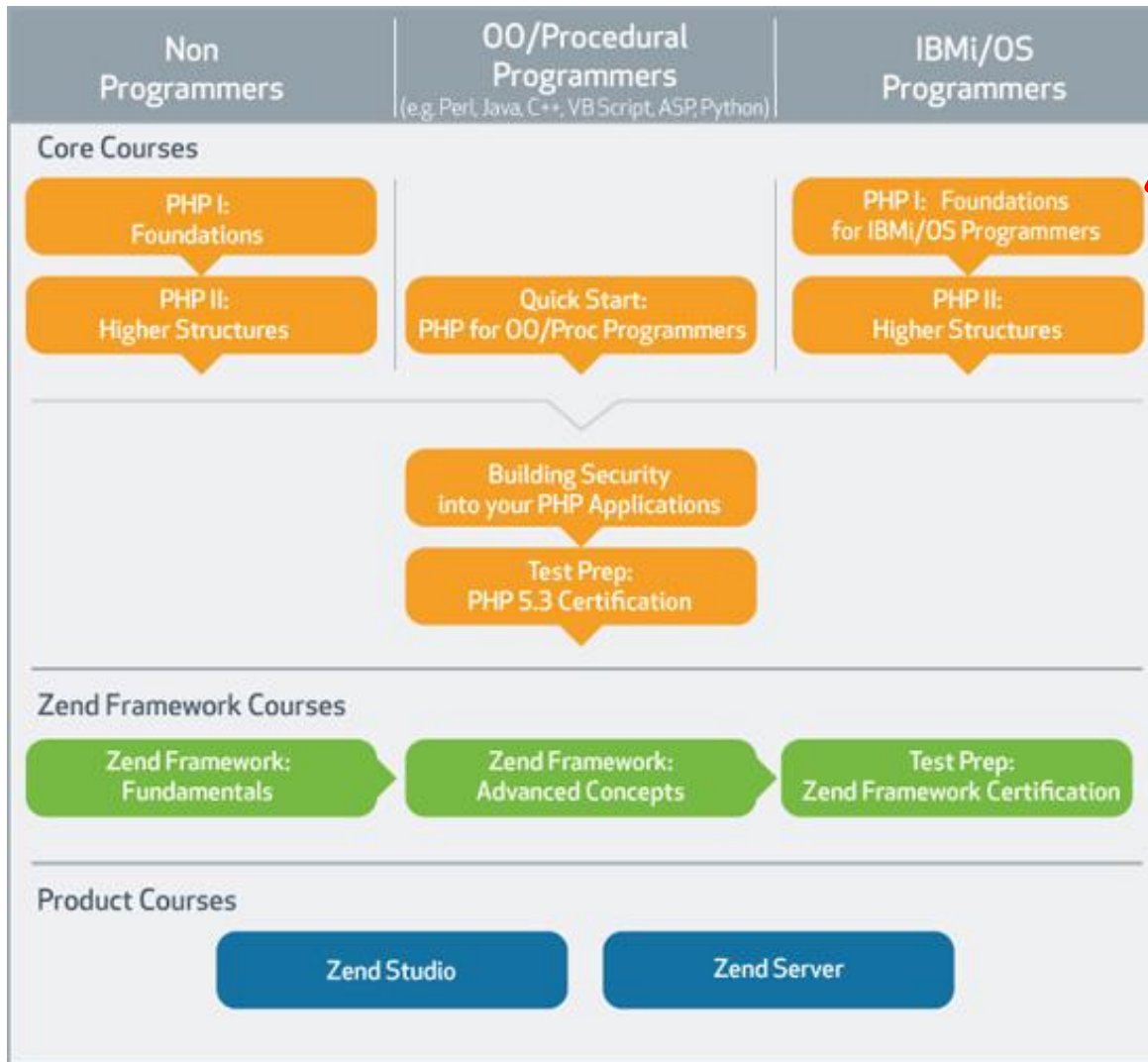




The PHP Company

# Introduction to Object Oriented PHP

Mike Pavlak  
Solutions Consultant  
[mike.p@zend.com](mailto:mike.p@zend.com)  
(815) 722 3454



**Foundations course starting next week!**

Call your account manager or go to the Zend website:  
<http://shop.zend.com/en/php-training.html>





**Zend PHP Conference**  
October 17-20, 2011 – Santa Clara, CA

# Join us at ZendCon

## The premier PHP conference!

October 17-19, 2011 – Santa Clara, CA



### Conference Highlights

- Learn PHP best practices for architecture, design and development
- Technical sessions for all knowledge levels
- In-depth tutorials for advanced learning
- PHP Certification courses and testing
- Exhibit hall showcasing the latest products
- Networking opportunities with peers and luminaries

### Conference Topics

- Architecture & Design
- Expanding Horizons
- IBM i
- Lifecycle Best Practices
- NoSQL / Alternative Stores / Search
- PHP Development
- Server/Operations
- SQL
- Zend Framework

[www.zendcon.com](http://www.zendcon.com)

# Audience

---

- **New to PHP**
- **New to Object Oriented Design**
- **Open to new ideas and methods**
- **This is not a DEEP DIVE**
  - ▶ Relax and enjoy the ride!

# Agenda

---

- Quick review of PHP basics
- The class and its object
- Components of an object
  - ▶ Properties
  - ▶ Methods
- Other OO stuff
- I don't know OO: Get Started NOW!

# Roadmap to Expert PHP

---

- RPG Green Screen → PHP Expert
- Start somewhere! Here isn't bad!
- Path
  - Procedural PHP
  - OO PHP
  - Zend Framework (or another)
  - PHP Certification

# Questions?

---

- Let's keep it interactive!

- Follow us!

 ▶ <http://bit.ly/cjueZg> (Zend Technologies or search for Zend)

 ▶ <http://twitter.com/zend>

# Introduction to OO PHP

[www.zend.com](http://www.zend.com)

Quick review of PHP basics

# Variables

- **Rules**

- ▶ Case sensitive
- ▶ Begin with \$
  - \$thisIsMyVariable
  - \$\_\_AnotherVariable
  - \$ this is not a variable
- ▶ Implicit casting
- ▶ Can be re-typed (Dynamically Typed Language)

- **Constant - Variable that doesn't change**

- ▶ Define('TEACHER', "Mike Pavlak");

```
<?php
```

```
$field1 = 5;
```

```
$field2 = 10;
```

```
$field3 = $field1 + $field2;
```

```
?>
```

# Variables and their types

- **Scalar**
  - ▶ Integer
    - -2,147,483,648 thru 2,147,483,647
    - Supports decimal, octal and hex representation
  - ▶ Floating-Point
    - 1.7E-308 thru 1.7E+308
    - 15 digits of decimal precision
  - ▶ Strings
    - Big. Really big. Too big to discuss!
  - ▶ Boolean
    - False is 0, 0.0, false keyword, empty string, object w/no values, null. All others are true
- **Object, array, null and resource**

# Variables...(cont.)

- **Scope**
  - ▶ Global - Available everywhere but inside function (sort of)
  - ▶ Local - Available only in a function, destroyed at end
  - ▶ Static - Available only in a function, but remains
  
- **Arrays (three types)**
  - ▶ Enumerated
  - ▶ Associative
  - ▶ Multi-dimensional

# Functions

- Functions come from 1 of 3 places
  - Built-in
    - Part of base PHP
  - Extensions
    - Components of modules like image functions in GD
  - User defined
    - You will create these!
- You may have been using functions...
  - `db2_connect()`
  - `print_r`
  - `strtoupper`
  - Etc.

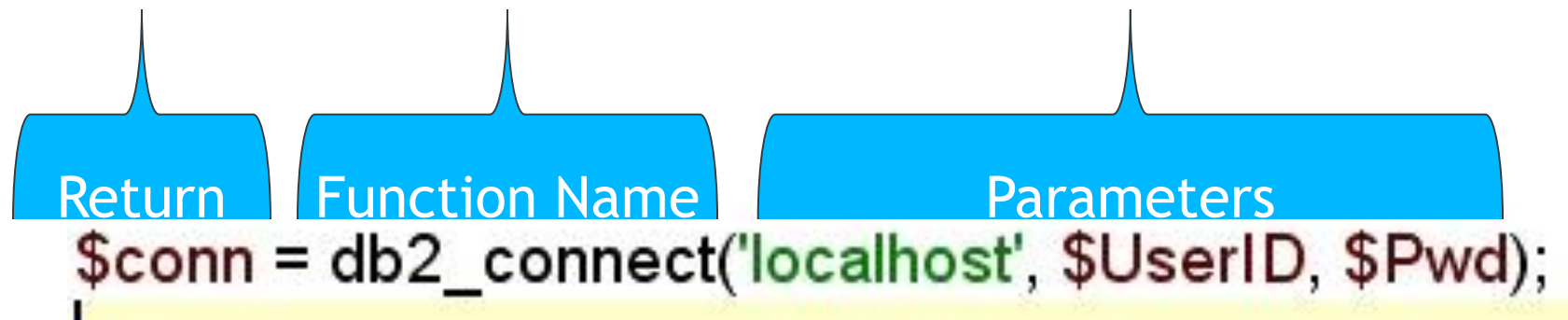
# What do functions look like?

## Function call has three major parts:

Function Name - Required

Parameters - Optional

Return Value - Always returned, default null



# Scope

- **Variables have scope, much like RPG ILE**
  - Function keeps its own set of variables
  - Global keyword can change all that (RPG III)
  - Better to pass, globals can't be trusted
- **Static variables**
  - Persistence from call to call

# Parameters

- **Default**
  - Assign value in interface
- **Pass by value**
  - This is the default behavior
- **Pass by reference**
  - This is done with & before the variable in the interface.
- **Variable number**
  - `func_get_args()`
  - `func_num_args()`
  - `func_get_arg(argument_#)`

# Introduction to OO PHP

[www.zend.com](http://www.zend.com)

## The class and its object

# Concepts

---

- **Keywords**
  - ▶ class
  - ▶ function
  - ▶ static
- **Concepts**
  - ▶ Class
  - ▶ Instance
  - ▶ Magic Methods
  - ▶ Visibility

# RELAX!!!

---

You may not be comfortable with the more modern structure

Don't feel like you **NEED** to follow it

Take your time learning it. PHP lets you do this

# Bare minimum

---

- Keep library files out of the document root
- Always name your PHP files .php
- Keep data out of the document root
- Do not deploy with a phpinfo.php file
  - ▶ Do a Google search for “inurl:phpinfo.php”
- Do not allow the web server to write to the document root
- Always validate your input.
  - ▶ Do a Google search for “inurl:page=home.php”

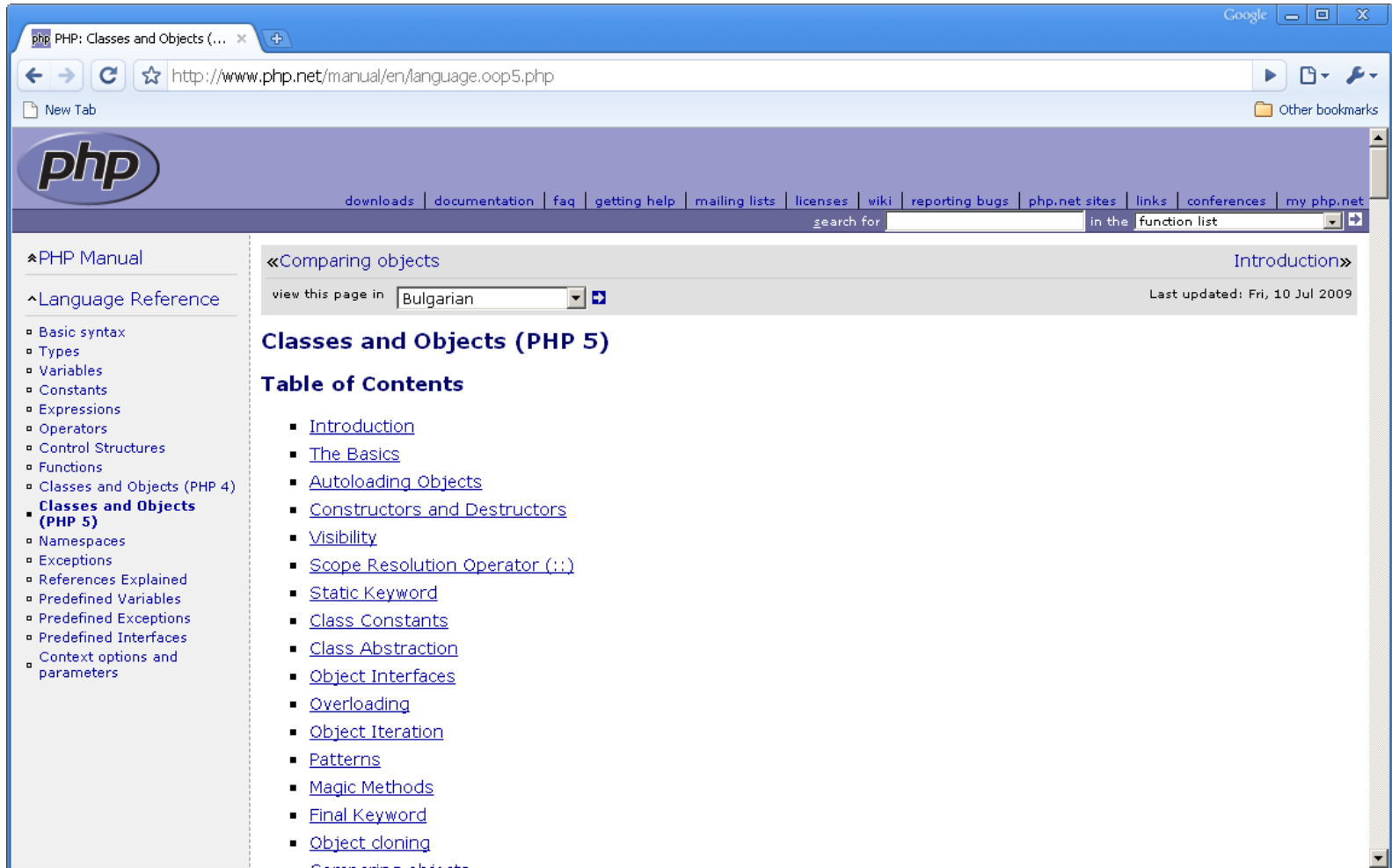
# Object Model

---

- **Object is centered around data, not logic**
  - Think about your RPG programs
  - You manipulate data.
  - Very few programs that have no data access
- **Object Definition:**
  - **Data structure** with **functions** and **variables** local to each object.
  - Keep routines similar to data element together

# Where to get more on OOP?

- <http://www.php.net/manual/en/language.oop5.php>



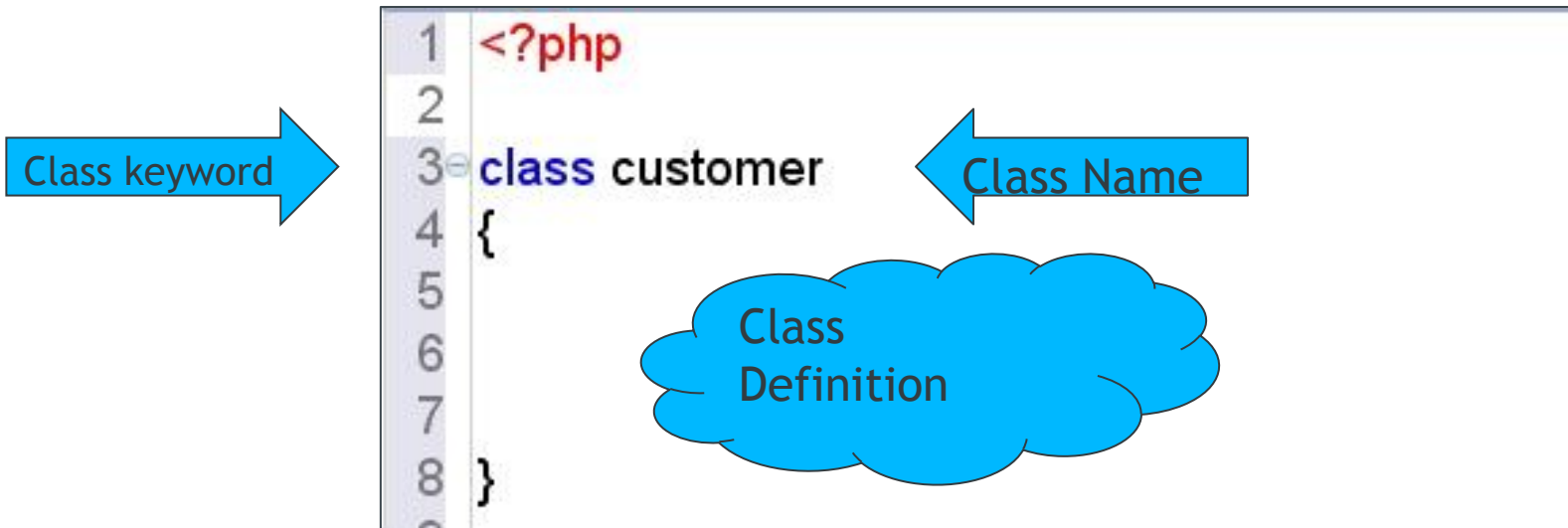
The screenshot shows a web browser window displaying the PHP manual page for 'Classes and Objects (PHP 5)'. The browser's address bar shows the URL <http://www.php.net/manual/en/language.oop5.php>. The page features the PHP logo and a navigation menu with links such as 'downloads', 'documentation', 'faq', 'getting help', 'mailing lists', 'licenses', 'wiki', 'reporting bugs', 'php.net sites', 'links', 'conferences', and 'my php.net'. A search bar is located below the navigation menu. The main content area is titled 'Classes and Objects (PHP 5)' and includes a 'Table of Contents' with the following items:

- [Introduction](#)
- [The Basics](#)
- [Autoloading Objects](#)
- [Constructors and Destructors](#)
- [Visibility](#)
- [Scope Resolution Operator \(::\)](#)
- [Static Keyword](#)
- [Class Constants](#)
- [Class Abstraction](#)
- [Object Interfaces](#)
- [Overloading](#)
- [Object Iteration](#)
- [Patterns](#)
- [Magic Methods](#)
- [Final Keyword](#)
- [Object cloning](#)

The page also includes a 'Table of Contents' section and a 'Last updated' date of Fri, 10 Jul 2009.

# Class

- Think of a class as Source Code for object
- Defines object via properties and methods



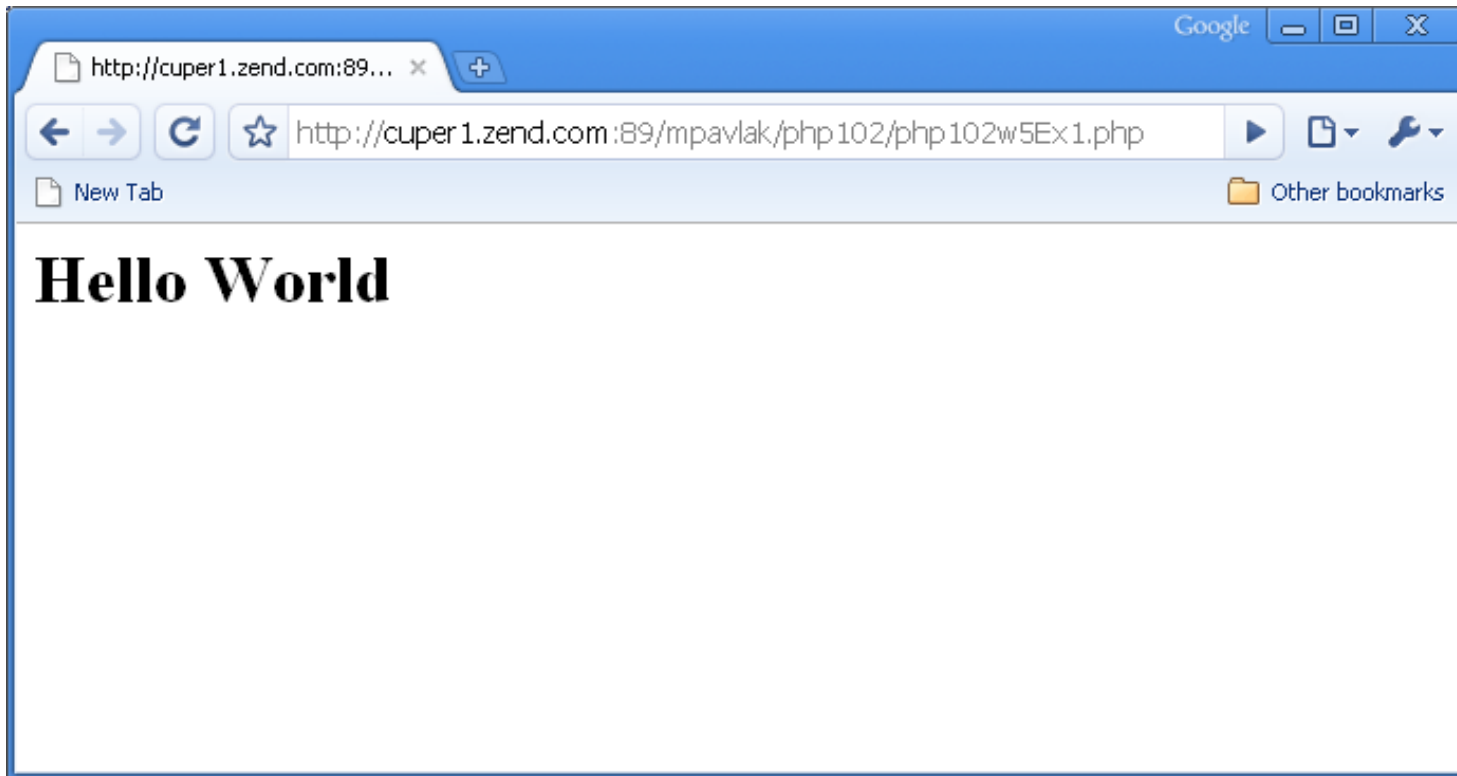
# Hello World Class

```
1 <?php
2 /**
3  * This class will display whatever is passed in to it...
4  *
5  */
6 class Hello {
7     public $message;
8     public function display () {
9         print_r($this->message);
10    }
11 }
12
13 $newMessage = new Hello();
14 $newMessage->message = "Hello World";
15
16 $newMessage->display();
17
18 ?>
```

# Hello Class

---

- Output looks strangely familiar!



# Introduction to OO PHP

[www.zend.com](http://www.zend.com)

## Components of an object

# Question

---

**Why would you not just use functions that are already in global scope?**

## **Problem**

- ▶ There is no auto-loading mechanism for procedural code
- ▶ This means that ALL functions need to be loaded for EACH request
- ▶ Structure is defined 100% by naming convention

## **Answer**

- ▶ Using classes to structure functionality means
  1. Only required functionality is compiled
  2. IFS access is minimized (this is good!)

# Class Properties

---

- Properties are also referred to as data
- Data representative of the class
- In our case, attributes about the customer
  - Name
  - Number
  - Address

```
3 class customer
4 {
5     private $name;
6     private $address;
7     private $city;
8     private $state;
9     private $zipCode
```

# Class Methods

---

- Methods are essentially functions in a class
- Provide database and some business logic
- In this example the method is display()

```
class Hello {  
    public $message;  
    public function display () {  
        print_r($this->message);  
    }  
}
```



# Notes on objects

---

- **Create Object:**
  - `$myObject = new Customer;`
  
- **Destroy Object**
  - `Unset $myObject;`

# Introduction to OO PHP

[www.zend.com](http://www.zend.com)

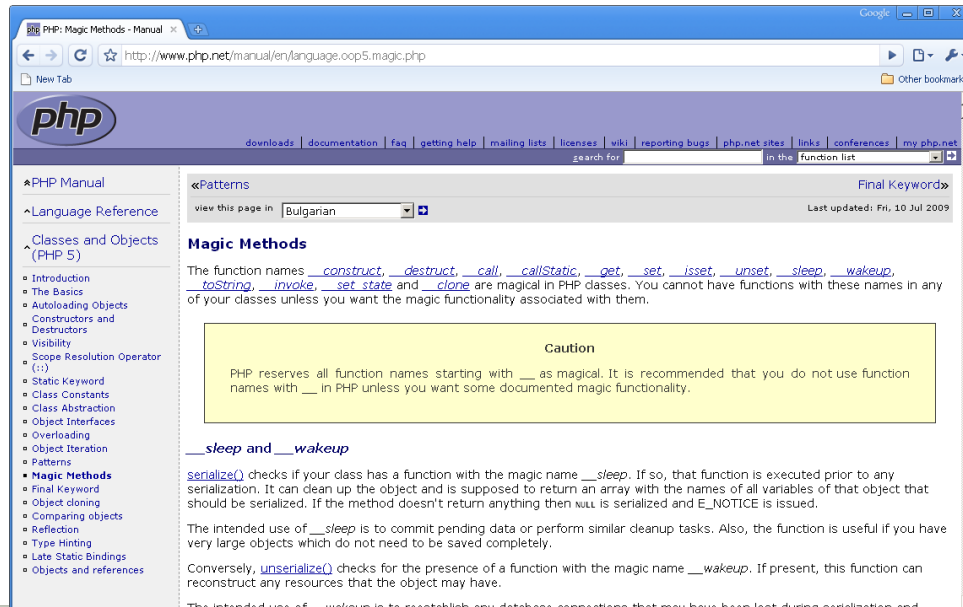
Other OO stuff

# Magic methods

- Methods are functions
- Magic methods are like special reserved functions in PHP
- Must begin with double underscore `__`

- More info?

<http://www.php.net/manual/en/language.oop5.magic.php>



The screenshot shows a web browser displaying the PHP Manual page for Magic Methods. The page title is "PHP: Magic Methods - Manual". The URL in the address bar is "http://www.php.net/manual/en/language.oop5.magic.php". The page content includes a navigation menu on the left, a search bar, and the main text of the article. The main text explains that magic methods are functions that begin with a double underscore and are reserved for PHP. It lists several magic methods: `__construct`, `__destruct`, `__call`, `__callStatic`, `__get`, `__set`, `__isset`, `__unset`, `__sleep`, `__wakeup`, `__toString`, `__invoke`, `__set_state`, and `__clone`. A yellow box with the heading "Caution" states: "PHP reserves all function names starting with `__` as magical. It is recommended that you do not use function names with `__` in PHP unless you want some documented magic functionality." Below this, the article discusses the `__sleep` and `__wakeup` methods. `__sleep` is used to commit pending data or perform similar cleanup tasks, and `__wakeup` is used to reestablish any database connections that may have been lost during serialization.

# Magic Methods

---

- Methods in objects that are called at special times
- They always start with double underscore (`__`)
  - ▶ `__construct()` \*
  - ▶ `__destruct()`
  - ▶ `__call()`
  - ▶ `__get()`
  - ▶ `__set()`
  - ▶ `__sleep()`
  - ▶ `__wakeup()`
  - ▶ `__toString()` \*

# Constructor

---

- `__construct()`
  - Think of it as \*INZSR
  - Executes code when object is instantiated

```
public function __construct() {  
    print_r("Object created<BR><BR>");  
}
```

# Destructor

---

- `__Destruct()`
  - Think of it as \*INLR
  - Executes code when object is destroyed

```
public function __destruct() {  
    print_r("<BR><BR>Object destroyed<BR><BR>");  
}  
}
```

# Hello Class with Magic Methods

```
class Hello {  
    public $message;  
    public function display () {  
        print_r($this->message);  
    }  
  
    public function __construct() {  
        print_r("<h2>Object created</h2>");  
    }  
  
    public function __destruct() {  
        print_r("<h2>Object destroyed</h2>");  
    }  
}  
  
$newMessage = new Hello();  
$newMessage->message = "<h1>Hello World</h1>";  
  
$newMessage->display();  
  
unset ($newMessage);
```



\*INZSR



\*INLR

# Visibility

---

- **Public**
  - Access to both inside and outside
- **Private**
  - Access to only inside
- **Protected**
  - Access only within the object itself, or other objects that extend the class.

# Visibility

- Allows you to limit what parts of a class are available to other parts of your application

	Application	Extending Class	Class Internals
Public	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Protected		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Private			<input checked="" type="checkbox"/>

# Visibility

```
class Hello {
    public $message;
    private $privateMessage="this is the private message";

    public function display () {
        print_r($this->message);
        echo $this->privateMessage;
    }

    public function __construct() {}

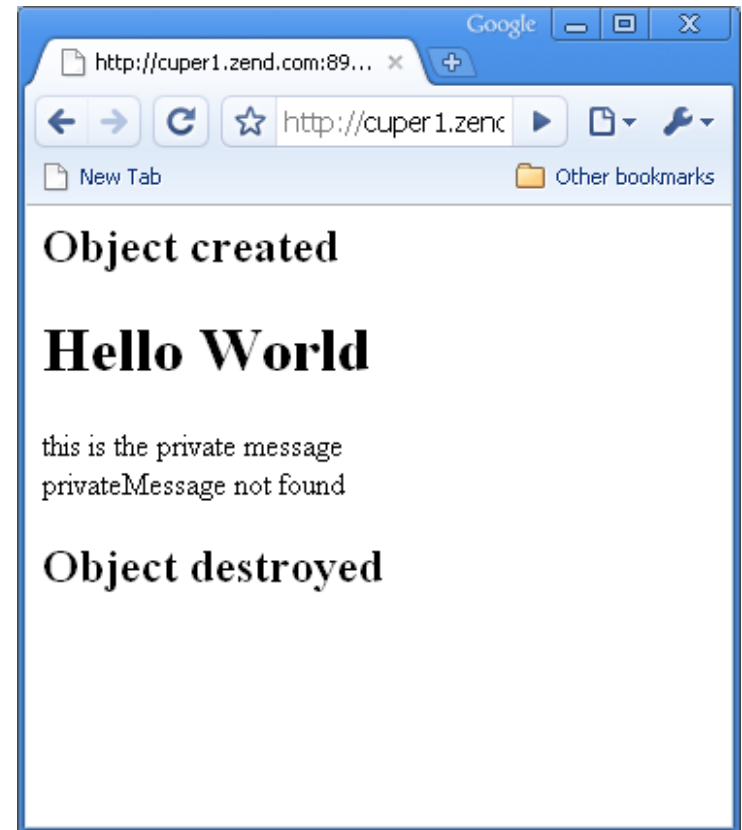
    public function __destruct() {}
}

$newMessage = new Hello();
$newMessage->message = "<h1>Hello World</h1>";

$newMessage->display();

if (isset($newMessage->privateMessage)) {
    print_r( "$newMessage->privateMessage <br>");
}
else { echo "<br>privateMessage not found"; }

unset ($newMessage);
```



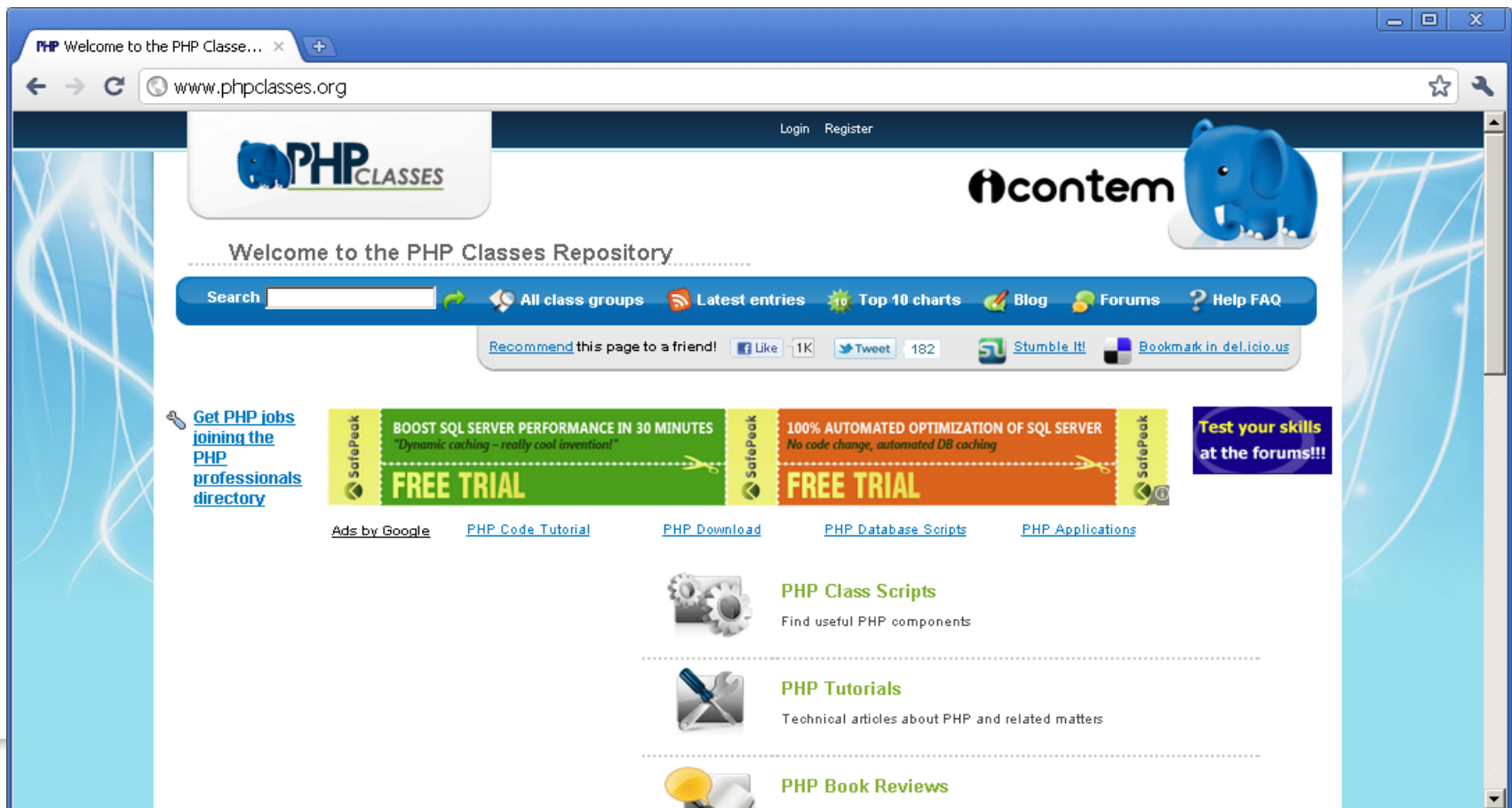
# Introduction to OO PHP

[www.zend.com](http://www.zend.com)

I don't know OO: Get Started NOW!!!

# You don't need to know OO to use it

- Using objects does not require OO knowledge
- Easy to weave into Procedural PHP applications
- <http://www.phpclasses.org/>



The screenshot shows a web browser window displaying the PHP Classes Repository website. The browser's address bar shows the URL [www.phpclasses.org/](http://www.phpclasses.org/). The website features a blue header with the PHP Classes logo and a navigation menu. Below the header, there is a search bar and a navigation bar with links for 'All class groups', 'Latest entries', 'Top 10 charts', 'Blog', 'Forums', and 'Help FAQ'. A social media bar includes links for 'Like', 'Tweet', 'Stumble It!', and 'Bookmark in del.icio.us'. The main content area contains several advertisements, including 'BOOST SQL SERVER PERFORMANCE IN 30 MINUTES' and '100% AUTOMATED OPTIMIZATION OF SQL SERVER', both offering 'FREE TRIAL'. A sidebar on the left promotes 'Get PHP jobs joining the PHP professionals directory'. The footer contains links for 'Ads by Google', 'PHP Code Tutorial', 'PHP Download', 'PHP Database Scripts', and 'PHP Applications'. The website is designed with a blue and white color scheme and features a blue elephant logo.

# email address validation

**Search this site**

Search for:

Sections:  Packages  Reviews  Blogs  Videos  Discussions

Search forums:  No  In selected sections  Only in forums

[Permalink](#)

Found: 116

- [Packages \(52\)](#)
- [Reviews \(3\)](#)
- [Blogs \(11\)](#)
- [Package forums \(46\)](#)
- [Blog forums \(4\)](#)

1-10 | [11-20](#) | [21-30](#) | [31-40](#) | [41-50](#) | [Next >](#)

## [View All Online Classes](#)

University of Phoenix®. Learn About Accredited Programs. Search Now!  
Phoenix.edu

Ads by Google

1. [Validations](#) ★★★★★ 100%

**Validations validations Validate** text values according several rules. This is a simple class implements several rules to **validate** text values on the server side, eventually submitted by the user in Web forms. Currently it implements the following **validation** rules: - **E-mail address** - Signed integer ...

2. [Generate Javascript Validation](#) ★★★★★ 70%

Generate Javascript **Validation** generatejsvalidation Generate Javascript code for **validation** Web forms. This class can be used to generate Javascript code for **validating** Web forms. It takes as parameter a list of form fields to be **validated** that details the field names, types of **validations** to be ...

3. [Combo Validation](#) ★★★ 53%

Combo **Validation** combovalidation **Validates** Web forms on browser and server side. Custom Custom Custom display - Custom Div Custom display , form center, custom display, inline div, Simple. This class can be used to validate Web forms both on browser and server side. It can generate ...



[Miami Painters](#)  
Find Miami Painting Contractors & Painters for Home or Business.  
[yellowpages.com](#)

Chitika | Select

**Ads by Google** [Email Address Validate](#) [Validate PDF](#) [Form Validation](#) [Validate Name](#)

[Classes of Mahmoud Farag](#) > **Email Validation** > [Download](#) > [Support forum](#) > [Blog](#) > [XML](#) [Latest changes](#)

Your custom journey begins **here.**

Call R. Crusoe & Son at 888-490-8049,  
or visit us at [www.rcrusoe.com](http://www.rcrusoe.com).

**Name:** Email Validation [Support forum](#)

---

**Base name:** [importer-mail](#)

---

**Description:** Validate a list of e-mail addresses from a file

---

**Related top rated classes:** [filter](#)

---

**Version:** -

---

**Required PHP version:** 5.0

---

**License:** [GNU General Public License \(GPL\)](#)

---

**All time users:** 272 users

---

**All time rank:** 4742

---

**Week users:** 269 users

---

**Week rank:** 4

[Author](#) [Groups](#) [Detailed description](#)

[User ratings](#) [Applications](#) [Files](#)



## Author



Name: Mahmoud Farag <[e-mail contact](#)>

Published packages: 1 [Browse this author's classes](#)

Country: [Egypt - PHP jobs in Egypt](#)

Home page: ???

## Groups

- [Email](#) Email sending and receiving [View top rated classes](#)
- [PHP 5](#) Classes using PHP 5 specific features [View top rated classes](#)
- [Validation](#) Validation algorithms [View top rated classes](#)

## Detailed description

This class can be used to validate a list of e-mail addresses from a file.

It can import a list of e-mail addresses from a given text file.

The class can validate the e-mail addresses discarding those with invalid syntax or are duplicated.

The filtered list of e-mail addresses is returned as an array.

## Applications that use this class

No application links were specified for this class.

If you know an application of this package, send a message to the [author](#) to add a link here.

## Files

File	Role	Description
<a href="#">emails.txt</a>	Data	Emails for example
<a href="#">example.php</a>	Example	example
<a href="#">validate.class.php</a>	Class	validate multi emails



Choose a Child

World Vision  
www.worldvision.org

# Copy files to IFS

The screenshot displays two windows side-by-side. The top window is a Windows Explorer window titled 'emailValidate' showing a local file system path. The bottom window is an iSeries Navigator window showing a remote file system structure.

**Windows Explorer (emailValidate):**

- Address: C:\Documents and Settings\mike.p\My Documents\Old laptop\Downloads\Software\Open Source\phpClassesOrg\emailValidate
- Files: validate.class (PHP file, 3 KB), example (PHP file, 1 KB), emails (Text Document, 1 KB)

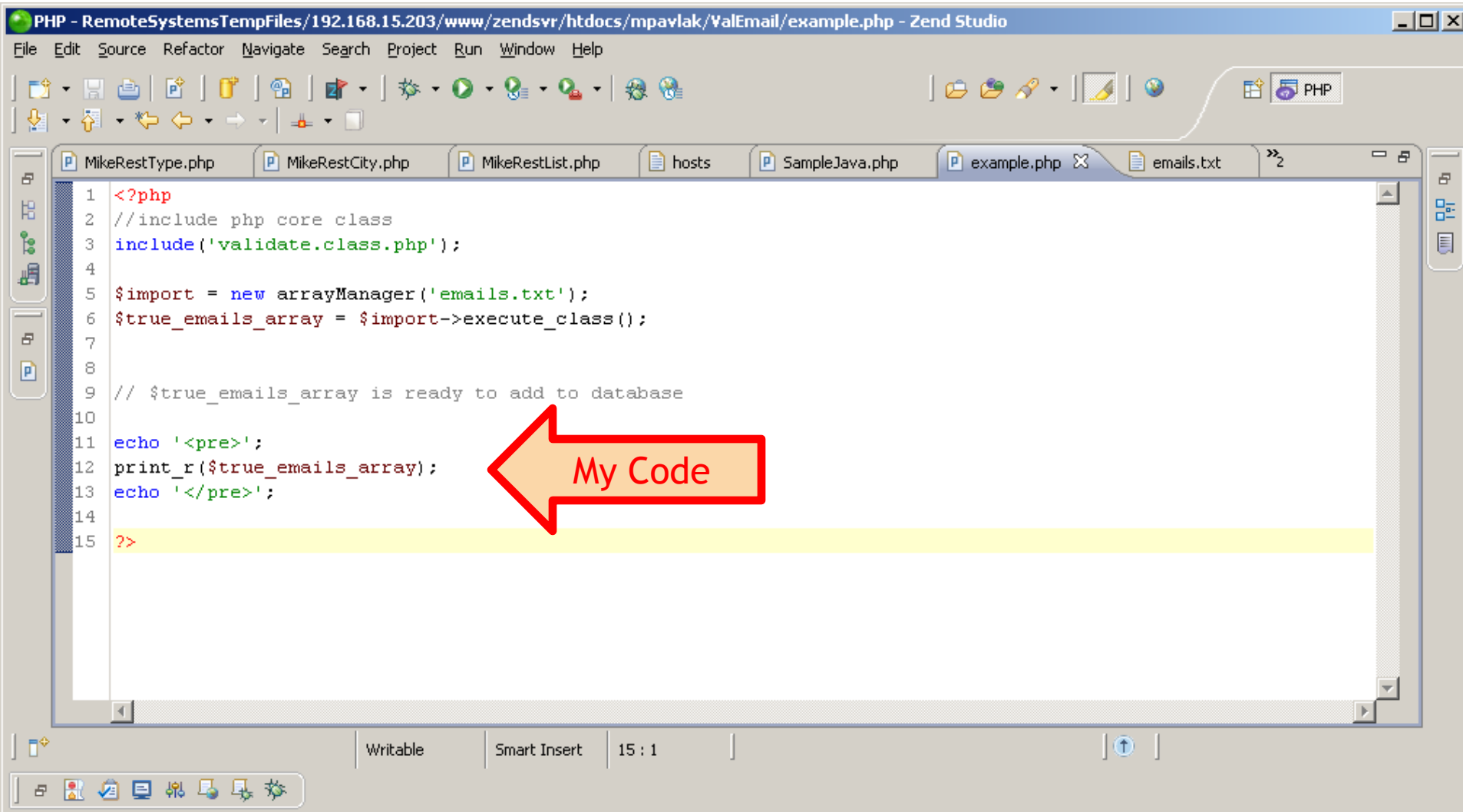
**iSeries Navigator:**

- Environment: My Connections
- 192.168.15.203: ValEmail Include: \*.\*
- 1 minutes old
- Table of files:

Name	Size	Type	Changed
emails.txt	1KB	Text Document	4/29/2010 3:50 PM
example.php	1KB	PHP file	4/29/2010 3:50 PM
validate.class.php	3KB	PHP file	4/29/2010 3:51 PM

1 - 3 of 3 objects

# Test the example...



PHP - RemoteSystemsTempFiles/192.168.15.203/www/zendsvr/htdocs/mpavlak/ValEmail/example.php - Zend Studio

File Edit Source Refactor Navigate Search Project Run Window Help

MikeRestType.php MikeRestCity.php MikeRestList.php hosts SampleJava.php example.php emails.txt

```
1 <?php
2 //include php core class
3 include('validate.class.php');
4
5 $import = new arrayManager('emails.txt');
6 $true_emails_array = $import->execute_class();
7
8
9 // $true_emails_array is ready to add to database
10
11 echo '<pre>';
12 print_r($true_emails_array);
13 echo '</pre>';
14
15 ?>
```

My Code

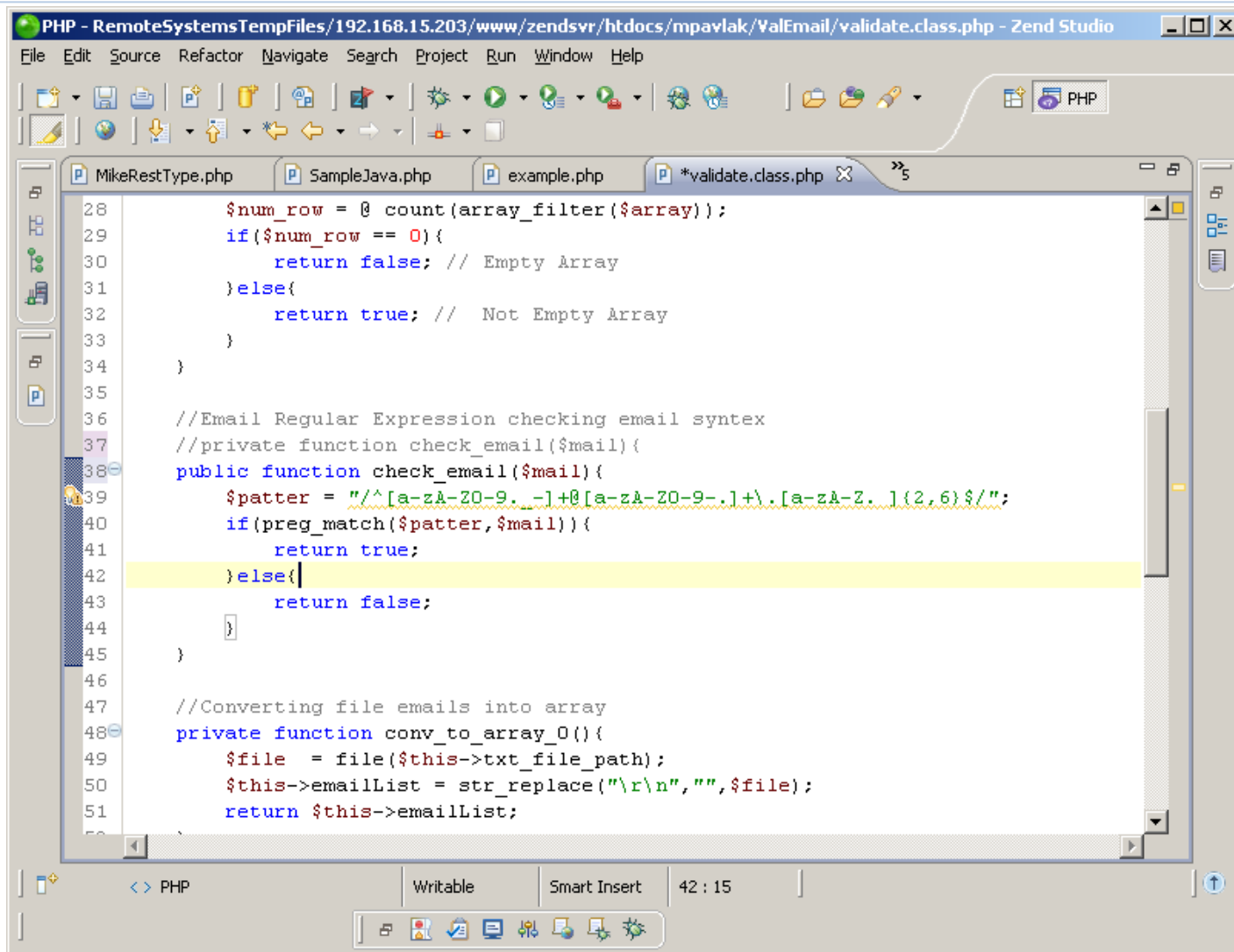
Writable Smart Insert 15 : 1

# Source Data → Array Output

```
emails - Notepad
File Edit Format View Help
'mahmoud@yahoo.com
farag@ymail.com
ahmed@hotmail
zipo@man.com
farag@ymail.com
mahmoud.oop@gmail.com
mahmoud@yahoo.com
ahmed_noo7@hotmail.com
ahmed@msn.com
mahmoud.oop@gmail.com
ahmed@msn.com
me@yahoo.com
dodo@kos.asd
down@hotmail.sss
down@hotmail.sss
down@hotmail.sss
magfy
asda
aljkshd askljdh asd
boss@gmailre.mss
```

```
validate.class.php - Email V... x http://192.168.15.203:10... x
http://192.168.15.203:10088/mpavlak/valemail/example.php
Array
(
    [0] => mahmoud@yahoo.com
    [1] => farag@ymail.com
    [2] => zipo@man.com
    [3] => mahmoud.oop@gmail.com
    [4] => ahmed_noo7@hotmail.com
    [5] => ahmed@msn.com
    [6] => me@yahoo.com
    [7] => dodo@kos.asd
    [8] => down@hotmail.sss
    [9] => boss@gmailre.mss
)
```

# But I only want to check one email...



The screenshot shows the Zend Studio IDE with a PHP file named `validate.class.php` open. The code is as follows:

```
28     $num_row = @ count(array_filter($array));
29     if($num_row == 0){
30         return false; // Empty Array
31     }else{
32         return true; // Not Empty Array
33     }
34 }
35
36 //Email Regular Expression checking email syntax
37 //private function check_email($mail){
38 public function check_email($mail){
39     $patter = "/^[a-zA-Z0-9._-]+@[a-zA-Z0-9-]+\.[a-zA-Z. ]{2,6}$/";
40     if(preg_match($patter,$mail)){
41         return true;
42     }else{
43         return false;
44     }
45 }
46
47 //Converting file emails into array
48 private function conv_to_array_0(){
49     $file = file($this->txt_file_path);
50     $this->emailList = str_replace("\r\n", "", $file);
51     return $this->emailList;
```

# Code and output...

The image shows a PHP IDE window on the left and a browser window on the right. The IDE window displays the code for `example.php`, which includes a class `validate.class.php` and uses an `arrayManager` to process email addresses. The browser window shows the output of the script, which is a PHP array of email addresses and two status messages.

```
1 <?php
2 //include php core class
3 include('validate.class.php');
4
5 $import = new arrayManager('emails.txt');
6 $true_emails_array = $import->execute_class();
7
8
9 // $true_emails_array is ready to add to datab
10
11 echo '<pre>';
12 print_r($true_emails_array);
13 echo '</pre>';
14
15 $email = 'joe@joesbeef.com';
16 $result = $import->check_email($email);
17 if ($result === TRUE)
18     echo "$email -- Email Good";
19 else
20     echo "$email -- Email Bad";
21
22 echo '<br><br>';
23
24 $email = 'joe@joesbee';
25 $result = $import->check_email($email);
26 if ($result === TRUE)
27     echo "$email -- Email Good";
28 else
29     echo "$email -- Email Bad";
30
```

The browser window displays the output of the script:

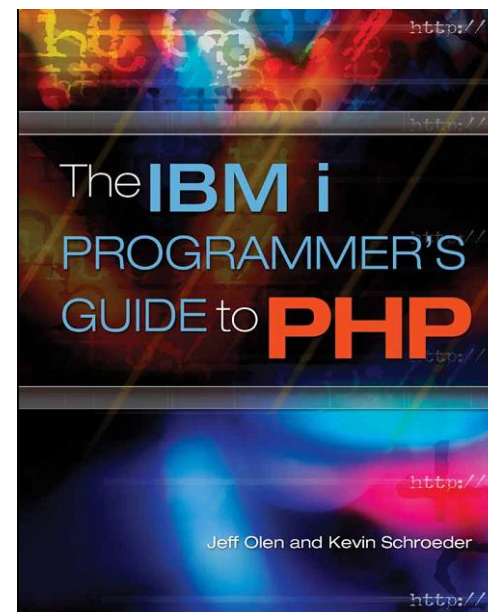
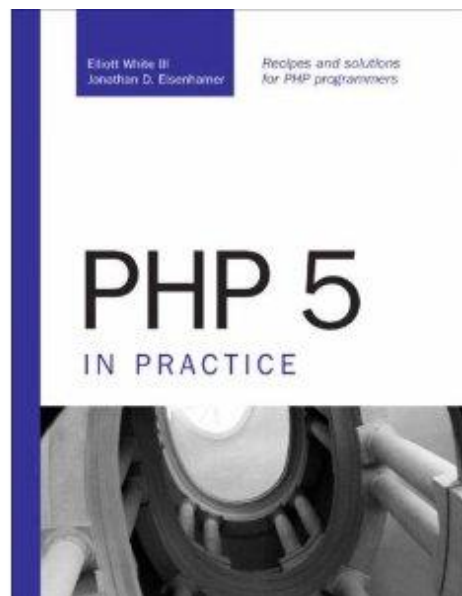
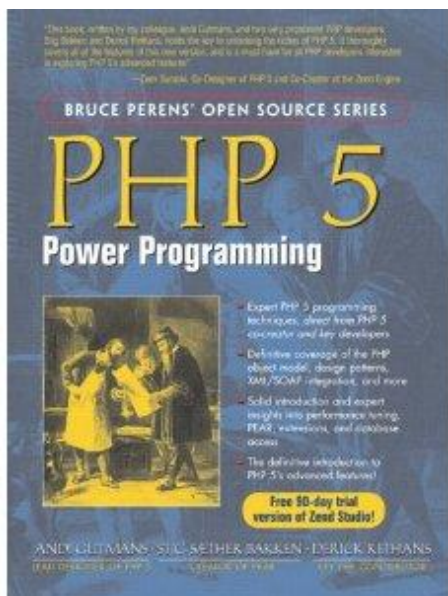
```
Array
(
    [0] => mahmoud@yahoo.com
    [1] => farag@ymail.com
    [2] => zipo@man.com
    [3] => mahmoud.oop@gmail.com
    [4] => ahmed_noo7@hotmail.com
    [5] => ahmed@msn.com
    [6] => me@yahoo.com
    [7] => dodo@kos.asd
    [8] => down@hotmail.sss
    [9] => boss@gmailre.mss
)

joe@joesbeef.com -- Email Good

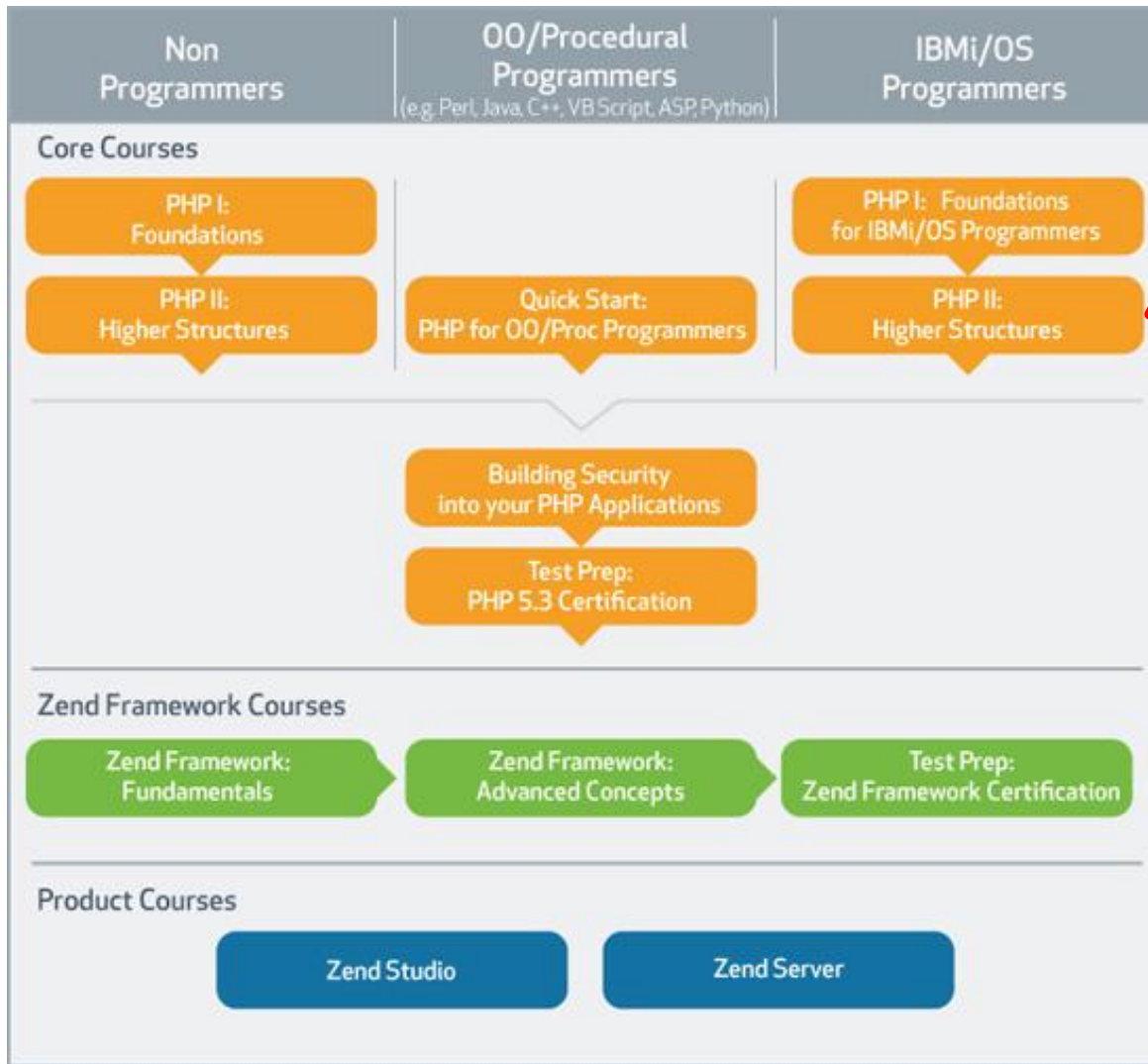
joe@joesbee -- Email Bad
```

# Where do I go next?

# Books!



- Gutmans - PHP 5 Power Programming
- White - PHP 5 in Practice
- Schroeder/Olen - The IBM i



# Build on OOP!

**Starts June 13!**

## PHP II Higher Structures!

Call your account manager or go to the Zend website:  
<http://shop.zend.com/en/php-training.html>

# Resources

---

- Recorded Webinars

- ▶ <http://www.zend.com/en/resources/webinars/i5-os>

- Zend Server for IBM i main page, link to downloads

- ▶ <http://www.zend.com/en/products/server/zend-server-ibm-i>

- Zend Server manual:

- ▶ PDF: <http://www.zend.com/topics/Zend-Server-5-for-IBMi-Reference-Manual.pdf>

- ▶ Online: [http://files.zend.com/help/Zend-Server-5/zend-server.htm#installation\\_guide.htm](http://files.zend.com/help/Zend-Server-5/zend-server.htm#installation_guide.htm)



Zend PHP Conference  
October 17-20, 2011 – Santa Clara, CA

# Join us at ZendCon

## The premier PHP conference!

October 17-19, 2011 – Santa Clara, CA



### Conference Highlights

- Learn PHP best practices for architecture, design and development
- Technical sessions for all knowledge levels
- In-depth tutorials for advanced learning
- PHP Certification courses and testing
- Exhibit hall showcasing the latest products
- Networking opportunities with peers and luminaries

### Conference Topics

- Architecture & Design
- Expanding Horizons
- IBM i
- Lifecycle Best Practices
- NoSQL / Alternative Stores / Search
- PHP Development
- Server/Operations
- SQL
- Zend Framework

[www.zendcon.com](http://www.zendcon.com)

# Q&A

[www.zend.com](http://www.zend.com)

[mike.p@zend.com](mailto:mike.p@zend.com)

*Please fill out your  
Session Evaluation!*

A small thumbnail image of a session evaluation form. The form is titled "2009 Business & Marketing Evaluation" and "Professional Development Evaluation". It contains several sections with checkboxes and a table for ratings. The table has columns for "Strongly Dislike", "Dislike", "Neutral", "Like", and "Strongly Like". The form also includes a section for "Comments" and a "Date" field.