/1.1
```

```
$_SERVER[REQUEST_METHOD] = GET
$_SERVER[QUERY_STRING] =
$_SERVER[REQUEST_URI] = /pres2/show.php/veracruz
$_SERVER[SCRIPT_NAME] = /pres2/show.php
$_SERVER[PATH_INFO] = /veracruz
$_SERVER[PATH_TRANSLATED] = /home/rasmus/phpweb/veracruz
$_SERVER[PHP_SELF] = /pres2/show.php/veracruz
$_SERVER[argv] = Array
$_SERVER[argc] = 0
```

$_ENV

Environment variables that were present at server startup time. Note that environment variables created by PHP using putenv() will not be shown here, nor do they persist beyond the request.

```
$_ENV[MANPATH] = /usr/man:/usr/local/man:/usr/share/man
$_ENV[SUPPORTED] = en_US:en
$_ENV[SSH_AGENT_PID] = 855
$_ENV[HOSTNAME] = thinkpad.lerdorf.com
$_ENV[HOST] = thinkpad.lerdorf.com
$_ENV[TERM] = xterm
$_ENV[SHELL] = /bin/tcsh
$_ENV[GTK_RC_FILES] = /etc/gtk/gtkrc:/home/rasmus/.gtkrc-1.2-g...
$_ENV[WINDOWID] = 27263119
$_ENV[QTDIR] = /usr/lib/qt3-gcc3.2
$_ENV[BK_HOST] = mysql.com
$_ENV[USER] = root
$_ENV[GROUP] = rasmus
$_ENV[LS_COLORS] = no=00:fi=00:di=00;34:ln=00;36:pi=40;33:s...
$_ENV[SUDO_USER] = rasmus
$_ENV[GDK_USE_XFT] = 1
$_ENV[SUDO_UID] = 500
$_ENV[HOSTTYPE] = i386-linux
$_ENV[SSH_AUTH_SOCK] = /tmp/ssh-XXpcCouR/agent.843
$_ENV[SESSION_MANAGER] = local/thinkpad.lerdorf.com:/tmp/.ICE-uni...
$_ENV[PAGER] = /usr/bin/less
$_ENV[BK_LICENSE] = ACCEPTED
$_ENV[BK_USER] = rasmus
$_ENV[PATH] = /usr/local/bin:/bin:/usr/bin:/usr/sbin:/...
$_ENV[MAIL] = /var/mail/rasmus
$_ENV[PWD] = /home/rasmus/phpweb/pres2/presentations
$_ENV[XMODIFIERS] = @im=none
$_ENV[EDITOR] = vi
$_ENV[LANG] = en_US
$_ENV[NPROMPT] = %{ [34;1m%}%t %m:%~>%{ [0m%} >
$_ENV[SSH_ASKPASS] = /usr/libexec/openssh/gnome-ssh-askpass
$_ENV[SUDO_COMMAND] = /bin/tcsh
$_ENV[HOME] = /home/rasmus
$_ENV[SHLVL] = 5
$_ENV[OSTYPE] = linux
$_ENV[GNOME_DESKTOP_SESSION_ID] = Default
$_ENV[VENDOR] = intel
$_ENV[LOGNAME] = root
$_ENV[MACHTYPE] = i386
$_ENV[MOZILLA_XFT] = 1
$_ENV[VISUAL] = vi
$_ENV[LESSOPEN] = |/usr/bin/lesspipe.sh %s
$_ENV[SUDO_GID] = 500
$_ENV[DISPLAY] = :0
$_ENV[G_BROKEN_FILENAMES] = 1
$_ENV[COLORTERM] = gnome-terminal
$_ENV[XAUTHORITY] = /home/rasmus/.Xauthority
$_ENV[_] = /usr/sbin/httpd
```

$_FILES

Used for the RFC1867 file upload feature.

```
$_FILES['userfile']['name']
```

```
$_FILES['userfile']['type']
$_FILES['userfile']['size']
$_FILES['userfile']['tmp_name']
```

$HTTP_RAW_POST_DATA

When the mime type associated with the POST data is unrecognized or not set, the raw post data is available in this variable.

Safe Mode

Safe Mode is an attempt to solve the shared-server security problem. It is architecturally incorrect to try to solve this problem at the PHP level, but since the alternatives at the web server and OS levels aren't very realistic, many people, especially ISP's, use safe mode for now.

The configuration directives that control safe mode are:

```
safe_mode = Off
open_basedir =
safe_mode_exec_dir =
safe_mode_allowed_env_vars = PHP_
safe_mode_protected_env_vars = LD_LIBRARY_PATH
disable_functions =
```

When safe_mode is on, PHP checks to see if the owner of the current script matches the owner of the file to be operated on by a file function.

For example:

```
-rw-rw-r--    1 rasmus   rasmus       33 Jul  1 19:20 script.php
-rw-r--r--    1 root     root       1116 May 26 18:01 /etc/passwd
```

Running this script.php

```
<?php
readfile('/etc/passwd');
?>
```

results in this error when safe mode is enabled:

```
<b>Warning</b>:  SAFE MODE Restriction in effect.  The script whose uid is 500 is
not allowed to access /etc/passwd owned by uid 0 in <b>/docroot/script.php</b> on
line <b>2</b>
```

If instead of safe_mode, you set an open_basedir directory then all file operations will be limited to files under the specified directory. For example (Apache httpd.conf example):

```
<Directory /docroot>
php_admin_value open_basedir /docroot
</Directory>
```

If you run the same script.php with this open_basedir setting then this is the result:

```
<b>Warning</b>: open_basedir restriction in effect. File is in wrong directory in
<b>/docroot/script.php</b> on line <b>2</b>
```

You can also disable individual functions. If we add this to our php.ini file:

```
disable_functions readfile,system
```

Then we get this output:

```
<b>Warning</b>:  readfile() has been disabled for security reasons in
<b>/docroot/script.php</b> on line <b>2</b>
```

Watch for uninitialized variables

```
<?php
    if($user=='rasmus') {
        $ok = true;
    }

    if($ok) {
        echo "$user logged in";
    }
?>
```

Catch these by setting the error_reporting level to E_ALL. The above script would generate this warning (assuming $user is set):

```
<b>Warning</b>:  Undefined variable:  ok in <b>script.php</b> on line <b>6</b>
```

You can of course also turn off register_globals, but that addresses the symptom rather than the problem.

Never trust user data!

```
<?php
    readfile($filename);
?>
```

Turning off register_globals doesn't make this any more secure. The script would instead look like this:

```
<?php
    readfile($HTTP_POST_VARS['filename']);
?>
```

The only way to secure something like this is to be really paranoid about cleaning user input. In this case if you really want the user to be able to specify a filename that gets used in any of PHP's file functions, do something like this:

```
<?php
    $doc_root = $HTTP_SERVER_VARS['DOCUMENT_ROOT'];
    $filename = realpath($filename);
    readfile($doc_root.$filename);
?>
```

You may also want to strip out any path and only take the filename component. An easy way to do that is to use the basename() function. Or perhaps check the extension of the file. You can get the extension using this code:

```
<?php
    $ext = substr($str,strrpos($str,'.'));
?>
```

Again, never trust user data!

```
<?php
    system("ls $dir");
?>
```

In this example you want to make sure that the user can't pass in $dir set to something like: ".;cat /etc/passwd" The remedy is to use escapeshellarg() which places the argument inside single quotes and escapes any single quote characters in the string.

```
<?php
    $dir=escapeshellarg($dir);
    system("ls $dir");
?>
```

Beyond making sure users can't pass in arguments that executes other system calls, make sure that the argument itself is ok and only accesses data you want the users to have access to.

Many users place code in multiple files and include these files:

```
<?php
    require 'functions.inc';
?>
```

Or perhaps

```
<?php
    require 'functions.php';
?>
```

Both of these can be problematic if the included file is accessible somewhere under the DOCUMENT_ROOT directory. The best solution is to place these files outside of the DOCUMENT_ROOT directory where they are not accessible directly. You can add this external directory to your include_path configuration setting.

Another option is to reject any direct requests for these files in your Apache configuration. You can use a rule like this in your "httpd.conf" file:

```
<Files ~ "\.inc$">
    Order allow,deny
    Deny from all
</Files>
```

Take this standard file upload form:

```
<FORM ENCTYPE="multipart/form-data" ACTION="upload.php" METHOD=POST>
<INPUT TYPE="hidden" name="MAX_FILE_SIZE" value="100000">
Send this file: <INPUT NAME="myfile" TYPE="file">
<INPUT TYPE="submit" VALUE="Send File">
</FORM>
```

The correct way to put the uploaded file in the right place:

```
<?php
    /* Not under DOCUMENT_ROOT */
    $destination = "/some/path/$myfile_name";

    move_uploaded_file($myfile, $destination);
?>
```

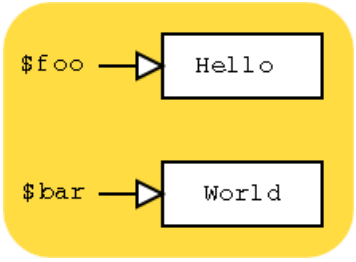
If you are uploading files to be placed somewhere under the DOCUMENT_ROOT then you need to be very paranoid in checking what you are putting there. For example, you wouldn't want to let people upload arbitrary PHP scripts that they can then browse to in order to execute them. Here we get paranoid about checking that only image files can be uploaded. We even look at the contents of the file and ensure that the file extension matches the content.

```
<?php
    $type = $HTTP_POST_FILES['myfile']['type'];
    $file = $HTTP_POST_FILES['myfile']['tmp_name'];
    $name = $HTTP_POST_FILES['myfile']['name'];
    $types = array(0,'.gif','.jpg','.png','.swf');
    list(,,$type) = getimagesize($file);
    if($type) {
        $name = substr($name,0,strrpos($str,'.'));
        $name .= $types[$type];
    }
    move_uploaded_file($myfile, "$DOCUMENT_ROOT/images/$name");
?>
```

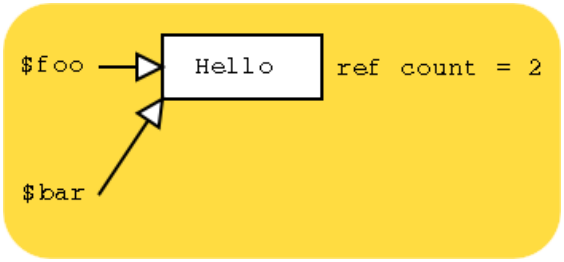
References

References are not pointers!

```
<?php
    $foo = 'Hello';
    $bar = 'World';
?>
```




```
<?php
    $bar = & $foo;
?>
```



Returning References

Passing arguments to a function by reference

```
<?php
function inc(& $b) {
    $b++;
}
$a = 1;
inc($a);
echo $a;
?>
```

Output:

```
2
```

A function may return a reference to data as opposed to a copy

```
<?php
function & get_data() {
    $data = "Hello World";
    return $data;
}
$foo = & get_data();
?>
```

Adding an extension

Problem

You need PHP's built-in ftp functions for the ultra-cool script you are writing, but your service provider does not have PHP compiled with the --enable-ftp option.

Solution

If you have a shell account on a system with the same operating system as your web server, grab the PHP source tarball and build using:

```
--with-apxs --enable-ftp=shared
```

You can check which flags your provider used by putting a phpinfo() call in a script on your server.

```
<?phpinfo()?>
```

Once compiled, you will find a "modules/ftp.so" file which you can copy to your web server and enable either by putting:

```
extension=ftp.so
```

in your php.ini file or by adding this to the top of your script:

```
<?php dl("ftp.so") ?>
```

$PATH_INFO

$PATH_INFO is your friend when it comes to creating clean URLS. Take for example this URL:

```
http://www.company.com/products/routers
```

If the Apache configuration contains this block:

```
<Location "/products">
  ForceType application/x-httpd-php
</Location>
```

Then all you have to do is create a PHP script in your DOCUMENT_ROOT named 'products' and you can use the $PATH_INFO variable which will contain the string, '/routers', to make a DB query.

ErrorDocument

Apache's ErrorDocument directive can come in handy. For example, this line in your Apache configuration file:

```
ErrorDocument 404 /error.php
```

Can be used to redirect all 404 errors to a PHP script. The following server variables are of interest:

- $REDIRECT_ERROR_NOTES - File does not exist: /docroot/bogus
- $REDIRECT_REQUEST_METHOD - GET
- $REDIRECT_STATUS - 404
- $REDIRECT_URL - /docroot/bogus

Don't forget to send a 404 status if you choose not to redirect to a real page.

```
<? Header('HTTP/1.0 404 Not Found'); ?>
```

Interesting uses

- Search for closest matching valid URL and redirect
- Use attempted url text as a DB keyword lookup
- Funky caching

An interesting way to handle caching is to have all 404's redirected to a PHP script.

```
ErrorDocument 404 /generate.php
```

Then in your generate.php script use the contents of $REDIRECT_URI to determine which URL the person was trying to get to. In your database you would then have fields linking content to the URL they affect and from that you should be able to generate the page. Then in your generate.php script do something like:

```
<?php
    $s = $REDIRECT_URI;
    $d = $DOCUMENT_ROOT;
    // determine requested uri
    $uri = substr($s, strpos($s,$d) + strlen($d) + 1);
    ob_start();  // Start buffering output
    // ... code to fetch and output content from DB ...
    $data = ob_get_contents();
    $fp = fopen("$DOCUMENT_ROOT/$uri",'w');
    fputs($fp,$data);
    fclose($fp);
    ob_end_flush(); // Flush and turn off buffering
?>
```

So, the way it works, when a request comes in for a page that doesn't exist, generate.php checks the database and determines if it should actually exist and if so it will create it and respond with this generated data. The next request for that same URL will get the generated page directly. So in order to refresh your cache you simply have to delete the files.

Variable variables

A variable variable looks like this: $$var

So, if $var = 'foo' and $foo = 'bar' then $$var would contain the value 'bar' because $$var can be thought of as $'foo' which is simply $foo which has the value 'bar'.

Variable variables sound like a cryptic a useless concept, but they can be useful sometimes. For example, if we have a configuration file consisting of configuration directives and values in this format:

```
foo=bar
abc=123
```

Then it is very easy to read this file and create corresponding variables:

```
<?php
$fp = fopen('config.txt','r');
while(true) {
    $line = fgets($fp,80);
    if(!feof($fp)) {
        if($line[0]=='#' || strlen($line)<2) continue;
        list($name,$val)=explode('=',$line,2);
        $$name=trim($val);
    } else break;
}
fclose($fp);
?>
```

Along the same lines as variable variables, you can create compound variables and variable functions.

```
<?php
  $str = 'var';
  $var_toaster = "Hello World";
  echo ${$str.'_toaster'};

  $str();  // Calls a function named var()
  ${$str.'_abc'}();  // Calls a function named var_abc()
?>
```

Optimization

Don't use a regex if you don't have to

PHP has a rich set of string manipulation functions - use them!

```
BAD: <?  $new = ereg_replace("-","_",$str); ?>

GOOD:<?  $new = str_replace("-","_",$str);  ?>

BAD: <?  preg_match('/(\..*?)$/',$str,$reg);?>

GOOD:<?  substr($str,strrpos($str,'.'));    ?>
```

Use References if you are passing large data structs around to save memory

There is a tradeoff here. Manipulating references is actually a bit slower than making copies of your data, but with references you will be using less memory. So you need to determine if you are cpu or memory bound to decide whether to go through and look for places to pass references to data instead of copies.

Use Persistent Database connections

Some database are slower than others at establising new connections. The slower it is, the more of an impact using persistent connections will have. But, keep in mind that persistent connections will sit and tie up resources even when not in use. Watch your resource limits as well. For example, by default Apache's

Using MySQL? Check out mysql_unbuffered_query()

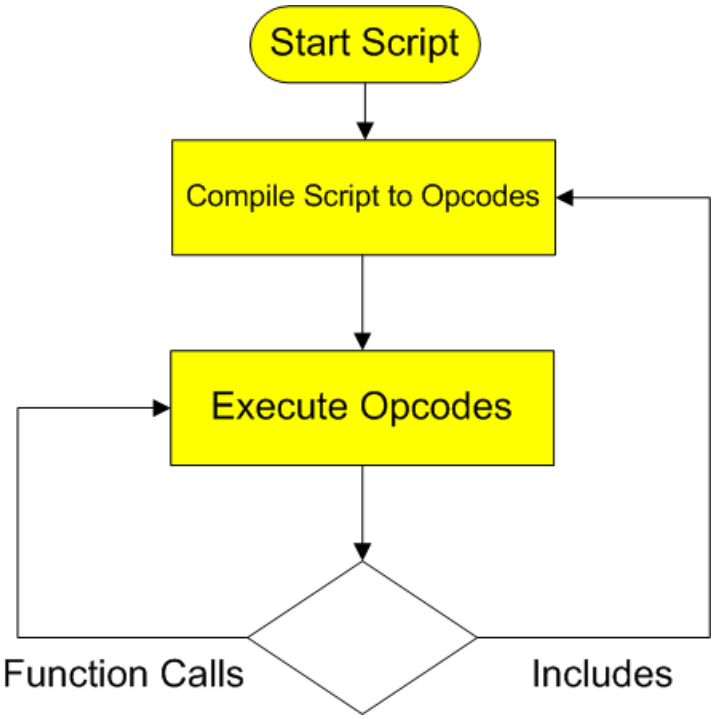
Use it exactly like you would mysql_query(). The difference is that instead of waiting for the entire query to finish and storing the result in the client API, an unbuffered query makes results available to you as soon as possible and they are not allocated in the client API. You potentially get access to your data quicker, use a lot less memory, but you can't use mysql_num_rows() on the result resource and it is likely to be slightly slower for small selects.

Hey Einstein!

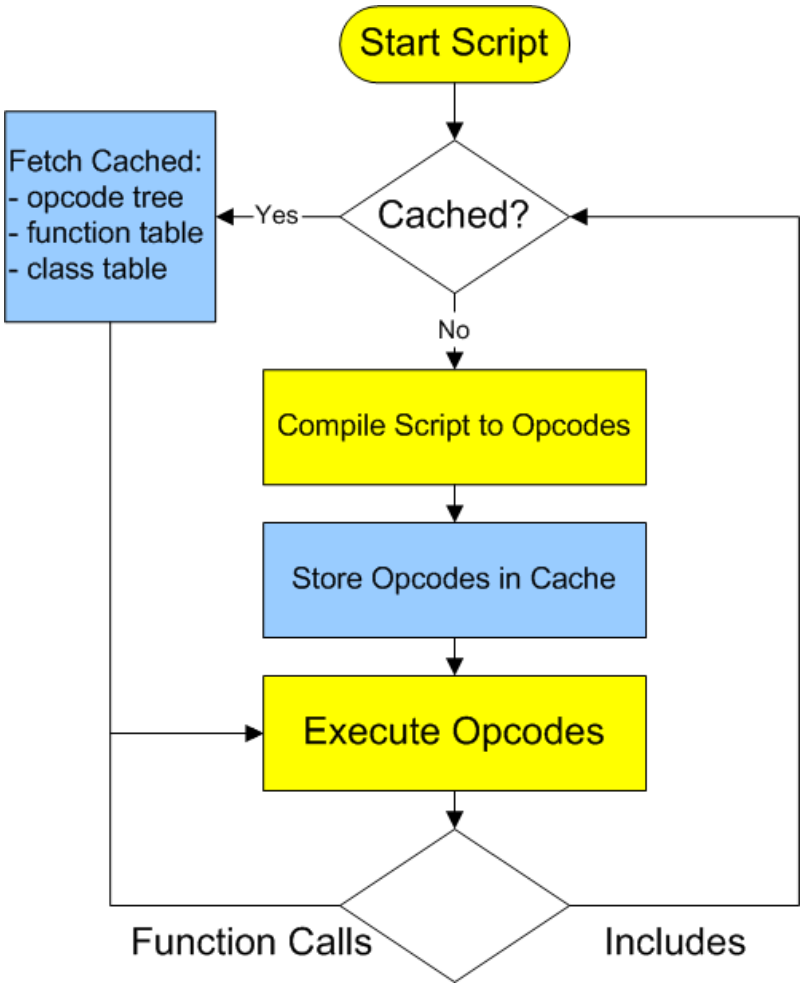
Don't over-architect things. If your solution seems complex to you, there is probably a simpler and more obvious approach. Take a break from the computer and go out into the big (amazingly realistic) room and think about something else for a bit.

PHP Opcode Caches

Standard PHP



PHP with an Opcode Cache


Start Script
Cached?
Yes
Fetch Cached:
- opcode tree
- function table
- class table
No
Compile Script to Opcodes
Store Opcodes in Cache
Execute Opcodes
Function Calls
Includes

There are a number of them out there.

- APC - open source
- IonCube Accelerator - free, but closed source
- Zend Cache - commercial

Installation

Typically very trivial. For IonCube, for example, in your php.ini add:

```
zend_extension = "/usr/local/lib/php/php_accelerator_1.3.3r2.so"
```

So why isn't this just built into standard PHP?

This gets asked often, and although not a good answer, the answer is that since it wasn't in PHP from the start a number of competing plugin cache systems were developed and choosing one over another at this point would effectively chop these projects, both commercial and free, off at their knees. This way they can compete and innovate against each other at the cost of duplicated effort.

Why Profile?

Because your assumptions of how things work behind the scenes are not always correct. By profiling your code you can identify where the bottlenecks are quantitatively.

How?

PEAR/Pecl to the rescue!

```
www:~> pear install apd
downloading apd-0.4p1.tgz ...
...done: 39,605 bytes
16 source files, building
running: phpize
PHP Api Version        : 20020918
Zend Module Api No     : 20020429
Zend Extension Api No  : 20021010
building in /var/tmp/pear-build-root/apd-0.4p1
running: /tmp/tmprFlAqf/apd-0.4p1/configure
running: make
apd.so copied to /tmp/tmprFlAqf/apd-0.4p1/apd.so
install ok: apd 0.4p1
```

Woohoo!

```
www:~> pear info apd
About apd-0.4p1
===============
+-----------------+-------------------------------------------------+
| Package         | apd                                             |
| Summary         | A full-featured engine-level profiler/debugger  |
| Description     | APD is a full-featured profiler/debugger that is |
|                 | loaded as a zend_extension. It aims to be an    |
|                 | analog of C's gprof or Perl's Devel::DProf.     |
| Maintainers     | George Schlossnagle <george@omniti.com> (lead)  |
| Version         | 0.4p1                                           |
| Release Date    | 2002-11-25                                      |
| Release License | PHP License                                     |
| Release State   | stable                                          |
| Release Notes   | Fix for pre-4.3 versions of php                 |
| Last Modified   | 2002-12-02                                      |
+-----------------+-------------------------------------------------+
```

```
www:~> pear config-show
Configuration:
==============
+----------------------+-----------------+-----------------------------------+
| PEAR executables     | bin_dir         | /usr/local/bin                    |
| directory            |                 |                                   |
| PEAR documentation   | doc_dir         | /usr/local/lib/php/docs           |
| directory            |                 |                                   |
| |ffff11|PHP extension|        | |ffff11|ext_dir|         |
|ffff11|/usr/local/lib/php/extensions/no-de| |
| |ffff11|directory|            |                 | |ffff11|bug-non-zts-20020429|
          |
| PEAR directory       | php_dir         | /usr/local/lib/php                |
| PEAR Installer cache | cache_dir       | /tmp/pear/cache                   |
| directory            |                 |                                   |
| PEAR data directory  | data_dir        | /usr/local/lib/php/data           |
| PEAR test directory  | test_dir        | /usr/local/lib/php/tests          |
| Cache TimeToLive     | cache_ttl       | <not set>                         |
| Preferred Package    | preferred_state | stable                            |
```

```
| State                  |                 |                                      |
| Unix file mask         | umask           | 18                                   |
| Debug Log Level        | verbose         | 1                                    |
| HTTP Proxy Server      | http_proxy      | <not set>                            |
| Address                |                 |                                      |
| PEAR server            | master_server   | pear.php.net                         |
| PEAR password (for     | password        | <not set>                            |
| maintainers)           |                 |                                      |
| PEAR username (for     | username        | <not set>                            |
| maintainers)           |                 |                                      |
+------------------------+-----------------+--------------------------------------+
```

```
www:~> cd /usr/local/lib/php
www:/usr/local/lib/php> ln -s extensions/no-debug-non-zts-20020429/apd.so apd.so
```

Then in your php.ini file:

```
zend_extension = "/usr/local/lib/php/apd.so"
apd.dumpdir = /tmp
```

It isn't completely transparent. You need to tell the profiler when to start profiling. At the top of a script you want to profile, add this call:

```
<?php
apd_set_pprof_trace();
?>
```

The use the command-line tool called pprofp:

```
wwww: ~> pprofp
pprofp <flags> <trace file>
    Sort options
    -a          Sort by alphabetic names of subroutines.
    -l          Sort by number of calls to subroutines
    -m          Sort by memory used in a function call.
    -r          Sort by real time spent in subroutines.
    -R          Sort by real time spent in subroutines (inclusive of child calls).
    -s          Sort by system time spent in subroutines.
    -S          Sort by system time spent in subroutines (inclusive of child
calls).
    -u          Sort by user time spent in subroutines.
    -U          Sort by user time spent in subroutines (inclusive of child calls).
    -v          Sort by average amount of time spent in subroutines.
    -z          Sort by user+system time spent in subroutines. (default)

    Display options
    -c          Display Real time elapsed alongside call tree.
    -i          Suppress reporting for php builtin functions
    -O <cnt>    Specifies maximum number of subroutines to display. (default 15)
    -t          Display compressed call tree.
    -T          Display uncompressed call tree.


www: ~> ls -latr /tmp/pprofp.*
-rw-r--r--    1 nobody   nobody      16692 Dec  3 01:19 /tmp/pprof.04545

www: ~> pprofp -z /tmp/pprof.04545


Trace for /home/rasmus/phpweb/index.php
Total Elapsed Time =    0.69
Total System Time  =    0.01
Total User Time    =    0.08


         Real         User        System             secs/    cumm
%Time (excl/cumm)  (excl/cumm)  (excl/cumm) Calls    call    s/call  Memory Usage
Name
--------------------------------------------------------------------------------------
 33.3  0.11  0.13   0.02  0.03   0.01  0.01     7   0.0043    0.0057       298336
require_once
 22.2  0.02  0.02   0.02  0.02   0.00  0.00   183   0.0001    0.0001       -33944
feof
 11.1  0.01  0.01   0.01  0.01   0.00  0.00     3   0.0033    0.0033       -14808
define
 11.1  0.04  0.04   0.01  0.01   0.00  0.00   182   0.0001    0.0001       112040
fgetcsv
```

```
 11.1  0.25  0.25   0.01  0.01   0.00  0.00     6   0.0017    0.0017         3768
getimagesize
 11.1  0.01  0.01   0.01  0.01   0.00  0.00    55   0.0002    0.0002         2568
sprintf
  0.0  0.00  0.00   0.00  0.00   0.00  0.00     7   0.0000    0.0000         -136
printf
  0.0  0.00  0.00   0.00  0.00   0.00  0.00     1   0.0000    0.0000          136
htmlspecialchars
  0.0  0.00  0.00   0.00  0.00   0.00  0.00     1   0.0000    0.0000          -16
mirror_provider_url
  0.0  0.00  0.00   0.00  0.00   0.00  0.00     7   0.0000    0.0000          112
spacer
  0.0  0.00  0.00   0.00  0.00   0.00  0.00    10   0.0000    0.0000         -552
delim
  0.0  0.00  0.00   0.00  0.00   0.00  0.00     1   0.0000    0.0000          112
mirror_provider
  0.0  0.00  0.00   0.00  0.00   0.00  0.00    20   0.0000    0.0000         -624
print_link
  0.0  0.00  0.00   0.00  0.00   0.00  0.00     1   0.0000    0.0000           24
have_stats
  0.0  0.00  0.00   0.00  0.00   0.00  0.00     1   0.0000    0.0000          -72
make_submit
  0.0  0.00  0.00   0.00  0.00   0.00  0.00     2   0.0000    0.0000          112
strrchr
  0.0  0.08  0.08   0.00  0.00   0.00  0.00     2   0.0000    0.0000          168
filesize
  0.0  0.00  0.00   0.00  0.00   0.00  0.00     1   0.0000    0.0000          -16
commonfooter
  0.0  0.00  0.11   0.00  0.00   0.00  0.00     2   0.0000    0.0000            0
download_link
  0.0  0.00  0.25   0.00  0.01   0.00  0.00     6   0.0000    0.0017          208
make_image
```

debug_backtrace

debug_backtrace() is a new function in PHP 4.3

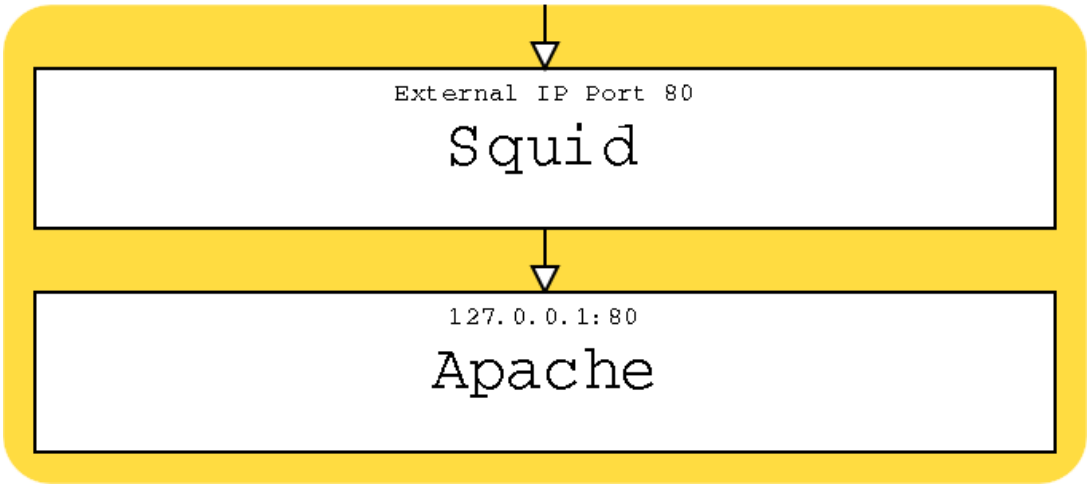
Custom error handler

```
<?
  function short($str) {
    if(strstr($str,'/'))
      return substr(strrchr($str,'/'),1);
    else return $str;
  }
  function myErrorHandler($errno,$errstr,$errfile,$errline)
  {
    echo "$errno: $errstr in ".short($errfile)." at line $errline<br />\n";
    echo "Backtrace<br />\n";
    $trace = debug_backtrace();
    foreach($trace as $ent) {
      if(isset($ent['file'])) $ent['file'].':';
      if(isset($ent['function'])) {
        echo $ent['function'].'(';
        if(isset($ent['args'])) {
          $args='';
          foreach($ent['args'] as $arg) { $args.=$arg.','; }
          echo rtrim(short($args),',');
        }
        echo ') ';
      }
      if(isset($ent['line'])) echo 'at line '.$ent['line'].' ';
      if(isset($ent['file'])) echo 'in '.short($ent['file']);
      echo "<br />\n";
    }
  }

  set_error_handler('myErrorHandler');
  include 'file2.php';
  test2(1,0);
?>
```

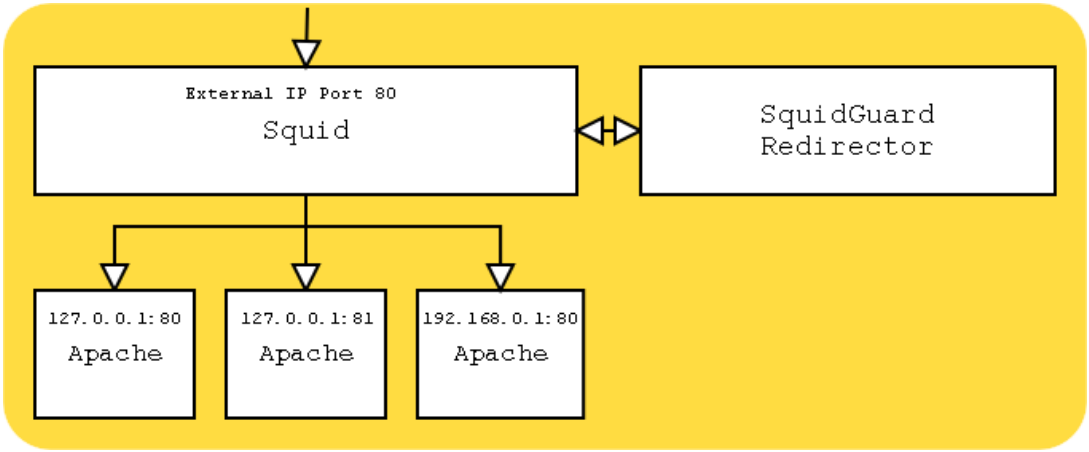
Custom error handler

```
<?
    function test1($b,$a) {
        $a/$b;
    }

    function test2($a,$b) {
        test1($b,$a);
    }
?>
```

For really busy sites, a reverse proxy like Squid is magical! Either run it as a single-server accelerator:



Or as a front-end cache to a number of local or remote servers:



Note:

Watch out for any use of $REMOTE_ADDR in your PHP scripts. Use $HTTP_X_FORWARDED_FOR instead.

Squid Configuration

Make it listen to port 80 on our external interface:

```
http_port 198.186.203.51:80
```

If we don't do cgi-bin stuff, comment these out:

```
#acl QUERY urlpath_regex cgi-bin
#no_cache deny QUERY
```

If we have plenty of RAM, bump this up a bit:

```
cache_mem 16MB
maximum_object_size 14096 KB
```

Specify where to store cached files (size in Megs, level 1 subdirs, level 2 subdirs)

```
cache_dir ufs /local/squid/cache 500 16 256
```

Get rid of the big store.log file:

```
cache_store_log none
```

Set our SNMP public community string:

```
acl snmppublic snmp_community public
```

Get rid of "allow all" and use list of hosts we are blocking (1 ip per line):

```
#http_access allow all
acl forbidden src "/local/squid/etc/forbidden"
http_access allow !forbidden
```

Set user/group squid should run as:

```
cache_effective_user squid
cache_effective_group daemon
```

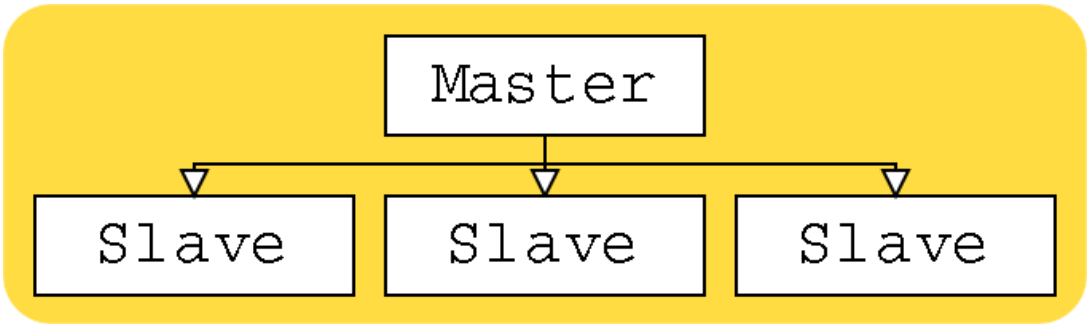
Single-server reverse proxy setup (set up Apache to listen to port 80 on the loopback):

```
httpd_accel_host 127.0.0.1
httpd_accel_port 80
httpd_accel_single_host on
httpd_accel_uses_host_header on
```

Only allow localhost access through snmp:

```
snmp_access allow snmppublic localhost
```

As of version 3.23.15 (try to use 3.23.29 or later), MySQL supports one-way replication. Since most web applications usually have more reads than writes, an architecture which distributes reads across multiple servers can be very beneficial.



In typical MySQL fashion, setting up replication is trivial. On your master server add this to your "my.cnf" file:

```
[mysqld]
log-bin
server-id=1
```

And add a replication user id for slaves to log in as:

```
GRANT FILE ON *.* TO repl@"%" IDENTIFIED BY 'foobar';
```

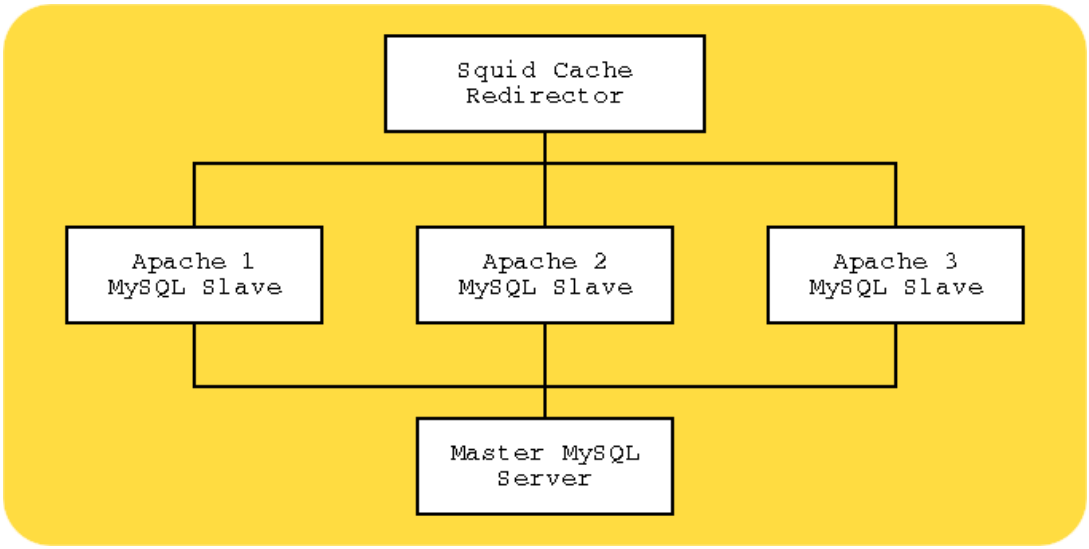
If you are using MySQL 4.0.2 or later, replace FILE with REPLICATION SLAVE in the above. Then on your slave servers:

```
[mysqld]
set-variable = max_connections=200
log-bin
master-host=192.168.0.1
master-user=repl
master-password=foobar
master-port=3306
server-id=2
```

Make sure each slave has its own unique server-id. And since these will be read-only slaves, you can start them with these options to speed them up a bit:

```
--skip-bdb --low-priority-updates
--delay-key-write-for-all-tables
```

Stop your master server. Copy the table files to each of your slave servers. Restart the master, then start all the slaves. And you are done. Combining MySQL replication with a Squid reverse cache and redirector and you might have an architecture like this:



You would then write your application to send all database writes to the master server and all reads to the local slave. It is also possible to set up two-way replication, but you would need to supply your own application-level logic to maintain atomicity of distributed writes. And you lose a lot of the advantages of this architecture if you do this as the writes would have to go to all the slaves anyway.

The Easy (And Expensive) Solution

You can buy dedicated boxes that do all sorts of advanced load balancing for you. Some popular ones are:

- Cisco CSS
- Foundry ServerIron
- Intel Netstructure
- F5 BIG-IP

If you are on Linux you could also try LVS. See www.LinuxVirtualServer.org.

There are two primary ways to configure Squid to be a load balancer.

Private /etc/hosts

The easiest is to simply list multiple ips for the httpd_accel_host in your /etc/hosts file and Squid will automatically round-robin across the backend servers.

Use cache_peer

This is more complex. Squid has advanced support for communicating with other caches. It just so happens that this communication happens over HTTP so you can set Squid up to treat the web servers you wish to load balance as peer caches. The configuration would look something like this:

```
httpd_accel_host www.visible-domain.com
httpd_accel_uses_host_header on
never_direct allow all
cache_peer server1 parent 80 0 no-query round-robin
cache_peer server2 parent 80 0 no-query round-robin
cache_peer server3 parent 80 0 no-query round-robin
```

no-query in the above tells Squid not to try to send ICP (Internet Cache Protocol) requests to the Apache servers. You could turn on the echo port on each server and redirect these ICP requests to there which would make this system automatically detect a server that is down.

Avoiding Redirectors

It is generally a good idea to avoid external redirectors. A lot of things can be done directly in Squid's config file. For example:

```
acl domain1 dstdomain www.domain1.com
acl domain2 dstdomain www.domain2.com
acl domain3 dstdomain www.domain3.com
cache_peer_access server1 allow domain1
cache_peer_access server2 allow domain2
cache_peer_access server3 allow domain3
cache_peer_access server1 deny all
cache_peer_access server2 deny all
cache_peer_access server3 deny all
```

This would configure Squid to send requests for pages on certain domains to certain backend servers. These backend servers could actually be aliases to different ips on the same server if you wanted to run multiple Apache instances on different ports on the same box.

Latest Developments

- PEAR and PECL
- SOAP
- Zend Engine 2
- New Object model
- Unified Constructors and Destructors
- Objects are references
- Exceptions
- User-space overloading
- SRM

Future

- Apache 2.0
- Extensions to talk to everything!
- php-soap
- Parrot

Resources

Home Page: http://www.php.net

Manual: http://php.net/manual

Tutorial: http://php.net/tut.php

Books: http://php.net/books.php

Index