PHP - Scripting the Web

November 2, 2002. USC, Los Angeles

Rasmus Lerdorf <rasmus@php.net>

http://lerdorf.com/socal.pdf

Simple Problem

HTTP is a stateless request-response plaintext protocol.

HTML files are simple and human-readable

Most web problems are actually very simple

A simple tool for a simple problem

Solution

A good solution should

- Have a shallow learning curve
- Instant gratification
- Build on what you know
- Great documentation
- Solve the simple problem easily
- Eliminate tedium
- Be able to solve even the most complex problem
- Be secure
- Steal/borrow existing technology
- Work everywhere

Bonus

- Be Free
- Teach the basics by not hiding the problem

The Good Old Days

Handling simple data coming from a form took something like this to do in C:

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>

#define ishex(x) (((x) >= '0' && (x) <= '9') || ((x) >= 'a' &&
                  (x) <= 'f') || ((x) >= 'A' && (x) <= 'F'))

int htoi(char *s) {
    int     value;
    char    c;

    c = s[0];
    if(isupper(c)) c = tolower(c);
    value=(c >= '0' && c <= '9' ? c - '0' : c - 'a' + 10) * 16;

    c = s[1];
    if(isupper(c)) c = tolower(c);
    value += c >= '0' && c <= '9' ? c - '0' : c - 'a' + 10;

    return(value);
}

void main(int argc, char *argv[]) {
    char *params, *data, *dest, *s, *tmp;
    char *name, *age;

    puts("Content-type: text/html\r\n");
    puts("<html><head><title>Form Example</title></head>");
    puts("<body><h1>My Example Form</h1>");
    puts("<form action=\"form.cgi\" method=\"GET\">");
    puts("Name: <input type=\"text\" name=\"name\">");
    puts("Age: <input type=\"text\" name=\"age\">");
    puts("<br><input type=\"submit\">");
    puts("</form>");

    data = getenv("QUERY_STRING");
    if(data && *data) {
        params = data; dest = data;
        while(*data) {
            if(*data=='+') *dest=' ';
            else if(*data == '%' && ishex(*(data+1))&&ishex(*(data+2))) {
                *dest = (char) htoi(data + 1);
                data+=2;
            } else *dest = *data;
            data++;
            dest++;
        }
        *dest = '\0';
        s = strtok(params,"&");
        do {
            tmp = strchr(s,'=');
            if(tmp) {
                *tmp = '\0';
                if(!strcmp(s,"name")) name = tmp+1;
                else if(!strcmp(s,"age")) age = tmp+1;
            }
        } while(s=strtok(NULL,"&"));

        printf("Hi %s, you are %s years old\n",name,age);
    }
    puts("</body></html>");
}
```

The Perl alternative

Perl became an obvious choice because it was made for text processing. The same thing in Perl using CGI.pm:

```
use CGI qw(:standard);
print header;
print start_html('Form Example'),
    h1('My Example Form'),
    start_form,
    "Name: ", textfield('name'),
    p,
    "Age: ", textfield('age'),
    p,
    submit,
    end_form;
if(param()) {
    print "Hi ",em(param('name')),
        "You are ",em(param('age')),
        " years old";
}
print end_html;
```

Much easier both to read and to write, at least to people with a bit of a programming background.

The PHP Approach

PHP has an HTML-centric approach. The same script in PHP became:

```
<html><head><title>Form Example</title></head>
<body><h1>My Example Form</h1>
<form action="form.phtml" method="POST">
Name: <input type="text" name="name">
Age: <input type="text" name="age">
<br><input type="submit">
</form>
<?if($name):?>
Hi <?echo $name?>, you are <?echo $age?> years old
<?endif?>
</body></html>
```

A block of raw HTML followed by the minimum amount of logic possible.

History

1994

- Initial inklings of PHP in the form of a bunch of small CGI programs written in C.

1995

- Took a job with the University of Toronto to build a system called UTorDial.
- June 8 - First official 1.0 release called PHP Tools
- September 18 - Version 1.90 included the first real YACC-based parser
- October 9 - Version 1.91 added FI tool

1996

- March 16 - PHP/FI released
- April 23 - Apache 1.0/1.1 module support
- September 22 - Apache 1.2 DSO support

1997

- October 29 - First 3.0 alpha release

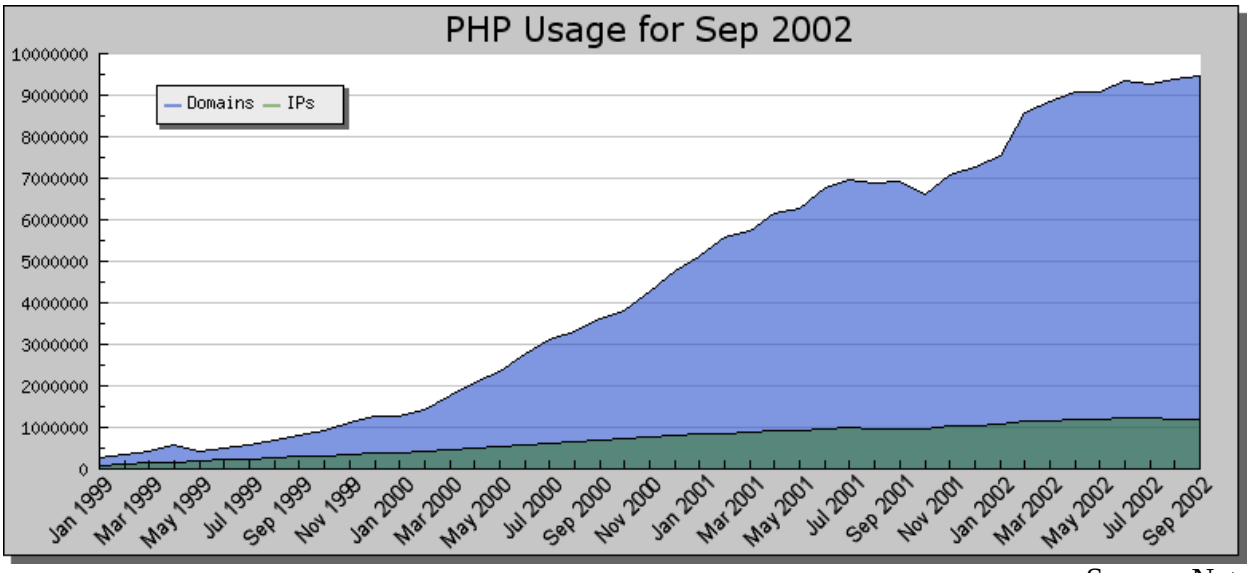
1998

- June 6 - First PHP 3.0 release

PHP Usage Growth

September 2002 Netcraft Report

- 35,756,436 Domains queried
- 9,458,364 Domains. 1,191,872 IP addresses
- PHP installed on 26% of all domains



Source: Netcraft

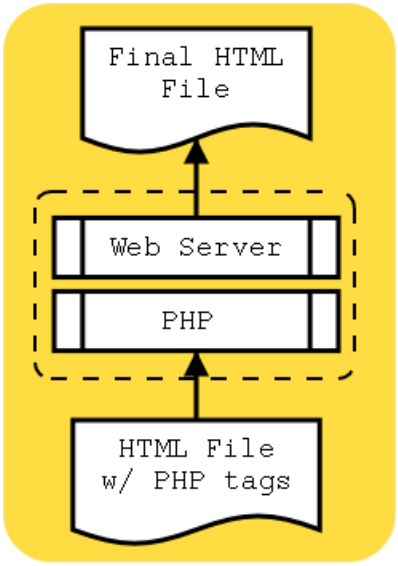
September 2002 Apache Module Report

- 5,758,921 Apache Servers surveyed
- 2,381,307 (39.44%) PHP
- 2,198,834 (36.42%) mod_perl
- 1,459,613 (24.18%) OpenSSL
- 1,406,702 (23.30%) mod_ssl
- 1,096,914 (18.17%) Frontpage
- 248,374 (4.11%) DAV
- 201,401 (3.34%) mod_throttle
- 155,147 (2.57%) AuthMySQL
- 141,285 (2.34%) ApacheJServ

Source: SecuritySpace.com

PHP is a Server-side language

Even though it is embedded in HTML files much like the client-side Javascript language, PHP is server-side and all PHP tags will be replaced by the server before anything is sent to the web browser.



So if the HTML file contains:

```
<html>
<?php echo "Hello World"?>
</html>
```

What the end user would see with a "view source" in the browser would be:

```
<html>
Hello World
</html>
```

Embedding PHP Tags

The 4 available tag styles

```
<html>
<body>
<? echo 'Short Tags - Most common' ?>
<br />
<?php echo 'Long Tags - Portable' ?>
<br />
<%= 'ASP Tags' %>
<br />
<script language="php">
 echo 'Really Long Tags - rarely used';
</script>
<br />
</body>
</html>
```

Output:

```
Short Tags - Most common
Long Tags - Portable


Really Long Tags - rarely used
```

Language Basics

Variables and Expressions

```
<?php
    $foo = 1;
    $bar = "Testing";
    $xyz = 3.14;
    $foo = $foo + 1;
?>
```

Arrays

```
<?php
    $foo[1] = 1;
    $foo[2] = 2;
    $bar[1][2] = 3;
?>
```

Functions

```
<?php
    phpinfo();
    foo();
    $len = strlen($foo);
?>
```

Control Structures

```
<?php
    while($foo) {
        ...
    }
?>
```

Output

```
<?php
    echo $foo;
    printf(".2f",$price);
?>
```

Switching Modes

Syntax and switching modes

```
<? if(strstr($_SERVER['HTTP_USER_AGENT'],"MSIE")) { ?>
<b>You are using Internet Explorer</b>
<? } else { ?>
<b>You are not using Internet Explorer</b>
<? } ?>
```

Output:

```
You are not using Internet Explorer
```

Form Handling

Traditional Form Handling

```
<form action="<?=$PHP_SELF?>" method="POST">
Your name: <input type=text name=name><br>
You age: <input type=text name=age><br>
<input type=submit>
</form>
```

Output:

```
Your name:
You age:
```

Receiving Script

```
Hi <?echo $name?>.
You are <?echo $age?> years old.
```

Register Globals

Some feel that automatically populating the symbol table with user-supplied data can lead to insecure programs, which to some extent is correct. To combat this the register_globals setting is off by default in PHP 4.2 and later.

```
Hi <?echo $_POST['name'] ?>.
You are <?echo $_POST['age'] ?> years old.
```

Basic Data Types

Numbers (integers and real)

```
<?php
    $a = 1234;
    $b = 0777;
    $c = 0xff;
    $d = 1.25;
    echo "$a $b $c $d<br />\n";
?>
```

Output:

```
1234 511 255 1.25
```

Strings

```
<?php
    $name = 'Rasmus $last'; // Single-quoted
    $str  = "Hi $name\n";   // Double-quoted
    echo $str;
?>
```

Output:

```
Hi Rasmus $last
```

Booleans

```
<?php
    $greeting = true;
    if($greeting) {
        echo "Hi Carl";
        $greeting = false;
    }
?>
```

Output:

```
Hi Carl
```

Dynamic Typing

- Don't have to declare types
- Automatic conversion done

```
<?php
    echo 5 + "1.5" + "10e2";
?>
```

Output:

```
1006.5
```

User Defined Functions

Typical User Defined Function

```
<?php
function log_data($user, &$data) {
 mysql_query("INSERT INTO userdata VALUES ('".
            uniqid()."', '$user', '$data')");
}
?>
```

Pass-by-reference

```
<?php log_data($PHP_AUTH_USER, $data); ?>
```

Default values

```
<?php
function head($title="Default Title") {?>
    <HTML><HEAD><TITLE>
    <? echo $title ?>
    </TITLE></HEAD><BODY><?
}

head();
?>
```

Class Definition

```
<?php
  class Cart {
    var $items;
    function add_item($artnr, $num) {
        $this->items[$artnr] += $num;
    }
  }
?>
```

Inheriting a class with a Constructor

```
<?php
  class NamedCart extends Cart {
    var $owner;
    function NamedCart($name) {
        $this->owner = $name;
    }
  }
?>
```

Invocation

```
<?php
  $cart = new NamedCart("PenguinGear");
  $cart->add_item(170923, 2);
?>
```

Static Method calls

```
<?php
  class foo {
    function bar() {
        echo "bar() called";
    }
  }

  foo::bar();
?>
```

Output:

```
bar() called
```

Calling methods in your parent class

```
<?php
  class foo2 {
    function foo2() {
        echo "Constructor for foo2";
    }
  }

  class bar extends foo2 {
    function bar() {
        echo "Constructor for bar<br />";
        $name = get_parent_class($this);
        parent::$name();
```

```
    }
  }

  $a = new bar();
?>
```

Output:

```
Constructor for barConstructor for foo2
```

Date/Time Functions

Simple Date Display

```
<?php echo date("M d, Y H:i:s", time()); ?>
```

Output:

```
Nov 02, 2002 14:47:46
```

How old is CArL?

```
<?php
  $birth = mktime(13,26,0,3,6,2002);
  $weeks = (int)((time() - $birth)/(7*86400));
  $days = (int)((time() - $birth)/86400) - $weeks*7;
  $hours = (int)((time() - $birth)/3600) - $days*24 - $weeks*7*24;
  $mins = (int)((time()-$birth)/60) - $hours*60 - $days*24*60 - $weeks*7*24*60;
  if($weeks>0) $str = "$weeks weeks, ";
  if($days>0) $str .= "$days day";
  if($days>1) $str .= "s";
  if($hours == 1) $str .= " $hours hour and";
  else $str .= " $hours hours and";
  if($mins == 1) $str .= " 1 minute";
  else $str .= " $mins minutes";
  echo "CArL is now $str old";
?>
```

Output:

```
CArL is now 34 weeks, 3 days 1 hour and 21 minutes old
```

Calendar Conversions

Converts between calendars (Julian, Gregorian, Mayan, etc)

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)?>">
```

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:
```

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;
?>
```

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:

$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.

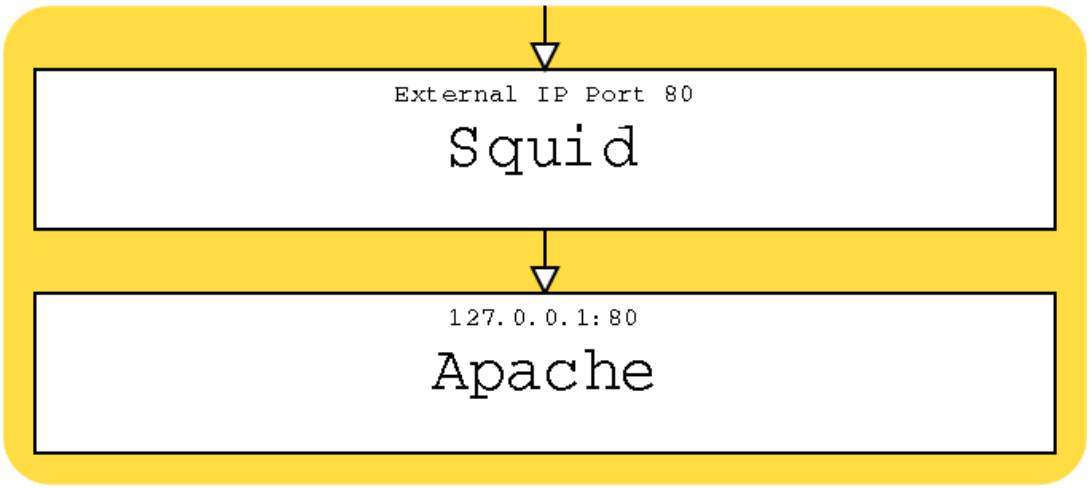
```
ErrorDocument 404 /generate.php
```

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:

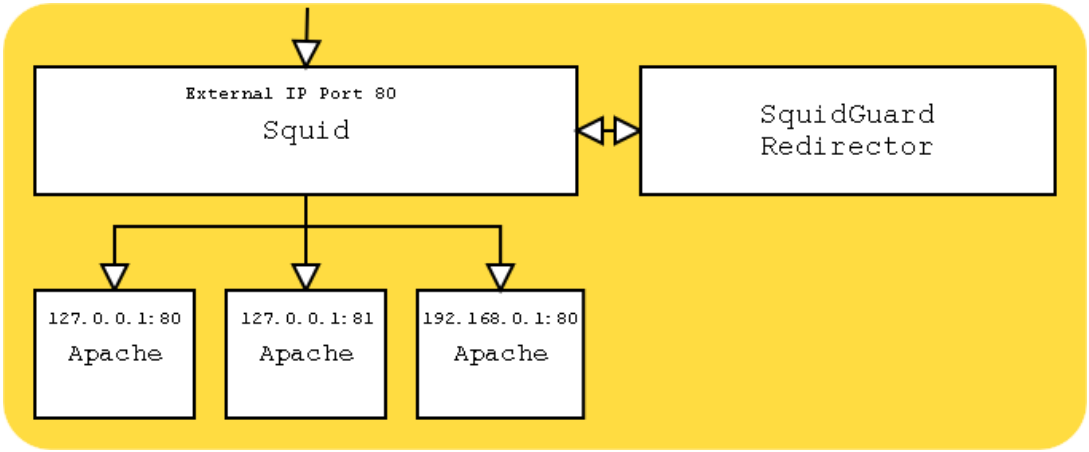
```
<?php
    $s = $REDIRECT_URI;
    $d = $DOCUMENT_ROOT;
    // determine requested uri
    $uri = substr($s, strpos($s,$d) + strlen($d) + 1);
    ob_start();  // Start buffering output
    // ... code to fetch and output content from DB ...
    $data = ob_get_contents();
    $fp = fopen("$DOCUMENT_ROOT/$uri",'w');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

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
```

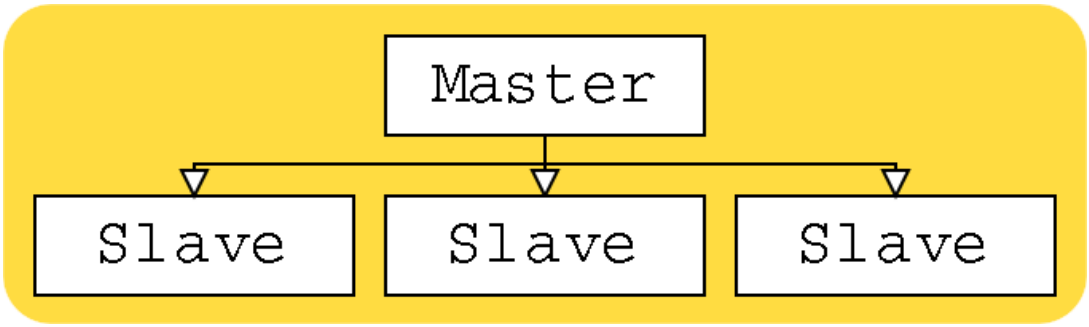
Single-server reverse proxy setup (set up Apache to listen to port 80 on the loopback):

```
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';
```

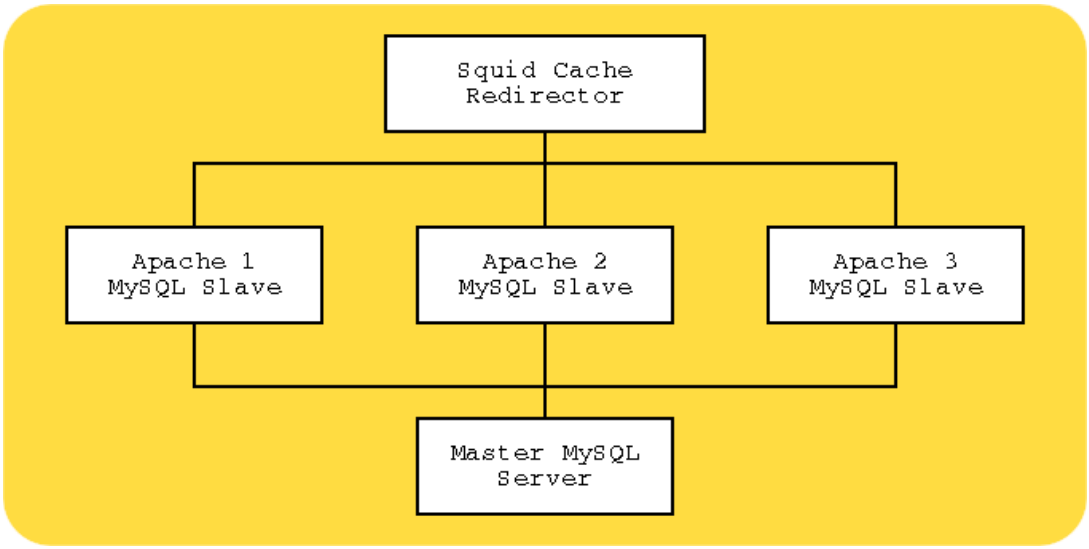
If you are using MySQL 4.0.2 or later, replace FILE with REPLICATION SLAVE in the above. Then on your slave servers:

```
[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.

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