Get Soaked

An in-depth look at streams

Davey Shafik

Developer at EngineYard/Orchestra.io

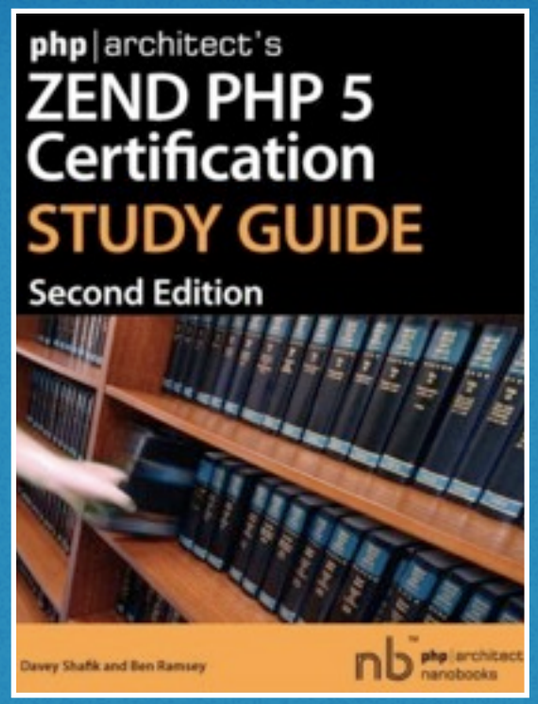
Author of Zend PHP 5 Certification Study Guide & Sitepoints PHP Anthology: 101 Essential Tips, Tricks & Hacks

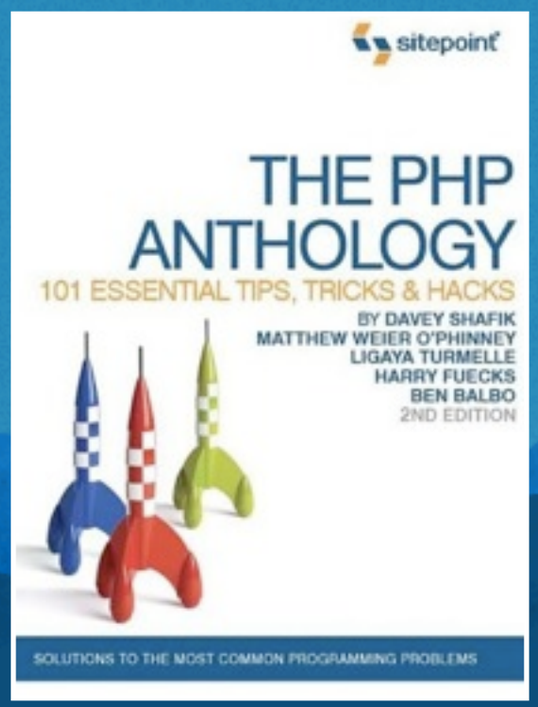
A long time contributor to PEAR, phpdoc; new contributor to internals, FCKEditor

Original Contributor to Zend Framework

Hard of hearing. Speak up!

(Buy my books!)





ORCHESTRA

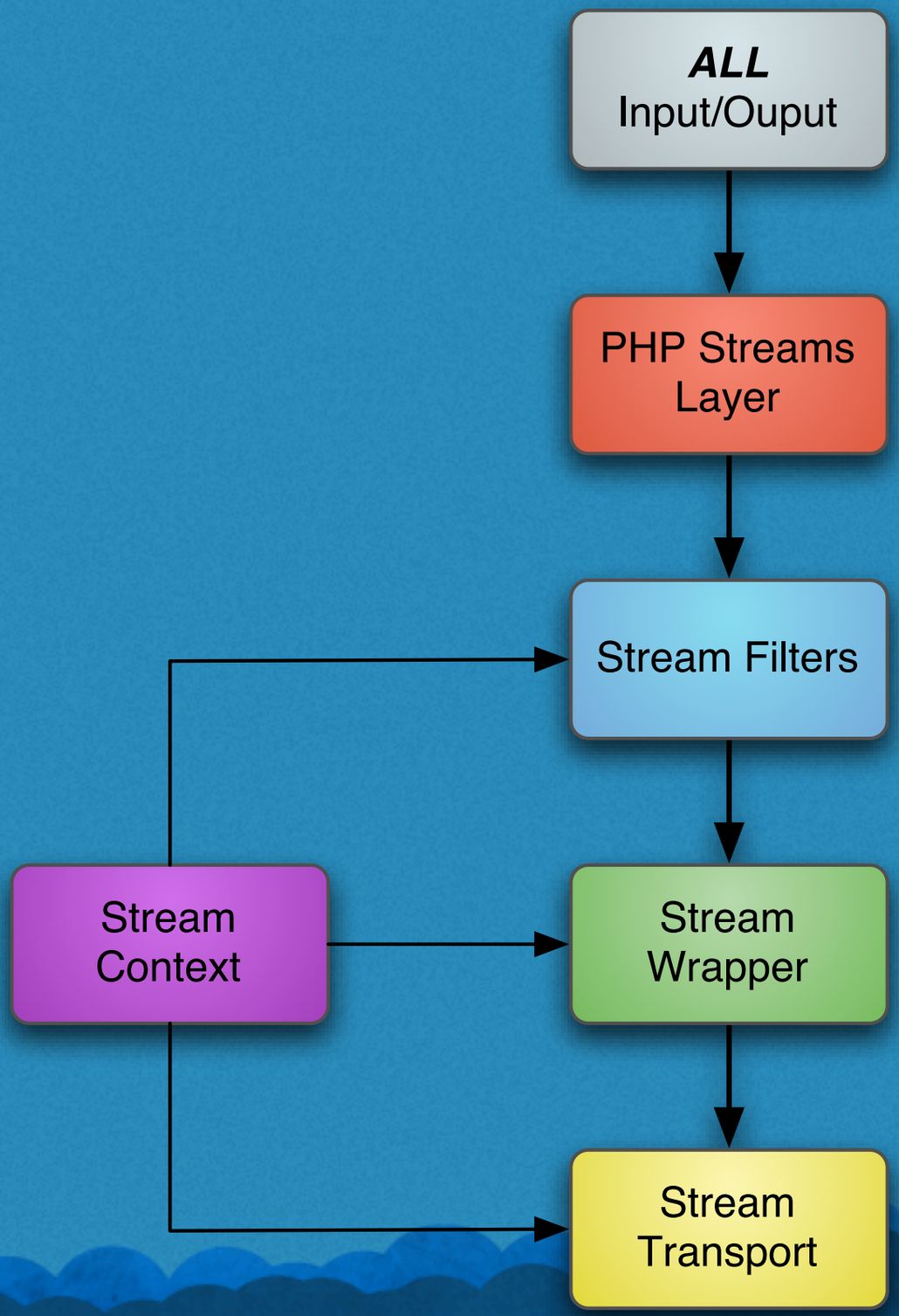
DEPLOY, MANAGE, SCALE
YOUR PHP APPLICATIONS.

Stream Basics

in

5 minutes

PHP Streams Layer



Input/Output

Input/Output

- include/require, and _once variants
- fopen, fread, fgets, fgetcsv, fputcsv, fwrite, fclose, feof, etc
- file_get_contents, file_put_contents, readfile, file
- mkdir, rename, unlink

Stream Filters [SUCK!]

Stream filters [suck!]

- Modify Data on-the-fly as it passes through the stream
- Appended/Removed on-the-fly
  - To the top and bottom of the stack
- Can be chained
- Defaults:
  - string.* (useless)

Stream Wrappers

Stream Wrappers

- Several Built-in by default
  - file:, http:, ftp:, data: (5.2+), glob: (5.3+)
  - With openssl extension: https:, ftps:
  - With zlib extension: zlib:
  - Special php:// stream

Stream Transports

Stream Transports

- TCP
- UTP
- UNIX/UDG
- + SSL, SSLv2, SSLv3 or TLS
  - HTTPS is TCP+SSL (1, 2, or 3)

Stream Contexts

Stream contexts

- Provides Context (duh) to stream wrappers
  - Configuration (options)
    - Request Type (HEAD, POST, GET)
    - Login Credentials

Using Streams

```php
<?php
// Define our context options
$options = array(
    // We use a nested array like:
    // $options['stream_wrapper']['option'] = 'value';
    'http' => array(
        'method' => 'HEAD'
    )
);

// Create the stream context
$context = stream_context_create($options);

// Pass it into readfile()
readfile("http://daveyshafik.com", false, $context);

// Check out the headers
var_dump($http_response_header);
?>
```

```
array(12) {
  [0]=>
  string(15) "HTTP/1.1 200 OK"
  [1]=>
  string(35) "Date: Sun, 20 Apr 2008 20:50:35 GMT"
  [2]=>
  string(55) "Server: Apache/2.2.6 (Debian) DAV/2 SVN/1.4.2 PHP/5.2.5"
  [3]=>
  string(23) "X-Powered-By: PHP/5.2.5"
  [4]=>
  string(62) "Set-Cookie: PHPSESSID=11581e8237532l3e5c8f36730f034; path=/"
  [5]=>
  string(10) "Expires: 0"
  [6]=>
  string(50) "Cache-Control: no-cache, pre-check=0, post-check=0"
  [7]=>
  string(16) "Pragma: no-cache"
  [8]=>
  string(22) "X-Session-Reinit: true"
  [9]=>
  string(19) "X-Blog: Serendipity"
  [10]=>
  string(17) "Connection: close"
  [11]=>
  string(43) "Content-Type: text/html; charset=ISO-8859-1"
}
```

```php
<?php
$options = array(
    'http' => array(
        'method' => 'HEAD'
    )
);

$context = stream_context_create($options);

// This time we use fopen()
$stream = fopen(
            "http://daveyshafik.com", "r",
            null, $context);

// and stream_get_meta_data()
$meta = stream_get_meta_data($stream);

var_dump($meta);
?>
```

```
array(10) {
  ["wrapper_data"]=>
  array(12) {
    // The same as before
  }
  ["wrapper_type"]=>
  string(4) "http"
  ["stream_type"]=>
  string(10) "tcp_socket"
  ["mode"]=>
  string(2) "r+"
  ["unread_bytes"]=>
  int(0)
  ["seekable"]=>
  bool(false)
  ["uri"]=>
  string(27) "http://pixelated-dreams.com"
  ["timed_out"]=>
  bool(false)
  ["blocked"]=>
  bool(true)
  ["eof"]=>
  bool(true)
}
```

```php
<?php
// The data to send
$data = array('username' => 'davey', 'password' => 'example');

// Our context options array
$options = array (
   'http' => array (
      'method' => 'POST',
      'header'=>
      "Content-type: application/x-www-form-urlencoded\r\n"
      . "Content-Length: " . strlen($urlencoded) . "\r\n",
      'content' => http_build_query($data)
   )
);

$context = stream_context_create($options);

$url = $_SERVER['HTTP_HOST'] . $_SERVER['SCRIPT_NAME'];

echo file_get_contents("http://example.org", false, $context);
?>
```

```php
<?php
$user = "php";
$pass = "streams";
$host = "localhost";
$file = "example/lipsum.txt";
$scheme = (extension_loaded('openssl')) ? 'ftps' : 'ftp';

$data = 'Donec condimentum leo id mi. Ut luctus mi pellentesque lor
em.';

$r = file_put_contents(
                    "$scheme://$user:$pass@$host/$file",
                     $data,
                    FILE_APPEND
                );

if ($r === false) {
    echo "An error occured!";
}
?>
```

```php
<?php
$user = "php";
$pass = "streams";
$host = "localhost";
$file = "example/lipsum.txt";
$scheme = (extension_loaded('openssl')) ? 'ftps' : 'ftp';

$file = "$scheme://$user:$pass@$host/$file";

if (is_readable($file)) {
    echo "We can read the file, ";
    if (is_writable($file)) {
        echo "and we can write to it too!";
    } else {
        echo "but we can't write to it!";
    }
} else {
    echo "We canna read the file cap'n!";
}
?>
```

[not so] practical example — Twitter Client

Wouldn’t it be nice...

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  </head>
  <body>
    <h1>Tweets</h1>
    <h2>Public Tweets</h2>
    <ul>
      <?php
      foreach (file("tweet://public") as $tweet) {
          echo "<li>" .$tweet. "</li>";
      }
      ?>
    </ul>
    <h2>Search for #works</h2>
    <ul>
      <?php
          foreach (file("tweet://search/#works") as $tweet) {
              echo "<li>" .$tweet. "</li>";
          }
      ?>
</ul>
<h2>Search for PHP and Python</h2>
<ul>
      <?php
          foreach (file("tweet://search/php/python") as $tweet) {
              echo "<li>" .$tweet. "</li>";
          }
      ?>
</ul>
```

- Output: http://daveyshafik.com/~davey/twitter.html
- Source: http://daveyshafik.com/~davey/twitter.phps

Something More Practical? OK!

Interfaces

```php
<?php
interface Stream_Interface {
    public function stream_open($path,$mode,$options,&
$opened_path);

    public function stream_close();

    public function stream_read($count);

    public function stream_write($data);

    public function stream_eof();

    public function stream_tell();

    public function stream_seek($offset, $whence);

    public function stream_stat();

    public function url_stat($path, $flags);
}
?>
```

```php
<?php
interface Stream_Interface_Directory {
    public function dir_opendir($path, $options);

    public function dir_readdir();

    public function dir_rewinddir();

    public function dir_closedir();

    public function mkdir($path, $mode, $options);

    public function rmdir($path, $options);
}
?>
```

```php
<?php
interface Stream_Interface_RenameUnlink {
    public function rename($path_from, $path_to);

    public function unlink($path);
}
?>
```

```php
<?php
interface Stream_Interface_Flush {
    public function stream_flush();
}
?>
```

Caching

```php
<?php
// Set the Context
// The 'cache' directory in the current dir
$dsn = 'file://' .realpath('./') .DIRECTORY_SEPARATOR. 'cache';
$options = array(
    'cache' => array(
        'dsn' => $dsn, // the cache directory
        'ttl' => 86400, // One day
    )
);

// Use stream_context_GET_default() in PHP versions lower than 5.3
stream_context_set_default($options);

if (file_exists('cache://blogroll.htm')) {
    readfile('cache://blogroll.htm');
} else {
    $pdo = new PDO($dsn);
    $sql = "SELECT * FROM blogroll WHERE blog_id=:id";
    $query = $pdo->prepare($sql);
    $query->execute(array(':id' => $CONFIG['blog_id']));
    $html = '';
    foreach ($query->fetchObj() as $url) {
        $html .= "<a href='$url->link'>$url->name</a>";
    }
    file_put_contents('cache://blogroll.htm', $html);
    echo $html;
}
?>
```

```php
public function stream_stat()
{
    // Run the fstat on the actual file
    $stat = fstat($this->fp);

    // Determine if the file has expired
    $ttl = $this->context_options['ttl'];
    if ($stat && $stat['mtime'] < time() - $ttl) {
        // Pretend the file doesn't exist
        $unlink = unlink($this->file);
        if ($unlink) {
            $msg = __METHOD__ . "() Unable to remove";
            $msg .= "stale cache ($this->file)"
            trigger_error($msg, E_USER_NOTICE);
        }
        return false;
    } else {
        // Return the real stat
        return $stat;
    }
}
```

```php
public function url_stat($path, $flags)
{
    // If there is no context, grab the default
    if (is_null($this->context)) {
        $this->context = stream_context_get_default();
    }

    // Retrieve just our context
     $options = stream_context_get_options($this->context);
    $this->context_options = $options['cache'];

    $file = $this->context_options['dsn'];
    $file .= DIRECTORY_SEPARATOR;
    $file .= str_replace("cache://", "", $path);
    $fp = @fopen($file, "r");

    if (!$fp) {
        return false;
    }

    $stat = fstat($fp);

    // If the file's modification time is less than
    // the current time minus the time to live, the
    // file has expired.
    $ttl = $this->context_options['ttl'];
    if ($stat['mtime'] < (time() - $ttl)) {
        // Remove the file and pretend the file doesn't exist
        unlink($file);
        return false;
    }

    return $stat;
}
```

What's the point?

```php
<?php
// Set the Context
// The 'cache' directory in the current dir
$dsn = 'mysql://username:password@localhost/cache';
$options = array(
    'cache' => array(
        'dsn' => $dsn, // the cache directory
        'ttl' => 86400, // One day
    )
);

// Use stream_context_GET_default() in PHP versions lower than 5.3
stream_context_set_default($options);

if (file_exists('cache://blogroll.htm')) {
    readfile('cache://blogroll.htm');
} else {
    $pdo = new PDO($dsn);
    $sql = "SELECT * FROM blogroll WHERE blog_id=:id";
    $query = $pdo->prepare($sql);
    $query->execute(array(':id' => $CONFIG['blog_id']));
    $html = '';
    foreach ($query->fetchObj() as $url) {
        $html .= "<a href='$url->link'>$url->name</a>";
    }
    file_put_contents('cache://blogroll.htm', $html);
    echo $html;
}
?>
```

PHP 5.3 Changes Stuff

Streams can be put in the include path:
set_include_path(“.:/lib:phar://my.phar”);
PHAR is include by default [woohoo!]
stream_context_SET_default() [my first patch :)]

PHAR

Phar archives are single-file applications that work without extraction:

php blah.phar
include 'blah.phar';
include 'phar://blah.phar/internal/file.php';
Phar archives are tar, zip, or custom phar file format
Native PHAR format is the fastest.
Stable: Used for PEAR for a long time!

Comments?

E-mail: dshafik@engineyard.com
Blog: http://daveyshafik.com
Twitter: http://twitter.com/dshafik