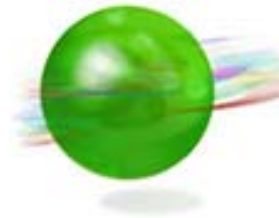


User Guide:

Zend Studio 5.5



By Zend Technologies, Inc.

Disclaimer

The information in this document is subject to change without notice and does not represent a commitment on the part of Zend Technologies Ltd. No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or information storage and retrieval systems, for any purpose other than the purchaser's personal use, without the written permission of Zend Technologies Ltd.

All trademarks mentioned in this document, belong to their respective owners.

© 1999-2007 Zend Technologies Ltd. All rights reserved.

Zend Studio User Guide issued May 2007.

Product Version: Zend Studio 5.5

DN: ZS-UG-180907-5.5 -003

Table of Contents

Chapter 1 - Introduction	8
Zend Studio Components	8
<i>Zend Studio</i>	9
<i>Zend Platform</i>	10
Zend Studio Workflow	12
New in this Version	13
Highlighted Features and Benefits	17
Support	18
Chapter 2 - Zend Studio User Interface	20
Layout	21
Menus and Toolbars	22
Menus and Toolbars	22
<i>Main Menu</i>	22
<i>Main Toolbar</i>	24
Editor Window	25
Browser	26
<i>Enable/Disable the Internal Browser</i>	27
Inspectors Window	27
File Manager	28
Debug Window	29
Messages Window	30
Output Window	30
Customizing Zend Studio	31
Quick Start	32
<i>Starting Zend Studio</i>	32
<i>Section 1: Starting a Project</i>	33
<i>Section 2: Debugging a Project</i>	34
<i>Section 3: Profiling a Project</i>	35
Managing Projects	36
<i>Create a New Project</i>	36
<i>Set Project Properties</i>	37
<i>Project Debugging and Encoding Settings</i>	37
Chapter 3 - File Management	39
Accessing Files	39
<i>Working with Files</i>	39
Filtering Files	39
Working with Remote (Server) Files	41
Adding an FTP Root	41
<i>Using FTP Over SSH2</i>	43
Chapter 4 - Editing	44
Using Code Completion	44
Framework Integration	45
<i>Zend Framework Components</i>	46
Referencing Files / URLs	47
@var tag as Class Type Hint	48
Using Templates	49
Indenting Code	50
Commenting Lines Blocks	50

Inserting HTML Tags	51
Usability Shortcuts/Timesavers	52
Matching Highlighted Elements	53
Clone View	54
Anti-Aliasing Support	55
Code Snippets.....	56
<i>Creating a Code Snippet</i>	56
<i>Editing a Code Snippet</i>	57
<i>Updating the Code Snippet Database</i>	58
Printing	58
<i>Printing an Active File</i>	59
Chapter 5 - Code Navigation	60
Bookmarks (set, demote, go).....	60
<i>Bookmark Manager</i>	61
Finding Matching Brackets	62
Forward Backward Navigation.....	63
Smart Goto Source	63
Goto PHP Resource.....	64
Recent Files	65
Goto Project File	65
Chapter 6 - Web Services	66
SOAP Client.....	67
<i>Creating a Soap Client</i>	67
Incorporating WSDL Files.....	68
Generating WSDL Files - WSDL Generator	71
Creating a Configuration Set	72
Chapter 7 - phpDoc Support	74
phpDoc Block.....	74
Add phpDoc Descriptions.....	75
phpDocumentor Support	76
Create a New Configuration	77
Chapter 8 - Searching	78
Searching Active Files	78
Searching in Multiple Files.....	78
Searching with Regular Expressions	79
Chapter 9 - Code Inspection	80
Inspecting Files	80
Inspecting Projects.....	82
Viewing PHP Functions	83
Chapter 10 - Debugging and Analyzing Code	84
Internal Debugger	85
Remote (Server) Debugging.....	86
<i>Debug URL</i>	87
<i>Running Debug URL</i>	88
Controlling Program Flow	89
<i>Using Breakpoints</i>	89
<i>Conditional Breakpoints</i>	90
Monitoring Program State.....	91
<i>Creating and Monitoring Watches</i>	91

<i>Removing Watches:</i>	92
<i>Tracking the Stack</i>	93
<i>Reviewing Variables/Assigning Values to Variables</i>	94
<i>Assigning a New Value to a Variable</i>	94
<i>Viewing Output in the Output Buffer</i>	95
Communication Tunnel	95
<i>Configuring the Communication Tunnel</i>	95
<i>Broadcasting Port</i>	96
<i>HTTP Authentication</i>	96
<i>Troubleshooting the Communication Tunnel</i>	97
Analyzing Code	98
Platform Integration	99
<i>Enabling Platform</i>	99
Chapter 11 - Profiling	102
Profiler Information Tab	103
Profiler Function Statistics Tab	104
Profiler Call Trace Tab	105
Chapter 12 - Source Control	107
Setting Source Control Default	107
Using Source Control DIFF	108
Configuring Zend Studio for CVS	110
Configuring the Zend Studio - CVS Communication Tunnel	111
Configuring Zend Studio for Subversion	111
Source Control File Status	113
<i>Configuring Source Control File Status</i>	113
Chapter 13 - SQL Support	115
About	115
<i>List of Functions</i>	115
SQL Settings	116
<i>Editing Server Settings</i>	118
<i>Connecting to an SQL Server</i>	119
File Manager: SQL	119
<i>Viewing the Schema Structure of the Database</i>	119
Main Workspace: Data Display	120
<i>About the Data Display</i>	120
<i>Functions, Shortcuts and the Right-click Menu</i>	121
<i>Viewing the Contents of a Table</i>	121
<i>Data Display</i>	122
<i>Data Display: Large Number of Results</i>	123
<i>Setting the Number of Results Displayed on a Page</i>	123
<i>Navigating to Results Not Shown in the Initial Display</i>	123
<i>Organizational Objects</i>	124
<i>Stored Procedures</i>	125
<i>Server Metadata</i>	127
<i>Editing the Contents of a Table</i>	129
<i>Primary Keys</i>	129
<i>Unlock Function</i>	129
<i>Data Display: Large Objects</i>	131
SQL Query Control	133
<i>Controls: Server, Database, and Schema</i>	134
<i>Functions: Go and Clear</i>	134
<i>Re-running a Query from the History Area</i>	134

SQL Messages.....	135
Chapter 14 - Preferences.....	136
Setting Desktop Preferences	137
Setting Editing Preferences.....	138
Setting Code Completion Preferences	140
Setting Debug Preferences.....	142
Colors and Fonts.....	144
<i>Colors and Fonts - General Tab</i>	<i>145</i>
<i>Colors and Fonts - Java Script Tab</i>	<i>145</i>
<i>Colors and Fonts - HTML Tab.....</i>	<i>145</i>
<i>Colors and Fonts - PHP Tab.....</i>	<i>146</i>
<i>Colors and Fonts - CSS Tab</i>	<i>146</i>
<i>Colors and Fonts - XML Tab.....</i>	<i>146</i>
<i>Colors and Fonts - SQL Tab</i>	<i>147</i>
<i>Colors and Fonts - DIFF Tab</i>	<i>147</i>
Configuring the KeyMap	148
<i>Customizing KeyMaps.....</i>	<i>148</i>
<i>KeyMap Properties</i>	<i>149</i>
Setting File Type Preferences.....	155
Setting Template Preferences	156
Setting Source Control Preferences	157
<i>General CVS Settings</i>	<i>157</i>
<i>General Subversion Settings.....</i>	<i>158</i>
Setting Dialog Preferences.....	158
Setting Global SQL Settings.....	159
Proxy Settings.....	160
Chapter 15 - Setting Server Components	161
PHP Settings.....	161
Configuring Zend Platform's Studio Server.....	161
<i>Security Settings</i>	<i>161</i>
<i>IP Permission Management</i>	<i>162</i>
<i>Check for Allowed.....</i>	<i>162</i>
<i>Checked for Denied</i>	<i>162</i>
Chapter 16 - Java Bridge	164
Code Completion	164
Assignments	164
Exclusions.....	165
Changing the JRE.....	166
Adding Java Objects	170
<i>Adding New Packages to the Code Completion Library.....</i>	<i>173</i>
<i>PHP /Java Integration Code Example.....</i>	<i>175</i>
Chapter 17 - Zend Guard™	176
Obfuscation	176
Zend License Manager	176
Encoding Applications	177
Table of Figures	178

Preface

Zend Studio is a functionality rich development environment. As such, a greater understanding of different features and functionality is required to fully benefit from the Zend Studio's capabilities and features. This User Guide has been designed to describe Zend Studio from the basic user interface layout to more complex features, each in their own dedicated chapter.

Chapters one, two, and three are introductions, describing the layout, main menus, and design, along with who should read this guide, and how to maximize the benefits of using Zend Studio to develop PHP Web applications.

The chapters that follow are each dedicated to a different Zend Studio feature, beginning with how to start using Zend Studio. The next chapters highlight the different features and how they can be used in a developers working environment.

The functionality of each screen is elaborated in the online Help, to provide context sensitive instructions on how to work with Zend Studio.

Audience

Zend Studio is a development environment that is directed towards developers and their requirements.

Chapter 1 - Introduction

IN THIS CHAPTER...

ZEND STUDIO COMPONENTS

ZEND STUDIO WORKFLOW

NEW IN THIS VERSION

HIGHLIGHTED FEATURES AND BENEFITS

SUPPORT

The Zend Studio User Guide is a task-oriented guide to procedures typically performed by developers. Throughout this online help you will find descriptions of the Zend Studio Features and functionality and how they can be used to streamline the development process using Zend Studio.

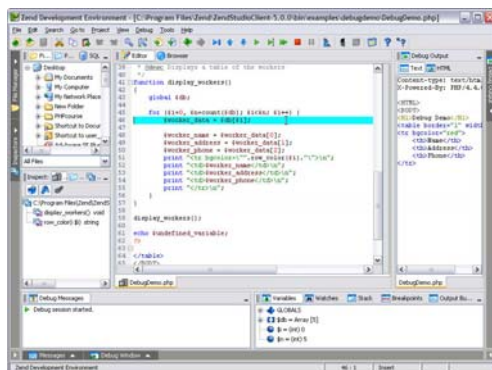
Zend Studio is a complete development environment for editing, debugging and optimizing code for PHP applications. The fully integrated user interface provides features that help streamline the development process.

This User Guide is intended for Zend Studio 5.5.

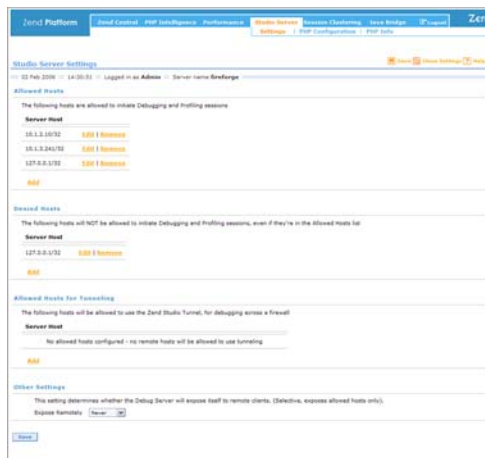
Zend Studio Components

Zend Studio consists of two main components:

Zend Studio – IDE.



Zend Platform – Remote interaction capabilities for working with remote servers.



Zend Studio

A powerful IDE for writing and maintaining PHP applications, Zend Studio provides a feature rich environment that includes, the following development tools:

- **Debugging** - Control the running process of your PHP application and receive valuable information on errors, call stack, variables and output.
- **Project and File Management** - Manage your files and folders in Zend Studio, including FTP functionality.
- **Project Inspector and File Inspector** - For viewing and navigating functions, constants, soap clients etc.
- **Code Completion** - Offers a wide selection of code completion features, including PHP, HTML, classes, member variables, variables, keyword, and object code completion listings. Lists are comprised from declarations as well as most of the standard PHP and HTML.
- **Code Templates** - Templates help to write code quickly and accurately. Templates are shortcuts used to insert a framework for the segment of code you are about to write.
- **Syntax Highlighting** - Apply automatic text highlighting to different syntax elements.
- **Code Indentation** - Use "as you type" automated indentation or apply indentation automatically - "all at once" - to format PHP code according to convention.
- **Online PHP Manual** - Find out more about a PHP function by directly referencing the online PHP manual opened in Zend Studio's Browser window (one button activation).
- **Go to navigation** - Provides multiple methods for jumping to the text and code you want.
- **Customizable Shortcut keys** - Define and customize keyboard shortcuts based on commonly used keystrokes to accommodate user preferences.
- **Version Control** - Zend Studio integrates directly with CVS and Subversion content management systems.

Zend Platform

Designed for IT personnel and businesses that require industrial-strength applications in highly reliable production environments, Zend Platform offers high performance and scalability to provide the best possible Web experience and response time.

Zend Platform uniquely guarantees application up-time and reliability through enhanced PHP monitoring and immediate problem resolution, removing troubleshooting and guesswork out of the equation.

Note:

The Zend Studio Installation includes Zend Platform and a free, 30-day license for its use. At any time during the 30 days, or afterwards, development users can extend this free license for Zend Platform, for development purposes only, by contacting Zend from the Zend Platform GUI or the appropriate page at www.zend.com and completing the appropriate License acquisition form.

Zend Platform is a browser-based application, that includes the following functionality:

- Zend Central - Central management functionality
- PHP Intelligence - Monitoring and event generation capabilities
- Performance - Performance enhancement tools
- Studio Server - Integration settings and PHP configurations
- Session Clustering - Session clustering statistics
- Java Bridge - incorporate a Java environment to enrich your applications

The Zend Platform functionality is given unlimited for a 30 day trial period. In this time users can experience and evaluate Zend Platform features such as Performance, PHP Intelligence, Session Clustering and the Java Bridge.

To find out more about Zend Platform's advanced workflow, performance and analysis features please see the Zend Platform user guide and learn how you can leverage these features to improve your organization's work processes product quality and performance. Zend Studio users will be able to use Zend Platform to replace the formerly used Standalone Studio Server Component. With Zend Platform's Studio Server Component users will be given the necessary functionality for working with remote servers.

The Zend Platform Studio Server Component's functionality includes:

- Remote debugging and profiling capabilities to remote Web servers
- Facilitates setting up a PHP-enabled Web servers (even if one does not already exist)
- Zend Debugger
- Zend Server Center
- Apache Web Server
- PHP 4 and 5 support

Zend Platform has a separate browser based user interface that is accessed from:

`http://<your-host>:<port>/ZendPlatform/`.

Use the password as you defined in the Zend Platform Installation.

Zend Studio Workflow

The illustration below shows how Zend components interact to debug a PHP application. All activities originate in the Zend Studio component.

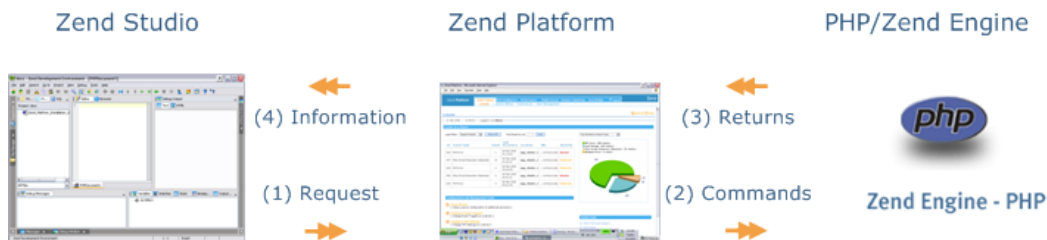


Figure: 1 - Zend Studio Workflow

1. While running a script in debug mode, a request is issued from the Zend Studio and sent to Zend Platform's Studio Server Component. A typical request might be a breakpoint, i.e., a command to stop the execution of the application script at a specified point.
2. The Studio Server Component issues commands to the PHP Engine to execute the request and return information about the input code. For example, when running the debugger to a breakpoint, the script stops at the breakpoint and generates the debugger returns (results).
3. The PHP/Zend Engine in turn reports the information it gathers such as: output, variables, call stack, and execution errors to the Studio Server Component.
4. The Studio Server Component takes the information it has received from the PHP Engine and sends it to Zend Studio to be displayed in the Debug Output and Messages windows.

New in this Version

Supported Versions:

- Zend Studio 5.5
- Zend Studio 5.2
- Zend Studio 5.1
- Zend Studio 5.0

Zend Studio 5.5 supports the following new features and improved functionality:

Anti Alias Support

Antialiasing is a software technique for smoothing out jagged-looking, step-like lines that should be smooth (*jaggies*).

Editor

Open <Filename/URL>: Enables the Editor to open referenced files as well as URLs.

Framework Integration

Integrates Zend Framework's APIs into Studio 5.5 's Code Completion features. Framework, based on the MVS (Model-View-Controller) pattern, enables you to create Web sites with virtually no effort.

Java Bridge

Zend Studio 5.5 supports Java Object instances. After instantiation of the PHP's Java object, the variable it was assigned to assumes the properties and methods of the Java class.

Platform Integration

Zend Studio 5.5 can activate Platform's Web UI. Using Studio 5.5, you can:

- Edit Platform's server URL.
- Activate Platform's GUI.

Source Control File Status

The Source Control File Status feature provides visual representation of the files' Subversion or CVS status. It does this by using colored fonts when displaying the file *names* in Studio's Project Explorer.

File Status highlights

- Newly added files.
- Modified files.
- Files that were merged with conflicts.
- Files not under the VCS (Version Control System).

- Files that are currently Up-To-Date (i.e., unmodified since last commit).

Web Services Support

Zend Studio 5.5 now supports instantiation of SoapClients using URLs in their constructor.

Zend Studio 5.2 supports the following new features and improved functionality:

Debugger

Integrated installation of Zend Platform and full compatibility with it.

PHP Language

Updating PHP elements code completion according to the latest changes in PHP.

Help

New daily tips

Zend Studio 5.1 supports the following new features and improved functionality:

Performance

- Enhanced performance for project loading and saving.
- Decreased the number of resources used by the application.

Text Encoding

New text Encoding per project option

Web Services Support (SOAP)

Global options for binding style (RPC and document oriented)

SQL Support

- MySQL 5.0 support
- Date and Time pickers to edit SQL table data

Zend Studio 5.0 supports the following new features and improved functionality:

FTP

Enhanced FTP support to include FTPS (FTP over SSL) and the ability to drag and drop files into or from your local file system.

Import and Export Templates

Share Templates with other PHP developers. Import templates or share your code with others by exporting templates.

Improved phpDoc Support

Indented phpDoc Blocks can be automatically inserted from the Inspectors window or by manually typing `/**/` and pressing Enter. Inserts Basic and Advanced (Adds Stubs) phpDoc Blocks

The first PHPDoc in files that is not associated with a class/function/constant, will automatically become the file's Doc Block.

Code Inspection

Enhanced Inspection window that is linked to the editor. The Code Inspector now lists all Classes, functions, Include Files and Soap Clients with grouping and sorting options.

Web Services Support (SOAP)

Generate WSDL files from PHP code (Tools -> WSDL Generator).

Other advanced WSDL features include:

- Parsing WSDL files used by SOAP client.
- Inspector view of WSDL classes and functions.
- Parsing WSDL documentation and showing it as a description for WSDL functions in inspectors and code completion.
- SOAP client object Code Completion includes all WSDL defined functions.

Subversion Support

Zend has added an additional version control system to the Zend Studio IDE. In addition to the existing CVS integration, users can now benefit from Subversion integration.

Zend Studio supports all common subversion actions, such as: Update, Commit, Add, Delete, Revert, Resolve, Status, Diff, Log and Checkout.

Both CVS and Subversion integrate with Project files and can be easily customized from the Preferences menu.

Editor

The Zend Studio Editor - the heart of the IDE now supports the following enhancements and new features:

- Strip trailing spaces before saving.
- Quick-change font sizes in the editor using Ctrl + mouse scroll.
- Automatically close PHP quotes, back quotes, double quotes, parentheses, and square brackets.
- Smart deletion support.
- Close HTML tags automatically after '>'.
- Automatic indentation and insertion of PHP curly brackets.
- Code folding.
- Code Completion Preferences - new support for nested functions and for function return values.

Go to PHP Resource Dialog

Navigate to any PHP resource in the Project: classes, functions, and constants. Including auto-complete and advanced filtering by classes, functions, and constants.

Go to Project File Dialog

Navigate to any file in the Project. Includes auto-complete and advanced file name filtering.

Open Files With External Programs

Open files with external programs based on program definitions for each binary file type.

Internet Browser Integration (Windows IE users only)

Embedded Internet Browser window that includes:

- Multiple tabs.
- Basic browsing actions: Back, Forward, Stop, Refresh.
- Integrated Debugger Toolbar with setting options.
- Opening the online PHP Manual directly in the internal browser.

Help Agent

Displays a useful Tip when using a feature for the first time.

Highlighted Features and Benefits

Zend Studio combines all the tools that you regularly work with to develop your application in one unified interface.

From the integrated Zend Studio workspace you can:

- Edit PHP, HTML, and JavaScript source code
- Debug your application
- Profile your application to find and fix performance bottlenecks
- Update, commit or perform DIFFs using the CVS and Subversion integration
- Bundle multiple files and directories into a single Project entity, making navigating and searching your application simple
- Display and study the hierarchy of the PHP functions, classes, and projects
- State-of-the-art code completion for every aspect of PHP
- Code templates for structuring PHP code rapidly
- Code Snippets for rapid application development
- phpDoc support for generating API documentation and building Code Completion library
- Syntax highlighting for PHP, HTML, and JavaScript code -- in the active Editor window -- and they will all be accurately color-coded at the same time.

Note:

Zend Studio's editor is currently the only editor on the market that supports all the different constructs of PHP, and the only one around that fully supports PHP 5's syntax.

- Seamlessly edit and deploy files on FTP servers
- SQL integration for interfacing with SQL databases

Zend Studio includes innovative features that simply don't exist anywhere else:

- Analyze your code using Zend Studio's built-in static code analysis tool. Find problems in your application even before you run it!
- Debug and Profile your application right from the browser. For the first time in the history of web development, debugging even the most complicated forms or session-based applications is one click away.

Support

Zend Technologies provides a wide range of resources for obtaining additional information and support, such as from developer resources, updates on new features, usability tips, and more.

Developer Center

The Developer Center is your online destination for up to date information, articles, and resources to assist in developing professional PHP applications with Zend Studio. The Zend Developer Center features the following:

- Technical Articles
- Plug-ins
- Support Resources
- Tips & Tricks

Visit: http://www.zend.com/products/zend_studio/developer_center.

Developer Zone

The Zend Developer Zone is the leading resource center for PHP developers. Learn about PHP and meet the experts. The Zend Developer Zone features the following:

- The PHP 5 Info Center
- Articles and Tutorials
- PHP and PEAR News Weeklies
- Worldwide Job Classifieds

Visit: www.zend.com/devzone.

Code Gallery (online)

At Zend, we encourage our developer community to take part in actively publishing their code and, in return, have a place where developers can benefit from others. Code in the gallery is reviewed and ranked by other developers and you can even participate in prize-winning contests.

Visit: <http://www.zend.com/codex.php>.

Discussion Forums

Our project managers and developers constantly visit and participate in a wide range of forums targeted for Zend Product owners and the PHP developer community.

Visit: <http://www.zend.com/phorum/>.

Newsletter

Zend's monthly Newsletter contains the hottest updates including special promotions and developer information such as summaries of articles featured on Zend.com. Every month, the Studio Development Team provides two usability tips that will assist in working smarter and more effectively with Zend Studio. Sign-up for the Zend Newsletter in the Developers Zone.

Visit: www.zend.com/devzone.

Zend Studio Support

Zend Studio Support provides Zend Product owners and prospective Zend Product owners with information regarding: System Requirements, Installation Instructions, General FAQ, Quick Start Guide, and much more.

Visit: http://www.zend.com/products/zend_studio/support.

Feedback

Send feedback, questions and comments on the Online Help and Documentation to: documentation@zend.com.

Send feedback, questions and comments on the Zend Studio IDE: studio-feedback@zend.com.

Note

Zend Studio system requirements for all versions can be found in the system requirements section on zend.com. (http://www.zend.com/products/zend_studio/system_requirements)

Chapter 2 - Zend Studio User Interface

IN THIS CHAPTER...

MENUS AND TOOLBARS

EDITOR

INTERNAL BROWSER

FILE MANAGER

INSPECTORS WINDOW

MESSAGES WINDOW

DEBUG WINDOW

OUTPUT WINDOW

CUSTOMIZING THE DESKTOP

Zend Studio's user interface consists of a multi-pane display including seven major windows. The various windows allow operations to be performed on code, or display the output of operations performed on code, as shown in the image below:

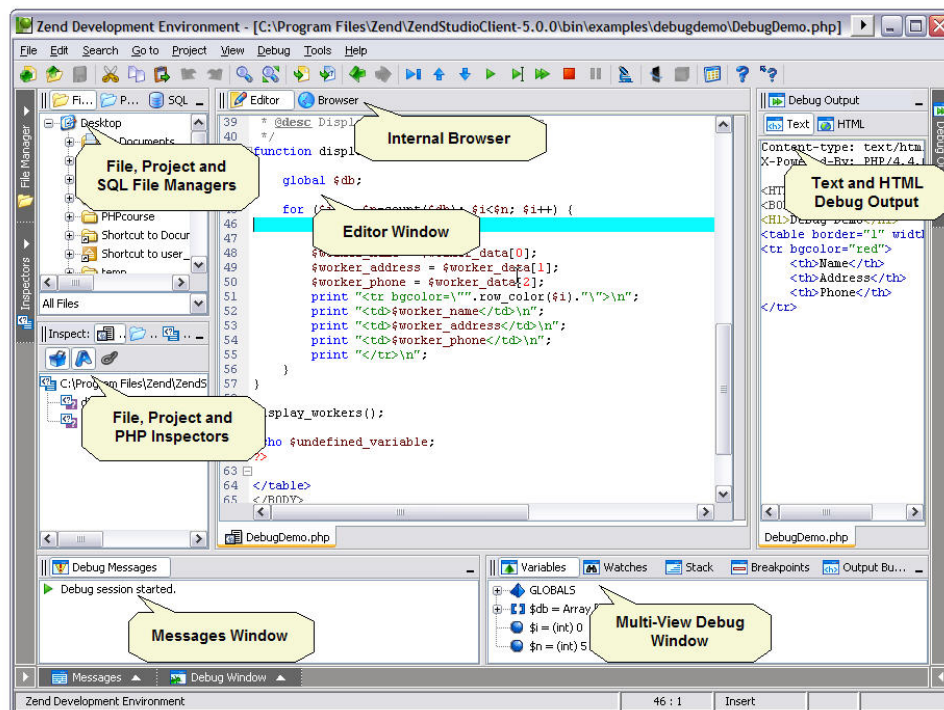


Figure: 2 - Zend Studio User Interface

The user interface includes:

- **System features** - such as the main menu and toolbar
- **Control features** - such as development, debugging, and deployment capabilities
- **Management features** - such as file, project, and debug settings

All these features are immediately reachable from a single view.

The Zend Studio windows are as follows:

- **File Manager** - an internal file system browser, to view the contents of all disks, mounted disks, mapping, etc.
- **Inspectors** - view constants, classes, and members declared throughout the Project files.
- **Editor** - the main area for writing and editing code.
- **Browser**¹ - an internal browser that functions as a standard web browser.
- **Debug Output** - a display window for viewing the output of the script currently being run in Zend Studio. Besides viewing the output, from this window you can Print, Copy, View (in a Browser), or Save the output to file.
- **Messages** - displays debug messages generated by the Debugger (Notices, Warnings, and Errors).
- **Debug** - an interactive window for managing tasks related to testing and correcting PHP code. The various debugging functions are organized into five tabs: Break Point, Stack, Watches, Variables, and Output Buffer.

Layout

Windows can be displayed or removed from the main window. Each window is a "floating and docking" window to ensure that components are easily accessed at all times.

Components are repositioned by dragging and dropping component windows to any area in the workspace. Most windows are docking windows that can be easily opened, closed and repositioned around the main screen or set to stay on top of the main screen.

If the area is not a docking area the window will remain on top.

¹ Available for windows users only.

Menus and Toolbars

Zend Studio menus and toolbars are used for instructing the IDE to initiate various development and system functions.

A wide selection of keyboard shortcuts are also provided to suit commonly used configurations (Visual Studio, Mac and Emacs). Keyboard shortcuts and settings can be configured from the Preferences menu (Tools | Preferences | KeyMap).

This paragraph describes the main toolbar and main menu features and functionality.

Main Menu

System and development commands are accessed from the main menu.

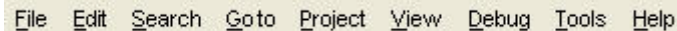


Figure: 3 - Main Menu

The Commands are as follows:

Menu Option	Description
File	<p>Basic file menu options such as opening/closing/saving/printing project and source files as well as FTP server configuration.</p> <p>The File menu options are: New File, Open File, Close (File) Close All, Save, Save As, Save All, Recent Projects, Recent Files, Recent Files Window, Open Project, Close Project, Add FTP Server, Print and Exit.</p>
Edit	<p>Standard clipboard operations, code editing options, and bookmark management.</p> <p>The Edit menu options are: Cut, Copy, Paste, Select All, Indent Code, Wrap Lines, Code Folding, Undo, Redo, To Lower Case, To Upper Case, Duplicate Line/Selection, Erase Line, Add/Remove Line Comments, Add/Remove Block Comment, Show Snippets, Create New Snippet, Add/Remove Bookmark, Remove all Bookmarks, Show Bookmark Dialog and HTML Tags.</p>
Search	<p>Find and replace operations. Search for words and symbols within an editor file.</p> <p>The Search menu options are: Find, Find and Replace, Find Next, Find Previous and Find in Files.</p>
Goto	<p>Instantly jump to a specific place in the project such as: a function's declaration source code, bookmarks, lines, etc.</p> <p>The Goto menu options are: Goto File, Goto Resource, Goto Line, Goto Matching Bracket, Goto Next Bookmark, Goto next Project Bookmark, Back, Forward and Open Next Messages Entry.</p>
Project	<p>Project management options such as project properties, opening, closing, and saving projects.</p> <p>The Project menu options are: New Project, Open Project, Save Project, Close Project, Check Included Files, Add to Project and Project Properties.</p>




















Menu Option	Description
View	<p>Use the View menu to show or hide component windows.</p> <p>The View menu options are: Show/Hide File Manager Window, Show/Hide Messages Window, Show/Hide Debug Window, Show/Hide Debug Output Window and Show/Hide Inspectors Window</p>
Debug	<p>Debugging and profiling actions and control.</p> <p>The Debug menu options are: Add/ Remove Breakpoint, Add Watch, Remove all Breakpoints, Step Over, Step Out, Step Into, Go, Go to Cursor, Run, Check Debug Server Connection, Debug URL, Profile URL, Tunneling Settings, Show in Browser, Stop Debugger and Pause Debugger</p>
Tools	<p>Run and use internal (debug, WSDL Generator) and external tools (PHPDocumentor, Zend Platform).</p> <p>The Tools menu options are: Preferences, Check Debug Server Connection, Debug URL, Profile URL, Tunneling Settings, Rebuild Inspection Data, Proxy Settings, Zend Platform, Encode Project, CVS/Subversion, PHPDocumentor, WSDL Generator and Analyze Code.</p>
Help	<p>Get answers to usability questions from the online help and tip of the day systems. Get updated with the latest product versions and register your copy of Zend Studio.</p> <p>The Help menu options are: Help Topics, Tip of the Day, Check for New Version, Support, Send Feedback, Register, Protect your PHP Code, Speed up Your PHP Site and About.</p>







Main Toolbar

Shortcuts to frequently used functionality:



Figure: 4 - Main Toolbar

Shortcut	Name	Description
	New File	Creates a new file and opens it in the editing window.
	Open File	Opens an existing file.
	Save	Saves the current active file.
	Cut	Removes the selected text (file, or directory) and copies it to the clipboard.
	Copy	Copies the selected text (file, or directory) to the clipboard.
	Paste	Copies the clipboard contents to the current cursor location.
	Undo	Reverses recent changes.
	Redo	Reapplies actions reversed with Undo.
	Find	Activates the Find dialog for searching for text within the active file.
	Find and Replace	Activates the Find and Replace dialog for finding and replacing text within the active file.
	Goto File	Use Goto File to open project files and files already open in the editor.
	Goto Resource	Go to functions, classes, and constants in project files and open files.
	Back and Forward	Jump between edited text in the active file or in previously visited files, following your path of actions.
	Step Over	Executes the current line of code and stops at the next executable line or breakpoint.
	Step Out	Executes the remaining code of the called function or file and stops at the next executable line or breakpoint after returning to the calling script.
	Step Into	Executes the current line of code and stops at the next executable line or breakpoint. If the line has a called function or file, execution stops at the first executable line inside the called function or file.
	Go	Executes the active script. The execution script stops if it encounters a breakpoint.
	Go to Cursor	Executes the active script up until the editing cursor is encountered in the line of code.
	Run	Executes to the end of the active script without breaking.

Shortcut	Name	Description
	Stop Debugger	Terminates the debug process.
	Pause Debugger	Breaks the execution of an application (Debug Session).
	Analyze Code	Runs the Code Analyzer on the active file.
	Zend Guard	(Shortcut to Zend Guard.) . If Zend Guard is currently installed in your system, this shortcut encodes your project. For details on Zend Guard, go to Help Protect your PHP Code!.
	Zend Platform	Takes you to Zend Platform.
	Open Preferences	Opens the Preferences window.

Editor Window

The Editor window is the main area for writing and editing code. Multiple Editor windows can be open at the same time.

Editor Right-Click Menu Options:

Close File, Save, Save As, Cut, Copy, Paste, Indent Code, Add/Remove Breakpoint, Add Watch, Add to Project, Analyze Code and CVS/Subversion.

Advanced Menu Options:

- **Clone view** - opens an identical copy of the file in a separate docking window.
- **Show Snippets** - opens the code Snippets dialog (Edit | Show Snippets) to add, edit, and view code snippets.
- **Create New Snippet** - creates a new snippet by importing the files content in to the snippet editor, where the code can be changed if necessary.

Note:

The Editor window is not a Docking window.

Browser

The Browser is an internal browser that functions as a regular web browser for searching and viewing web pages and debugging local web applications.

Compatibility

Currently supported for Windows IE users only.

Debugging is enabled when Zend Platform's Studio Server Component is installed on the Web server and Zend Studio is an authorized IP.

Note:

The internal browser cannot be used to remote debug Web applications.

Functionality

Allows users to open the PHP Manual directly from Zend Studio to quickly find additional information about PHP (by selecting a function and pressing F1 on the code, or through the Inspectors view).

- **Browser Right-Click Menu Options:**

Back, Forward, Save Background As, Set as Background, Copy Background, Set as Desktop Item, Select All, Paste, Create Shortcut, Add to Favorites, View Source, Encoding, Print, Refresh, Zend Studio – Debug Current Page, Zend Studio – Debug Next Page and Properties.

Debug and Profile web pages

Browse to a specific web page and use the internal browser's toolbar to locally Debug and Profile web pages .

Note:

Make sure Debugging and Profiling are set up before using these features.

Internal Browser Toolbar Options:



Figure: 5 - Browser Toolbar

The Browser Toolbar offers the following functions:

- **Debug Current Page** - Use this button to debug the page currently open in the browser, or use the drop down menu to select one of the additional options: Next Page Only, All Forms (POST) or All Pages on this Site. The Debugger results are displayed in the Debug output window.
- **Profile Current Page** - Use this button to profile the page currently open in the browser or use the drop down menu to select one of the additional

options: Next Page Only, All Forms (POST) or All Pages on this Site. The profiler results are displayed in the profiler output window.

- **Settings** - Determine from where to take source files (server or local copy) for debugging and profiling code.

The Debug Toolbar also includes the following standard navigation actions: Open in a new tab, Go Back, Go Forward, Stop, Refresh, and Address bar.

Enable/Disable the Internal Browser

Disabling the Internal Browser will remove the Browser entirely from the display and the only way to make the Browser visible again is by enabling the Internal Browser.

To disable/enable the Internal Browser:

1. From the main menu, select Tools | Preferences.
2. Select the Desktop Tab and choose the option "Use Internal Browser".
 - a. Select Enabled to make the Internal Browser appear.
 - b. Select Disabled to remove the Internal Browser view from the Main Screen.

Note:

The Browser is not a Docking window.

Inspectors Window

The Inspectors window is used to graphically map code to display Classes, Functions, constants, and Include Files.

The Inspectors window includes three inspection tabs:

File Inspector Tab

Displays in a tree directory all Class names and members, functions, and constants, which are declared in the current active file.

- **Right-click menu options:**
Add Description and Goto Source

Project Inspector Tab

Displays in a tree directory all Class names and its members, functions and constants, which are declared in the files belonging to the project or open in the Editor.

- **Right-click menu options:**
Add Description and Goto Source

PHP Tab

Displays a tree directory of all the PHP Classes and Functions. When a function is selected, pressing F1 will open the PHP manual in the Internal browser.

- **Right-click menu options:**
Open Manual

To find a specific code element in an active file or project, use the Go to resource option from the "Go to" menu to filter by Class, Function and Constant.

To find out more about navigating through code elements go to Code Navigation on page 60.

Note:

The Inspectors window is a Docking window

File Manager

The File Manager is an internal file system browser to view the contents of all disks, mounted disks, mapping, etc.

The File Manager includes three tabs:

File System Tab

Displays file tree directory of all local, network, and FTP files and folders.

- **Right-click menu options:**
New Project, Open Project, Add FTP Server and Refresh.
- **Right-click menu options when selecting an item:**
Open File, Cut, Copy, Paste, Delete, Rename, Add to Project and Refresh

Project Tab

Displays file tree directory of all local, network, and FTP files and folders in the active project.

- **Right-click menu options:**
New Project, Open Project, Save Project, Close Project, Check Included Files, Add to Project, Remove All, Refresh, Encode Project and Project Properties
- **Right-click menu options when selecting an item:**
Open File, Cut, Copy, Paste, Delete, Rename, Remove from Project, Remove All, Refresh, Analyze Code, PHPDocumentor and CVS/Subversion.

SQL Tab

SQL Server support.

- **Right-click menu options:**
Add Server and Global Settings.
- **Right-click menu options when selecting an item:**
Refresh.

Note:

The File Manager window is a Docking window.

Debug Window

The Debug window is an interactive window for managing tasks related to testing and correcting the code for PHP applications. The various debugging functions are organized into five tabs: Break Points, Stack, Watches, Variables, and Output Buffer.

Description

The Debug window includes five debug tabs:

- **Breakpoints** - Displays the defined breakpoints
 - **Right-click menu options:**
Remove All, Enable All and Disable All.
 - **Right-click menu options when selecting an item:**
Remove, Enable, Disable, Go to Source and Edit Condition.
- **Stack Tab** - Displays the state of call stack while debugging
 - **No right-click menu**
- **Watches** - Displays defined watches
 - **Right-click menu options:**
Add Watch and Remove All
 - **Right-click menu options when selecting an item:**
Edit Expression, Remove, Assign Value and Copy Value.
- **Variables** - Displays both Global and Local Variables while debugging
 - **No right-click menu**
- **Output Buffer** - Displays the buffered data from a script while debugging
 - **Right-click menu options:**
Print, Copy, Save Output and Clear.

Note:

The Debug window is a Docking window

Messages Window

The Messages window displays messages generated by the Zend Studio components. There are three types: Notice, Warning, and Error.

Description

To View messages generated by Zend Studio for CVS/Subversion, WSDL (Web Services), phpDocumentor, Debugger Code Analyzer and Find

Right-click menu options²:

Print, Copy, Save Output, Clear. and Debug Message Filters (Opens the Preferences window in the Debug tab).

Note:

The Messages window is a Docking window

Output Window

The Output window is a display window for viewing the output of the script currently being run by Zend Studio. Besides viewing the output, from this window you can Print, Copy, View (in a Browser), or Save the output to file.

Description

Displays the output generated by PHP script execution.

The Zend Studio Windows version includes a second Output window tab for viewing output as HTML.

Text Tab right-click menu options:

Close, Print, Copy, Show in Browser, Save Output and Clear.

Note:

The Output window is a Docking window

² Messages tabs for other components may have slightly different right-click menu options.

Customizing Zend Studio

Settings are customized and controlled from the Preferences Window (Tools | Preferences).


The Preferences Window is divided into tabs representing the customizable preferences as follows:

- **Desktop** - Customize the desktop icons, fonts, background colors, language, and more.
- **Editing** - Customize editing tools and appearance.
- **Code Completion** - Control PHP and HTML code completion
- **Colors & Fonts** - Contains color assignments and font settings for Syntax Highlighting (General, PHP, HTML, Javascript, CSS, XML, SQL) and for Highlighting DIFF elements.
- **Debug** - Customize the debugger process.
- **Keymap** - Customize shortcuts.
- **File Types** - Customize the list of file types and associated file extensions.
- **Templates** - Add, Edit, or Remove templates.
- **Source Control** – Settings and configurations for supported version control management tools: CVS and Subversion.
- **Dialogs** - Customize optional dialog prompts.
- **SQL** - SQL options and preferences.

Note:

Restore default settings by right clicking in the Preferences Window and choosing "Restore all Defaults" from the menu.

Quick Start

The Zend Studio Quick Start helps you to "get-up-and-running" immediately. To reach the debug demo, click the Debug Demo button  that appears in the Tip of the Day window. The Quick Start instructions appear in this User Guide and can also be viewed directly from Zend Studio's online Help.

Quick Start is divided into three main sections:


- Section 1 teaches you to open a new document and use the Code Completion window.
- Section 2 teaches you to run a brief Debug Session.
- Section 3 teaches you how to use the Profiler.

Starting Zend Studio

This section details how to start Zend Studio after installation.


Starting Zend Studio is straightforward.

Windows

Click the icon  on your desktop or go to the installation folder and click the exec file ZDE.exe. The default installation is located at:

C:\Program Files\Zend\ZendStudio-<VERSION>\bin\ZDE.exe

Mac

Click the icon  on your desktop or go to the installation folder and click the exec file ZDE.exe. The default installation is located at:

/Applications/Zend/ZendStudio-<VERSION>/bin/ZDE.app

Linux

Run the executable binary named **ZDE** located in the install folder. The default folder is:

/usr/local/Zend/ZendStudio-<VERSION>/bin/ZDE

Section 1: Starting a Project

This section describes the procedure for entering several lines of code into a blank document.

In general, Zend Studio's Code Completion feature automatically displays the relevant list of completion options based on its identifying the code section as PHP or HTML. If Code Completion is used within a PHP section, only PHP functions will be offered in the list. PHP sections are identified as lines of code between `<?php` and `?>` (or `<?` to `?>`) tags. In all areas other than PHP-HTML Code Completion windows will appear.

Note:


If Support ASP Tags is selected as an Editing option, Code Completion treats ASP sections of code the same as PHP sections of code. For details on supporting ASP tags, see Setting Editing Preferences.

Quick Start Section 1

1. To start a new file, go to the main toolbar and select, **File | New File** (CTRL+N). A blank Editor window opens.
2. In the Editor window, type `<`. The Code Completion window appears displaying a list of tags.
3. Select `<?php` from the list and press ENTER (or Double-Click). The php opening tag appears in the Editing window.
4. Press ENTER to begin a new line.
5. Type `pri`. The PHP Code Completion opens the display to show suitable code completion options.
6. Select the `print_r` function from the Code Completion window and press ENTER. `print_r()` appears on the edit line.
7. Type "hello" between the parentheses ().
8. Go to a new line and close the php tab by typing `?>`.

The Code should appear as follows

```
<?php
print_r(Hello);
?>
```

To view the output of the code in the Debug Output window, select Go  (F5) from the main toolbar.

Note:

Recommended coding practice in PHP is to add a semicolon (;) at the end of an expression as follows:
`print_r();`

Section 2: Debugging a Project

This section deals with controlling the Debugging process and accessing the information that Zend Studio provides.









For this procedure open the DebugDemo.php file. You can do this from the "Tip of the Day" dialog box at startup.

If the "Tip of the Day" containing the Debug Demo does not appear when you start Zend Studio, go to Help | Tip of the Day.

Note:

To enable the "Tip of the Day" so that it opens automatically at startup, go to Help | Tip of the Day and select the Open on Startup option at the bottom of the dialog box.

Quick Start Section 2

1. Start Zend Studio: from the "Tip of the Day" dialog. Click Debug Demo  to open the file DebugDemo.php in the Editing window.
2. On the main toolbar click Run . Output appears in the Debug Output Window.
3. Place the cursor in the Debug Output Window, right click, and select Show in Browser from the shortcut menu. The output is displayed in the internal browser.
4. Move the cursor to the Editing Window and press Go  to start the Debugger. The debugger icon  will appear while the Zend Debug Server runs, and remains onscreen until the Debugger detects a breakpoint at line 46.
5. Click (Step Over)  multiple times until the cursor arrives at line 51.
6. Place and hold the cursor over \$worker_name, \$worker_address, and \$worker_phone. A tooltip appears displaying the variable values.
7. Click (Step Into) . The Debugger advances to line 26.
8. In the Debug Window click the Stack tab, and then click the node to the right of row_color. The call stack tree expands displaying variable i.
9. Click (Step Out) . The cursor arrives at line 51.
10. Click Run . Output appears in the Output Window and a Notice appears in the Debug Messages Window.
11. In the Debug Messages Window, double-click on the Notice. The cursor jumps to line 61 in the Editing Window.

Section 3: Profiling a Project

Use the Zend Studio integrated Performance Profiler to optimize overall performance of your applications.

Zend Profiler detects bottlenecks in scripts by locating problematic sections of code. These are scripts that consume excessive loading-time. The Profiler provides you with detailed reports that are essential to optimizing the overall performance of your application.

The Zend Studio Profiler performs the following:

- Monitors calls to functions
- Monitors the number of times that a section of code is executed
- Calculates the total time spent on execution
- Generates reports that reflect the time spent on execution
- Graphically displays information of time division
- Enables comparison statistics between functions
- Enables viewing the file from the server just by clicking on any function
- Shows the hierarchical structure of the functions involved in the script execution

Note:

The Zend Debugger must be installed on the Server of the selected URL.

Quick Start Section 3:

1. From the Tools menu, select Profile URL. The Profiler automatically detects the application's URL yet enables you to type another URL for profiling, if needed.
2. Accept the default URL or change and click OK. The browser presents the requested page and after a few seconds, during which the Profiler accumulates information, the Profiler Information window appears.

Managing Projects

Zend Studio provides a wide variety of file management options from the application workspace.

Two tabs, the File System tab and the Project tab, contain file management tools that help to effectively manage files in the file system and project. Filter the file and project tabs to display specific file types or view All Files using the File Type drop down list.

With Projects users can:

- Collect several files under a single context and settings - Create New Project
- Define Properties for the Project - Set Project Properties
- Set unique project settings - Project Debugging and Encoding Settings

Create a New Project

You can create a New Project if you wish to define a working environment with unique characteristics such as Debug configurations, Bookmarks and Watches storage, etc.

Note:

Project definition files are assigned the *.zpj file extension.

To create a new project:

1. From the Main Menu select Project | New Project. The New Project Wizard dialog box appears.
2. Type the name of the new project. The location is updated accordingly. Click Next to define specific properties for the new project, or you may skip all the following dialogs by clicking Finish.
3. To add the files/directories that will comprise the new project, click Add Path and browse for the files/directories to be included in the new project.
4. Click Next to continue or Finish to skip.
5. The next window displays the default settings defined in the Debug tab in the Preferences window. If you wish to apply specific debug settings for the current new project, un-check the Use System Defaults check box and modify the settings.
6. Select the Debug Mode. For server debugging change the server URL and the port number, and specify the temporary output file location.
7. Click Finish.

Note:

These settings are reflected in the Project Properties dialog. To view a project's debug settings at any time, open the project and from the main menu, go to Project | Project Properties.

Set Project Properties

Zend Studio allows you to set or modify properties for a project. Configurable properties include:

- Encoding
- Debug Mode (Server or Internal)
- Debug Server URL
- Debug Port
- Temporary Output Files Location

To reach the Project Properties window, do the following:

1. Create a new project or open an existing project from File Manager's Project tab (or use the right-click menu).
2. Once the project is open in the editor, open the Project Properties dialog from the Project menu by selecting Project | Properties.
3. Define/Edit the properties for the current project.
4. Click OK.

Project Debugging and Encoding Settings

Debug Settings determine the type of debugging applied to the project files and the basic configurations that may vary from project to project (see Debugging and Analyzing Code on page, 84). Language encoding, determines the language and character set used for the code created in Zend Studio.

Debugging and Encoding settings can be determined on two levels, by project or a Zend Studio default setting.

To initially determine the default project settings go to: Tools | Preferences | Desktop tab. Select the appropriate encoding language, from the Encoding drop-down menu and define the debugging settings.

When creating a new project, users may decide if they want to use the default settings or define specific settings that will only be applied to the new project.

Existing project settings can also be modified and users may decide if they want to use the default settings or change the project's settings.

To change settings in the new Project Wizard and Project Properties:

- **To use default-encoding settings**
Select the option: Use System Defaults. This will apply the settings in Preferences to the new project.
- **To set different encoding for a specific project**
Make sure the Use System Defaults option is not selected and that the Debugging and Encoding menus are enabled. Modify the debugging and encoding settings and press OK to apply the settings to the project.

Note:

Adding, changing and applying, project settings, does not affect the default encoding setting in the preferences menu. Moreover, changing the settings in the Preferences menu only affects projects that are set to use system defaults.

Chapter 3 - File Management

IN THIS CHAPTER...

ACCESSING FILES
 FILTERING FILES
 FILTERING USER DEFINED FILE TYPES
 WORKING WITH REMOTE (SERVER) FILES
 ADDING AN FTP ROOT

With Zend Studio, users can easily view, access, and modify files on local and servers using state-of-the-art secure communication technologies.

- Local and remote files can be viewed and accessed through the File Manager.
- Files on remote servers can be accessed with three different types of file transfer protocols (FTP, Secure FTP, FTP over SSL).

Accessing Files

The file management system is essentially a tree structured file view. From the File Manager you can copy, cut, paste, move, rename, and delete files and folders. This includes the ability to drag and drop files and folders. Through the File Manager developers can access Local, Network, and FTP drives.

Working with Files

The following features provide advanced file accessibility and usability:

- Filtering Files
- User Defined Filters

Filtering Files

Zend Studio allows you to filter the files displayed in the File Manager by type. A drop-down list, containing the types of files currently defined for the system, is located at the bottom of the File Manager.

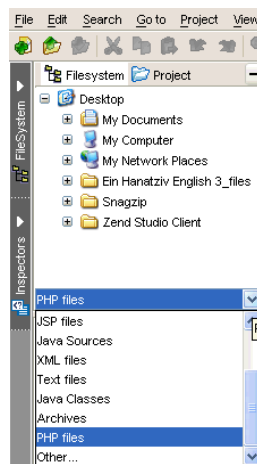


Figure: 6 - File Type Drop-Down Menu

There are several different options for filtering files by type:

- **Filter default file types** – Use the file manager's file type filter to display files based on the default subset of file types.
To see the list of known file types go to: Tools | Preferences | File Types.
- **Add user-defined file types** – define additional file types and extensions that are recognized by the file manager and therefore can be used to filter the current view.
To add additional file types and file extensions to the list of known file types and extensions, go to: Tools | Preferences | File Types and press add next to the file types or extensions list. Each file type can be associated with more than one extension.
Once a new file type/extension is defined files appearing in the file manager can be filtered accordingly.
- **Add user-defined file type groups** – take a selection of default and user defined file types and group them into a single category for filtering. These file type groups can group any type of file type and the file manager automatically recognizes them.
To group file types go to: Tools | Preferences | File Types and press add in the custom file type groups section.

To display files with a particular file extension:

1. Click on the file type field to open the File Type drop-down list.
2. Select the type of files you wish to view in the File Manager window. The File Manager automatically displays files of the selected type only.

Filtering with User Defined File Types

The File Manager allows you to filter for file types other than those listed in the drop-down list.

To filter for a file type that is not listed:

1. Open the file types drop-down menu.
2. Select "Other ..." from the list.
3. Overwrite "Other..." in the file type field with a wild card and the file extension for the type of file you wish to view, e.g., *.doc. The File Manager displays files of the type defined.

Note:

A file type defined by overwriting "Other..." in the file-type drop-down menu is not retained in memory. When you exit Zend Studio this definition is lost. To define a new file type that will be retained by the system use the Add New File Type routine from the Preferences window: Tools | Preferences | File Types.

Working with Remote (Server) Files

Zend Studio allows you to define paths to files that are not located on your local drive. You can add an FTP root to your File System window, and the application treats your FTP site as if it were another drive.

There are three types of connection methods:

1. **Regular FTP** - FTP is an application protocol that uses the TCP/IP protocols for file transfer over the network.
2. **Secure FTP** - SFTP, or secure FTP, is an FTP protocol that uses SSH to transfer files (not a standard FTP protocol). Files, commands, and data are encrypted before transfer, preventing passwords and sensitive information from being transmitted over the network.
3. **FTP over SSL (FTPs)** - FTPs (FTP over SSL) is a standard FTP over an SSL connection that offers all the features found in the FTP component, with the added ability to encrypt FTP data using SSL (Secure Sockets Layer).

Adding an FTP Root

You can add FTP Roots to the File System window. Adding an FTP Root allows you to use your FTP as if it were another drive.

1. To add/configure an FTP Root go to the main menu and select File | Add FTP Server or, from the File Manager's File System tab, open the right-click menu and select Add FTP Server.
2. In the Configure FTP Server dialog box select one of the connection types and configure the parameters.
3. Click OK. The new FTP Icon appears in the file system.
4. Once in the file system, the file manager will handle remote files the same as local files.

Note:

The FTP Root Properties can be edited by selecting an FTP Root, opening the right-click menu, and selecting Properties.

In order to add an FTP root, you will need to provide the following information:

Field	Description	Connection Type
Connection Name	Connection Name is an alias that identifies the server in the Zend Studio file system.	All
Connection Type	There are three connection options (Regular FTP, Secure FTP and FTP over SSL (FTPs). The fields in the configuration dialog will change according to the selected connection option.	
Host Name	The valid Host name or IP address for the FTP Server.	All
User Name	The account or user's login name.	All
Password	The password that corresponds to the user name.	All
Initial Directory	An FTP virtual directory where the subfolder templates will be placed.	All
Timeout (Sec)	The amount of time allowed for making a connection to the Server.	All
Remote Port	The FTP port on the server.	All
Passive Mode	Selected for Passive mode transfer, when required. When not selected the mode defaults to Active transfer.	Regular and FTP over SSL
Anonymous Login	Select when the FTP Directory allows using an Anonymous Login password to connect.	Regular and FTP over SSL
Save Password	Select when you would like the password retained in the setup. If not selected, the password must be entered at each time you log in to the FTP location.	All
Reconnect on Startup	Reconnects automatically to FTP site when Zend Studio is invoked.	All
Implicit SSL	The valid Host name or IP address for the SSH Host Server.	FTP over SSL
Explicit SSL	The SSH account or user's login name.	FTP over SSL

Note:

The FTP Root can be used to read and write to FTP servers using standard Windows methodology: copy-paste and drag-and-drop files.

Using FTP Over SSH2

File Transfer Protocol (FTP) provides secure file transfer functionality over any reliable data stream. It is the standard file transfer protocol for use with SSH.

To enable FTP for an SSH server:

1. Go to File | Add FTP Server. The Configure FTP Server dialog box opens.
2. Check the Secure FTP option.
3. In the Connection Name area, enter the Connection Name.
4. In the Login Information area, enter the Host Name, User Name, Password and Initial Directory.
5. Select the Options desired (if any).
6. Click OK. FTP will be enabled over the SSH connection you have defined.

Chapter 4 - Editing

IN THIS CHAPTER...

EDITING
 CODE COMPLETION
 PHPDOC SUPPORT
 @VAR TAGS
 TEMPLATES
 CODE INDENTING
 COMMENTING LINES/BLOCKS
 HTML TAG INSERTION
 USABILITY SHORTCUTS
 SYNTAX HIGHLIGHTING
 CLONE VIEW
 CODE SNIPPETS
 PRINTING

The Editing window is the main active window for composing code for a PHP application. It features a suite of tools designed to help programmers streamline the development process.

Using Code Completion

The Code Completion feature enables you to write code faster. It provides you with easy access to PHP Classes, Functions, Variables, Constants, Keywords, HTML tags, attributes, attribute values and more.

Two Code Completion types are available: PHP and HTML.

Location	Command	Results
HTML	CTRL+SPACE	Popup of the HTML Code Completion.
PHP	CTRL+SPACE	Popup of the PHP Code Completion.
	CTRL+SHIFT+SPACE	Popup of the Function Parameters list.

PHP Code Completion includes all PHP Classes, Interfaces, Functions, Constants and Keywords, as well as user defined Classes, Functions, and Constants. User declarations can be declared in the project files as well as in the open files. PHP Code Completion lists names and syntax.

PHP Code Completion includes:

- Code Completion for nested functions
- Code Completion for function return values (using PHPDoc support)

HTML Code Completion includes:

HTML tags, attributes, and attribute values. This Code Completion lists names and syntax.

Framework Integration

Framework Integration integrates Zend Framework's APIs into Studio 5.5's *Code Completion* features. Framework is based on the MVS (Model-View-Controller) pattern and enables you to create Web Services and Web sites easily.

Framework requires PHP version 5.14 and higher. See the website (<http://framework.zend.com/faq/installation>) for current requirements.

1. Open Studio's Preferences (click **Tools | Preferences**).
2. Select the **Desktop** tab.
3. Mark the checkbox named **"Include Zend Framework Resources"**.
4. Click **Apply/OK** to exit and return to the IDE.
5. Framework's resources are now included in the Code Completion library.

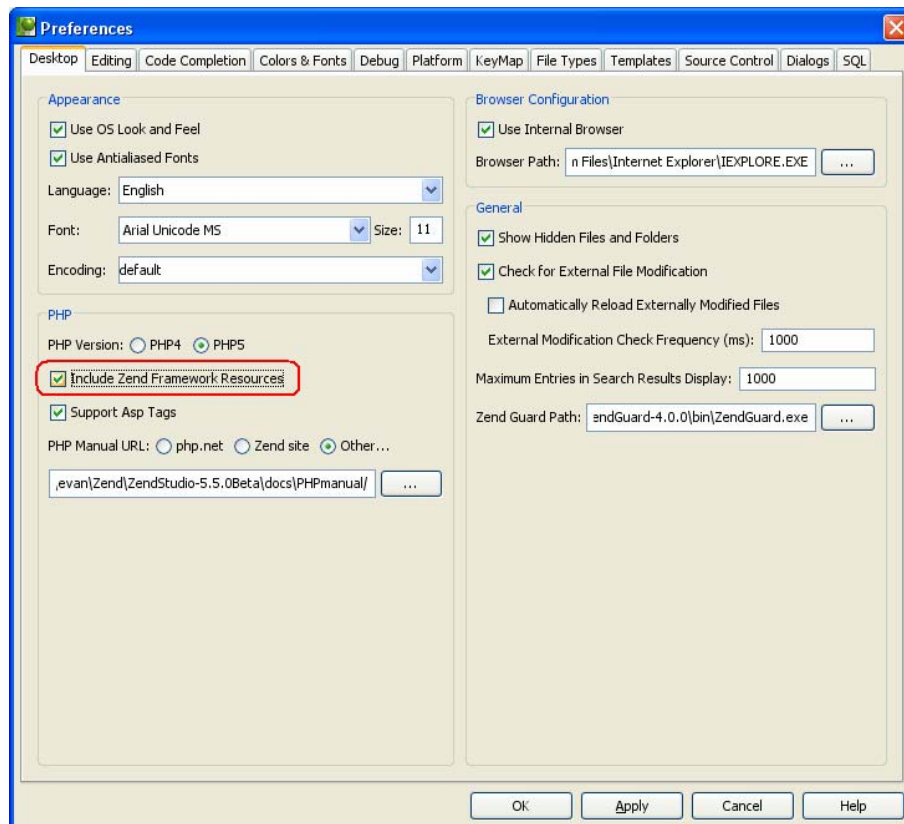


Figure: 7 - Desktop Preferences

Zend Framework Components

Zend_Controller	A module to provide overall control for the application. It translates requests into specific actions and makes sure they get executed.
Zend_Db	Based on PHP Data Objects (PDO); it provides access to databases in a generic way.
Zend_Feed	Makes it easy to consume RSS and Atom feeds.
Zend_Filter	Provides string-filtering functions, such as <code>isEmail()</code> and <code>getAlpha()</code> .
Zend_InputFilter	To <code>Zend_Filter</code> , works with arrays such as form inputs.
Zend_HttpClient	Enables you perform HTTP requests easily.
Zend_Json	Enables you to easily translate PHP objects into JavaScript Object Notation, and vice-versa.
Zend_Log	Provides general-purpose logging functionality.
Zend_Mail	Enables you to send text and multipart MIME e-mail.
Zend_Mime	Helps decode MIME messages; it is used by <code>Zend_Mail</code> .
Zend_Pdf	Enables you to create new PDF documents and load and edit existing PDF documents.
Zend_Search	Enables you to perform sophisticated searches on your own text. For example, you can build a search engine that returns results based on relevancy or other factors.
Zend_Service_Amazon, Zend_Service_Flickr, & Zend_Service_Yahoo	Provide easy access to these Web service APIs.
Zend_View	Handles the "view" portion of the MVC pattern.
Zend_XmlRpc	Enables you to easily create an XML-RPC client. (Server capabilities are planned for the future.)

Referencing Files / URLs

Studio Editor enables opening a **referenced file or URL** that is contained inside a PHP script. When a script in File #1 contains a reference to another file or to a URL, that file/URL can be opened directly.

Files open in the Editor; URLs open in the selected browser.

1. Hover the cursor over the **file** reference. The **Open** option appears in the drop down menu.

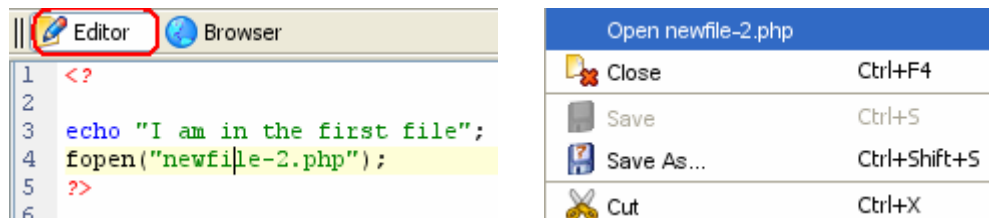


Figure: 8 - Open <File>

2. Select "Open File" or "Open URL". The file will open in the **Editor** (see figure on the *left*, below).
3. Hover the cursor over the **URL** reference and right click. The **Open** option appears in the drop down menu.

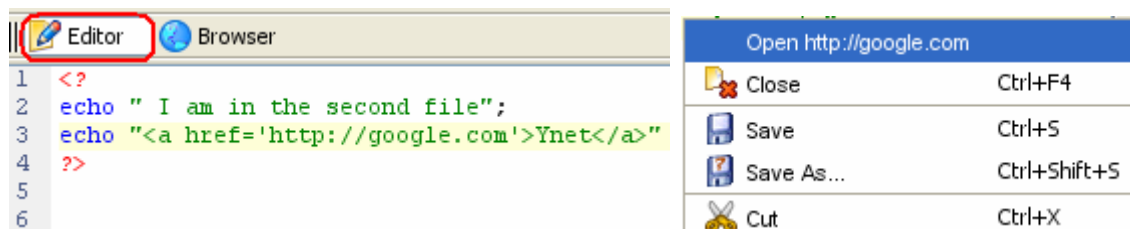


Figure: 9 - Open <URL>

4. Select “**Open ..**”. The URL will open in the **Browser**.

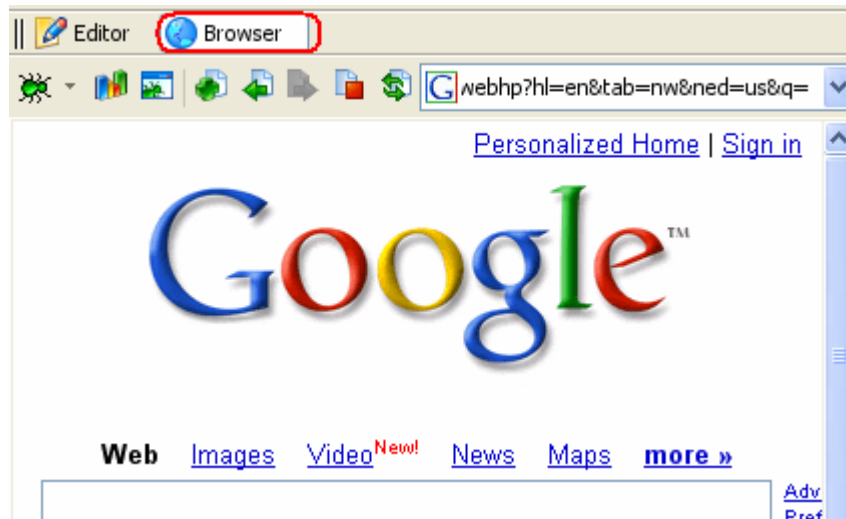


Figure: 10 - Open <URL>

@var tag as Class Type Hint

By using a comment you can assign a variable its exact class value. This assignment will affect the code completion of this variable accordingly.

In the example below, '\$myVar->' opens the code completion with 'Test' class function as defined in the comment.

```
function getClass() {
    return new Test();
}
class Test {
    function printValues($a, $b) {
        echo "Values: $a, $b";
    }
}
$myVar = getClass();
/* @var $myVar Test */
$myVar->
```

Code completion dropdown showing: `Test::printValues($a, $b) void`

Note:

Without the comment, code completion will not be available for the function.

Using Templates

Templates help beginner and advanced developers write code quickly and accurately. Templates are shortcuts used to insert a framework for the segment of code you are about to write.

Once a template is inserted, developers can compose code using a combination of manual and automated code entry methods.

Templates are subject to the following conditions:

- A functional template must be defined in the Templates list in order for the shortcut to work (see Preferences | Templates).
- Templates work for one context only: HTML, PHP, PHP DOC, JavaScript or CSS. This means that the section of code you are working on—depending on its context—will define which templates work and which do not.

To insert a template:

1. Place your cursor at the desired insertion point.
2. Type a combination of keys. A completion list opens (PHP only) listing all available templates and completion options that begin with that combination of keys.

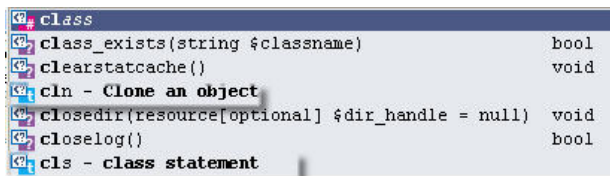



Figure: 11 - Templates in the Code Completion Menu.

3. Templates are marked in the code completion list with a blue square , select a template from the list, or press Tab.
4. You can now navigate from variable to variable within the template framework and complete the code, using the Tab and Shift-Tab keys.

To define a template:

1. Go to Preferences | Templates and press Add. The Add a new template dialog opens.
2. Enter the template's details as follows:
 - a. **Abbreviation** – a short name that is given to each template (by the user) for identifying and differentiating between templates (e.g. a template for a while loop may be called while).
 - b. **Context** – The context in which the template will be functional (PHP, PHPDoc, HTML, JavaScript or CSS).
 - c. **Description** – A short description of the code in the template.
 - d. **Template Code** – The actual code that will be inserted in the editor when selecting the template.

- e. **Add Var** – Enables adding a variable to the template from a list of commonly used variables.
3. Click OK to confirm the details of the new template or Cancel to delete and not save.

Indenting Code

PHP indentation applies conventional indentation to your PHP code, including tabs, spaces, and matching-up brackets.

Maintain PHP indentation in applications in several ways:

- Manually
- Apply indentation "as you type" using Auto Indentation
- Apply indentation all at once

Zend Studio supports automated PHP indentation. This helps you save time in adding and correcting indentation. This feature is dependent on another user-customizable feature called – “Tab Size” (Tools | Preferences | Code Style). Indenting code will perform according to the tab size settings.

Code indentation settings are defined in the preferences menu (Tools | Preferences | Editing). See “Setting Code Completion Preferences”.

Commenting Lines Blocks

Zend Studio allows you to select a location or block of text and tag it as a comment.

There are two types of comment code that can be added or removed:

- Line Comment
- Block Comment

The behavior of this feature changes according to the selected area of code. PHP sections and HTML sections are treated differently. The comment added depends on the context (i.e., PHP or HTML).

To add a line comment:

Go to Edit | Add/Remove Line Comment (or Ctrl+Slash).

To add a line comment:

Go to Edit | Add/Remove Block Comment (or Ctrl+Shift+Slash).

For PHP Sections:

The Line Comment option inserts the double forward-slash characters (//) to the line or lines selected as comments:

```
// Comment Text Line 1
// Comment Text Line 2
// Comment Text Line 3
```

The Block Comment option inserts the beginning (/*) and ending (*/) coding in order to mark the selected line or lines as comments.

```
/* Comment Text Line 1
Comment Text Line 2
Comment Text Line 3 */
```

For HTML Sections:

Line Comment and Block Comment options behave the same, inserting beginning (<!--) and ending (-->) tags to mark the selected line or lines as comments.

```
<!-- Comment Text Line 1
Comment Text Line 2
Comment Text Line 3 -->
```

Inserting HTML Tags

Zend Studio allows you to add HTML tags either manually or from a menu of choices.

To add an HTML tag to a line of code:

1. Select the location or text on which to place the HTML tag.
2. From the main menu, select Edit | HTML Tags. The menu of currently defined HTML tags opens.
3. Select the HTML tag you wish to add to the line of code. The HTML tag is added to the line of code.

You can add the following HTML tags to a selected line or lines of code in the Editing Window:

Name	Tags
Break	
Space	
Bold	
Italic	<I></I>
Heading 1	<H1></H1>
Heading 2	<H2></H2>
Heading 3	<H3></H3>

Tip:

Use the shortcut keys to insert HTML tags rapidly. The default shortcuts are shown to the right of the menu option.

Usability Shortcuts/Timesavers

Zend Studio supports a number of timesaving features to streamline the editing process. A short description of each follows.

- **To maximize the Editing window:**
 - When the Editing window and the secondary windows are all open, press Esc. To "restore down" the Editing window and open all secondary windows.
 - When the Editing window is maximized and the secondary windows are all closed, press Shift + Esc.
- **To Convert to All Uppercase:**
In the Editing window, select the text to convert and press CTRL+U.
- **To Convert to All Lowercase:**
In the Editing window, select the text to convert and press CTRL+L.
- **To duplicate a line:**
In the Editing window, place the cursor on the line and press CTRL+D.
- **To erase a line:**
In the Editing window, place the cursor on the line and press CTRL+E.
- **To increase multiple line indentation:**
In the Editing window, select as many lines of code as you like. Then press Tab. The entire segment will be indented by a tab (To remove the indent, select the lines and press Shift Tab).

Note:

The shortcut keys for these conversions can be changed in the KeyMap Tab (Tools | Preferences | KeyMap).

Matching Highlighted Elements

The enclosing brackets, braces, parenthesis, greater-than and less-than characters, single quote-marks, and double quote-marks are highlighted in pairs within the Editing window. The highlighting indicates the start- and end-points of the section. It is intended to help you check for errors and missing braces/quotes.

Highlighting behaves as follows:

- Highlighting appears when the cursor is placed immediately before the beginning enclosing-character or directly after the end enclosing-character.
- If the enclosing-characters are mismatched, each enclosing-character will be highlighted in a different color.

Customize the highlighting color for:

- **Matched Brackets**
Affects highlighting on matching () { } [] < > characters
- **Mismatched Brackets**
Affects highlighting on mismatched " " () { } [] < > characters
- **Matched Quotes**
Affects highlighting on " " ' ' characters

Disable and Enable Highlighting:

Select or Deselect the option in the Tools | Preferences | Editing window.

Jump Between Matching Brackets:

Jump from the one bracket to the other, to reduce the need for scrolling.

To do this:

1. Open the Goto main menu and select Goto Matching Bracket.
2. The Editor will advance to the matching bracket.

Clone View

The Clone window is a duplicate Editing window of an open file. In the Clone window-as in the original window-you can view, cut, copy, paste, and search text. This feature allows you to move blocks of code or to view two areas of code at once, thereby eliminating excessive scrolling.

The Clone window can be a floating window or it can be docked in any of the docking areas of the Zend Studio. Furthermore, you can open multiple Clone windows for any of the open documents. This allows you to view several different files at once.

Note:

The clone-view is not a duplicate file. Rather, a single file shared by multiple windows.

To Duplicate the Editing Window (Clone View)

Open the Clone window from the Editing window by selecting “Clone View” from the right-click menu.

Anti-Aliasing Support

Anti-aliasing is a software technique that reduces jagged, stairstep-like lines (that should be smooth). These (i.e., *Jaggies*) occur when the resolution of the output device is not high enough to properly represent a smooth line.

Anti-aliasing reduces the prominence of jaggies by surrounding the jagged “steps” with intermediate shades of gray or color (device dependant). This reduces the jagged appearance of the lines but simultaneously makes them fuzzier.

1. Open the Preferences dialog (click **Tools | Preferences**).
2. Select the **Desktop** tab.
3. Check the **Use Antialiased Fonts** checkbox.

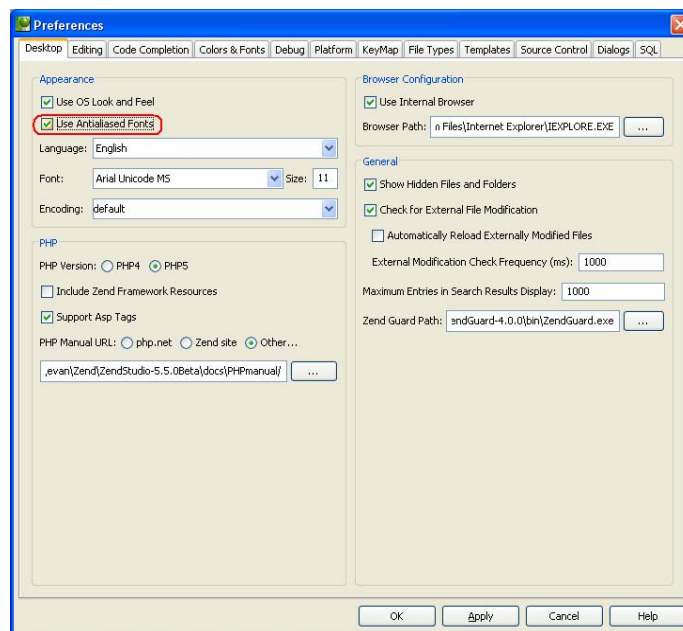


Figure: 12 - Anti-Aliasing

4. A **Restart** message will appear followed by a **Save** message.

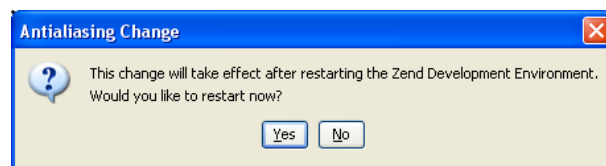


Figure: 13 - Restart Studio

5. Save your work and Restart Studio. After it restarts, all GUI text elements (editor, menus, etc.) will be anti-aliased.

Code Snippets

Zend Studio supports "Code Snippets." Code Snippets are reusable pieces of code that can be accessed to facilitate the application development process.

The feature set allows you to:

- View and use existing Code Snippets
- Create and update your own Code Snippets
- Update the database from Zend's online Code Gallery

To insert and view Code Snippets:

1. From the Edit menu, select Show Snippets. Alternatively, right-click anywhere in the main Editor workspace. Then select Show Snippets from the right-click menu. The Code Snippets window opens.
2. From the Code Snippets tree to the left of the Code Snippets window, click the snippet you wish to view. The snippet description and code will display in the Preview pane of the Code Snippets window.
3. On the Code Snippets window, click Insert. Zend Studio pastes the code snippet into the code at the caret position.

Note:

The tree of code snippets consists of user defined snippets and snippets from Zend's Code Gallery. Only snippets that appear in the user defined section of the tree are fully editable. To edit a snippet from the Code Gallery, you must first save it (Save As), which creates an editable copy of the snippet under user defined.

Creating a Code Snippet

To create a new code snippet:

1. From the Edit menu, select Create New Snippet. Alternatively, right-click anywhere in the main Editor workspace. Then select Create New Snippet from the right-click menu. The Import Content prompt appears asking you if you wish to import the contents of the file to the new snippet.
2. Choose an import code import option:
 - a. **Import Selection** - Imports the selected lines from the Editor workspace into the Add a New Snippet window. (Note: this option will appear only if there is a selection.)
 - b. **Import File** - Imports the entire file from the Editor workspace into the Add a New Snippet window.
 - c. **None** - Opens the Add a New Snippet window with a blank Snippet Code workspace.
3. In the Add a New Snippet window, enter the code you wish to store as a Code Snippet.
4. Enter a name and description in the fields provided.

5. Click OK.

Zend Studio will add a new Code Snippet to the user-defined section of the tree under a user assigned name.

Editing a Code Snippet

Only user defined snippets are fully editable.

To edit a snippet from the Code Gallery, first save it (Save As) to create an editable copy of the snippet (under user defined).

To edit a Code Snippet:

1. From the Edit menu, select Show Snippets. Alternatively, right-click anywhere in the main Editor workspace. Then select Show Snippets from the right-click menu. The Code Snippets window opens.
2. From the Code Snippets tree to the left of the Code Snippets window, click the snippet you wish to edit. The code snippet appears in the Preview pane of the Code Snippets window.
3. To edit the code snippet, click Edit. The Edit Snippet window opens with editable code displayed in the Snippet Code workspace.
4. Edit the code as you wish.
5. Click OK.
Zend Studio overwrites the Code Snippet with the edited version.

Note:

Zend Studio allows you to remove a user defined Code Snippet simply by selecting the code snippet from the list and clicking Remove.

Updating the Code Snippet Database

Zend Studio's Code Snippets database consists of User Defined snippets and snippets imported from Zend's Code Gallery.

The update policy for keeping the database current is as follows:

- On first-time use of the Code Snippets functionality, Studio prompts you to update the Code Snippets database - i.e., load the contents of Zend's Code Gallery.
- If 30 days has passed since your last update, Studio prompts you to update the Code Snippets database.
- The Code Snippets window allows you to manually initiate an update at any time by clicking Update Snippets.

Printing

Zend Studio supports a range of printing options. These options are configurable from the Print dialog (File | Print)

Print options are as follows:

Print Option	Description
Print Selected Lines	Prints the selected lines only. This option is disabled until lines are selected.
Print File Name	Active only when a file name exists. For example, if you wish to print the contents of Debug Messages, [Unknown] will appear in the File Name field and the Print File Name option will be grayed out. Once you have given the file a name, the Print File Name option becomes active.
Color Printing	Prints in color.
Style Printing	Prints formatting such as bold or underlined code. Choosing Color Printing automatically selects Style Printing. This option can be manually selected for black and white printing.
Draw Border	Prints a rectangle (border) around the print area.
Print Page Numbers	Prints page numbers.
Print Line Numbers	Prints line numbers.
Orientation	Defines the orientation of the page: portrait or landscape. Note: The print dialog page orientation settings override the OS's page setup settings.
Wrapping	Wraps or doesn't wrap the printed text.

Note:

You can open the Page Setup dialog native to the Operating System by clicking the Page Setup option at the bottom of the Print dialog.

Printing an Active File

To set options for a print task:

1. Click in the workspace whose contents you wish to print.
2. Select the lines you wish to print. If you make no selection Studio will print the entire contents of the workspace.
3. Select Print from the File menu or from the Right-click menu. The Print dialog opens.
4. Select the print settings you wish to apply to the contents of the selected workspace.
5. Click Print.
Studio opens the native print dialog of the Operating System.
6. Click OK.
Studio executes the print task using the settings you have entered.

Chapter 5 - Code Navigation

IN THIS CHAPTER...

FINDING MATCHING BRACKETS
BOOKMARKS
FORWARD/BACKWARD NAVIGATION
GOTO SOURCE
GOTO PHP RESOURCE
GOTO PROJECT FILE
RECENT FILES DIALOG

Zend Studio provides a number of navigation tools to help you advance quickly to critical lines in the application code. These tools save time in the development process.

Navigation tools include:

- Bookmarks
- Matching Bracket Navigation
- Forward/Backward Navigation
- Goto options (Goto main menu option)
- Recent Files Dialog

Bookmarks (set, demote, go)

Use bookmarks in the active document to mark lines of code. This feature allows you to navigate quickly throughout your scripts. Studio allows you to attach a description to each bookmark, which can be viewed later as a tool tip next to the bookmark.

To add a bookmark:

1. Go to the Editor and select the line of code to bookmark.
2. Right-Click on the Line Number

Bookmarks appear as Green highlighting on the left-margin line numbers. Placing the cursor on a bookmark opens the description, if a description exists.

Bookmarks can be edited from the Edit Menu with the following Options:

- **Add/Remove Bookmark** – Select a line in the code and use this option to add a bookmark.
- **Remove all Bookmarks** – deletes all bookmarks from the editor.
- **Show Bookmarks Dialog** – Opens the Bookmark management dialog that includes the following options:
 - **Goto** – Closes the dialog and navigates to the line in the code.
 - **View Source** - navigate to the line in the code.
 - **Remove** – Select a bookmark and press Remove to delete a single bookmark
 - **Remove All** – Press to delete all bookmarks from the bookmarks Dialog

Once you have added a bookmark, you can advance from the current cursor position to the next bookmark.

There are two options for advancing to a bookmark:

1. **Go to Next Bookmark** - Moves the cursor to the next bookmark in the active file.
2. **Go to Next Project Bookmark** - Moves the cursor to the next bookmark in the entire project and opens the file in the Editor.

Bookmark Manager

The Bookmarks Manager lists all the existing bookmarks and the description attached to each bookmark. In addition, the Bookmarks Manager enables you to do the following:

- **Add Description** - Click the description line next to the Bookmark you wish to describe and type your note.
- **Goto** - Closes the Bookmarks Manager window and displays the selected bookmark in the editing window.
- **View Source** - Leaves the Bookmarks Manager window open and displays the selected bookmark in the editing window.
- **Remove** - Removes the selected bookmark from both the file and the Bookmarks Manager window.
- **Remove All** - Removes all bookmarks from the files and the Bookmarks Manager window.
- **Close** - Closes the Bookmarks Manager window.

Finding Matching Brackets

The enclosing brackets, braces, parentheses, greater-than and less-than characters, single quote-marks, and double quote-marks within the editing window are highlighted in pairs.

Highlighting behaves as follows:

- Highlighting appears when the cursor is placed before the beginning enclosing-character or directly after the end enclosing-character.
- If the enclosing-characters are mismatched, the highlighting will appear with a different highlight color. Highlighting allows you to see the start and ending of these sections. Use this feature to check for errors and missing braces.

Customize highlighting for:

- **Matched brackets** - Affects highlighting on matching () { } [] < > characters
- **Mismatched brackets** - Affects highlighting on mismatched " " () { } [] < > characters
- **Matched quotes** - Affects highlighting on " ' ' ' characters

Highlighting can be enabled and disabled from Preferences: Tools | Preferences | Editing.

The Matched Brackets feature allows you to jump from the one bracket to the other, which reduces the need for scrolling.

To do this, select the main menu option: "Go to Matching Bracket." This option causes the Editing window to advance to the matching bracket.

Forward Backward Navigation

Moving to a specific line of code can be quite a task if you have to manually scroll many lines of code.

Zend Studio provides a wide selection of Goto options:

- **Goto menu options:**
 - **Goto File** - Navigates to open or Project files from the Go to File Dialog.
 - **Goto Resource** - Navigates to PHP resources in an open file or project from the Go to PHP Resource Dialog.
 - **Goto Line** - Jumps to a line in the active file.
 - **Goto Matching Bracket** - Jumps between beginning and ending braces.
 - **Goto Next Bookmark** - Jumps to the next bookmark the active file.
 - **Goto Next Project Bookmark** - Jumps to the next bookmark in another open file.
 - **Back/Forward** - Jumps back and forth along navigated items.
 - **Open Next Message Entry** - Jump to next line in the Find In files search result (displayed in the Debug Messages window).
- **Embedded³ Goto Options:**
 - **Goto Cursor-Line Box** - Jumps to a line and character position (accessed from the Debug menu).
 - **Goto Class or Class Member Declaration** - Jumps to the function declaration in the project, within all project files or only in the active document. (Use CTRL/Command⁴ +hovering over the area or select the option from the right-click menu).
 - **Goto Source of Debug Message** - Jumps to the source that caused the Warning, Notice, or Error (double click on a debug message to jump to the specific line in the editor).
 - **Go to Include File** - (active file only) Jumps to the include file. (Use CTRL/Command +hovering over the area or select the option from the right-click menu.)

Smart Goto Source

Zend Studio's Smart Goto Source enables PHP code developers to instantly jump to a function's declaration source code. Developers will find this feature helpful in understanding the code flow and as an easy way of navigating through the code.

³ Embedded options are options that are provided in addition to the main Goto menu option and are embedded in other Zend Studio components such as the Debugger Messages window.

⁴ Use the Command key for MAC instead of CTRL.

To use Smart Goto Source:

1. Hold down CTRL/Meta⁵ to activate the Smart Goto Source functionality. Hover with the mouse over the PHP element whose declaration statement you wish to view.
The element in the "user code" becomes an active link to its function declaration statement as you hover over it.
2. Left click with the mouse. The Editor's code display jumps to the selected element declaration statement.

Smart Goto Source supports navigation to the following code elements: classes, class elements (constants, variables and static variables, regular and static methods), functions, constants (via 'declare' keyword), included files, and more.

Zend Studio also supports Smart Goto Source for class variables whose type is hinted in '@var' tag in the phpDoc comment.

Goto PHP Resource

Use Goto Resource to navigate to PHP resources in an open file or project using the Goto PHP Resource Dialog.

To open the Go to Resource dialog:

1. From the main menu select Goto | Goto Resource.
2. Enter the resource name and choose the applicable resource.
3. Select a resource and click OK to navigate to the resource in the open file or project. Alternatively, double-click to navigate to a resource in the open file or project.

The Go to Resource dialog has an auto completion search capability that alphabetically lists resource names according to the letters typed in the Enter Resource Name field.

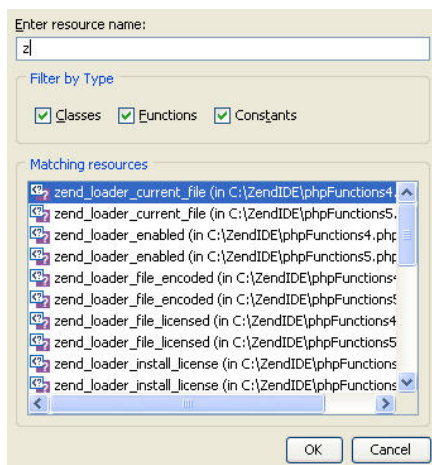


Figure: 14 - Goto PHP Resource

⁵ Use the META Key for MAC instead of CTRL.

The Filter by Type option provides a way to reduce or increase the amount of listed options by type. This is useful when searching for a single type of resource for example a class. Removing all the other options (functions and constants) reduces the list of applicable resources listed in the dialog.

Recent Files

Studio keeps track of recently opened files.

Re-open a recently used file by going to the File menu and selecting Recent Files. In this submenu you will be given the option to open one of the last used files.

The Recent Files window, which contains a much longer list of recently used files, can also be opened from the File menu.

Goto Project File

Use Go to PHP File to navigate to open or Project files from the Go to File Dialog

1. To open the Goto File dialog, select Goto | Goto File from the main menu.
2. Enter the file name and choose the applicable file.
3. Select a file and click OK to navigate to the file.

The Go to File dialog has an auto completion search capability that alphabetically lists file names according to the letters typed in the Enter file name field.

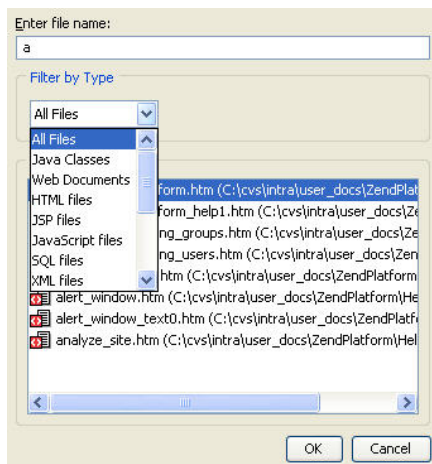


Figure: 15 - Goto Project File

The Filter by Type option provides a way to reduce or increase the amount of files listed by type. This is useful when searching for a single type of file, such as a PHP File. Removing all the other file types (HTML, XML, JavaScript, etc.) reduces the list of applicable file names listed in the dialog.

Chapter 6 - Web Services

IN THIS CHAPTER...

SOAP CLIENT

INCORPORATING WSDL FILES

GENERATING WSDL FILES

WSDL (Web Services Description Language) is an XML-formatted language used to describe Web service capabilities.

Web services are a standardized way of allowing applications to interface and share data across the network. Web service messages are written in XML, thus allowing for different applications in different programming languages to interface with each other.

WSDL files define how the Web services work and the operations they perform.

Zend Studio provides an integrated means for incorporating and inspecting WSDL files and a wizard for generating your own WSDL files (Tools | WSDL Generator).

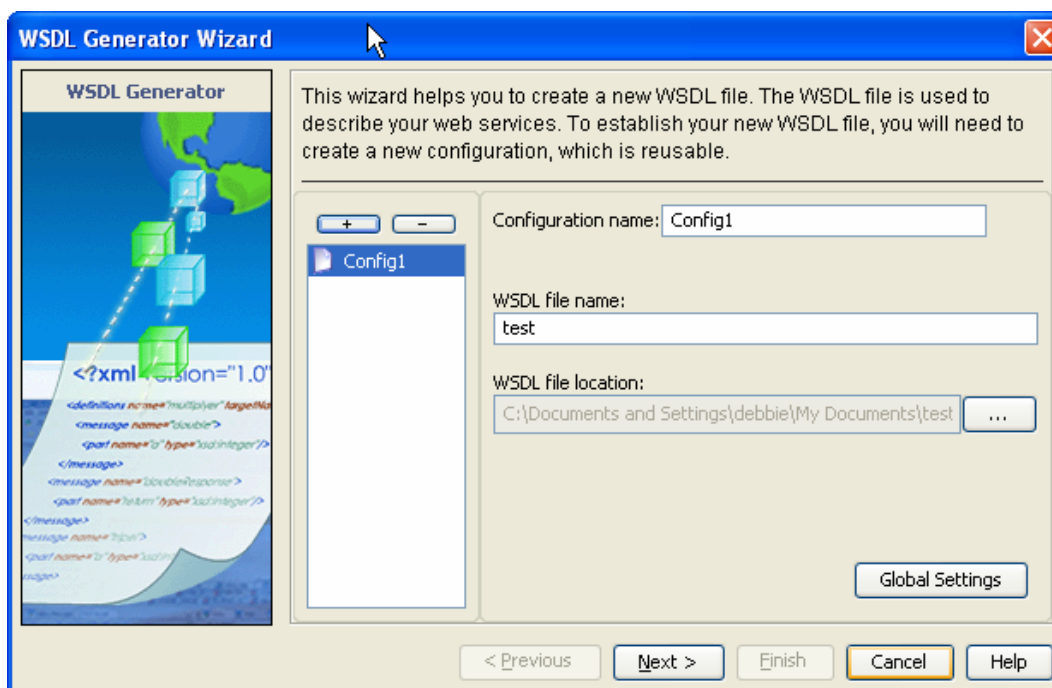


Figure: 16 - WSDL File Generator Wizard

SOAP Client

Studio 5.5 supports instantiation of Soap Clients containing URLs and/or files in their constructors. This is done using an URL or by referencing a local file. Both methods are functionally identical and completely transparent to the user.

After the client has been instantiated, an action — Refresh — is added to the Soap Client context menu (i.e., right-click) in the editor and in the Inspections view.

Creating a Soap Client

Method 1

1. Add the following line of code to your PHP file:

```
$a = new SoapClient("http://api.google.com/GoogleSearch.wsdl");
```
2. Save the file/project.
3. Code completion now includes all of the SOAP Client's methods, classes, functions, etc.).
4. The Client now appears in the Inspection pane.

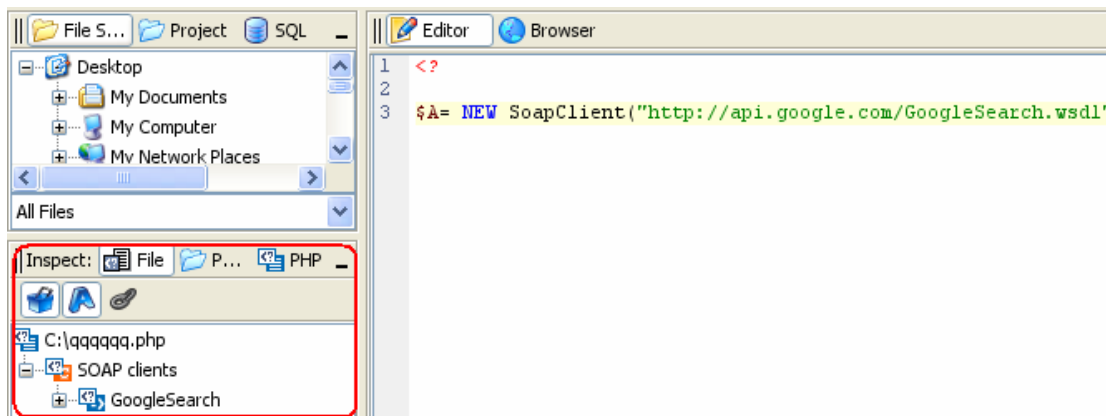


Figure: 17 - SOAP Client

5. Hover the mouse over the URL (the internal as well as the external browser can be used) and right click. **The following additional action now appears in the menu:**
 - Refresh WSDL File

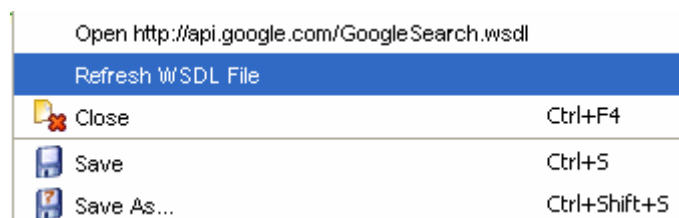


Figure: 18 - Additional Actions Added to Menu

Acquire a WSDL File

1. Use an external browser to open the URL referred to above.
2. Save the contents of the URL as a file; name the file "C:\GoogleSearch.wsdl".

Method 2

This is a feature of Studio version 5.2.

1. Create the soap client by referencing the WSDL file. Do this by adding the following line of code to your PHP file:


```
$a = new SoapClient("C:\GoogleSearch.wsdl");
```
2. Code completion now includes all of the SOAP Client's methods, classes, functions, etc.). Hover the mouse over the file path (the internal as well as the external browser can be used) and right click. **The following action now appears in the menu:**

Refresh WSDL File

Incorporating WSDL Files

Incorporating a WSDL file is the process of taking a service or services from an existing WSDL file and integrating their capabilities (functions) into the PHP code.

To reference a WSDL file and benefit from full integration into Zend Studio (code completion and function display in the Inspectors tab) the file must be present in your local file system.

External Web Service files - WSDL, that are referenced by a full path can also be referenced in the code by specifying the URL.

To reference a WSDL file in the PHP code:

1. Create a new Soap Client instance
2. Reference the path to the WSDL file as a parameter.
For Example:

```
<?
```

```
$client = new SoapClient("GoogleSearch.wsdl");
```

Once a WSDL file is properly referenced – and in this example above, the file named "GoogleSearch.wsdl", Zend Studio's behavior changes to offer the following advanced features:

- **Inspector**

The inspector view automatically fills-up with all the functions included in the referenced WSDL file.

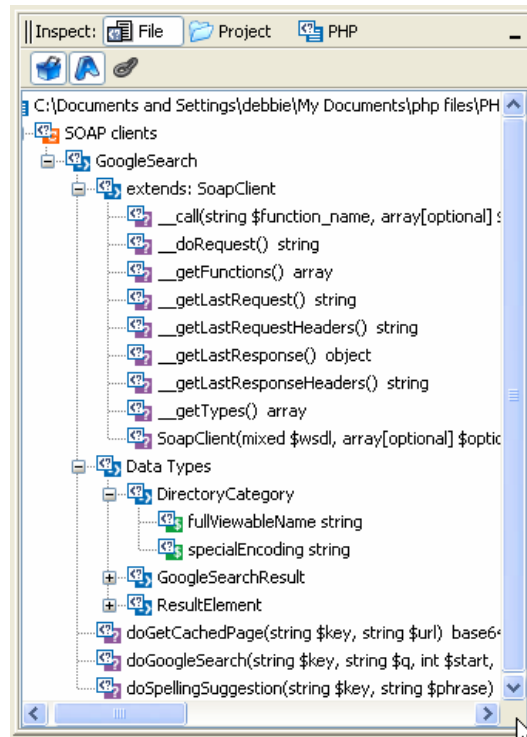
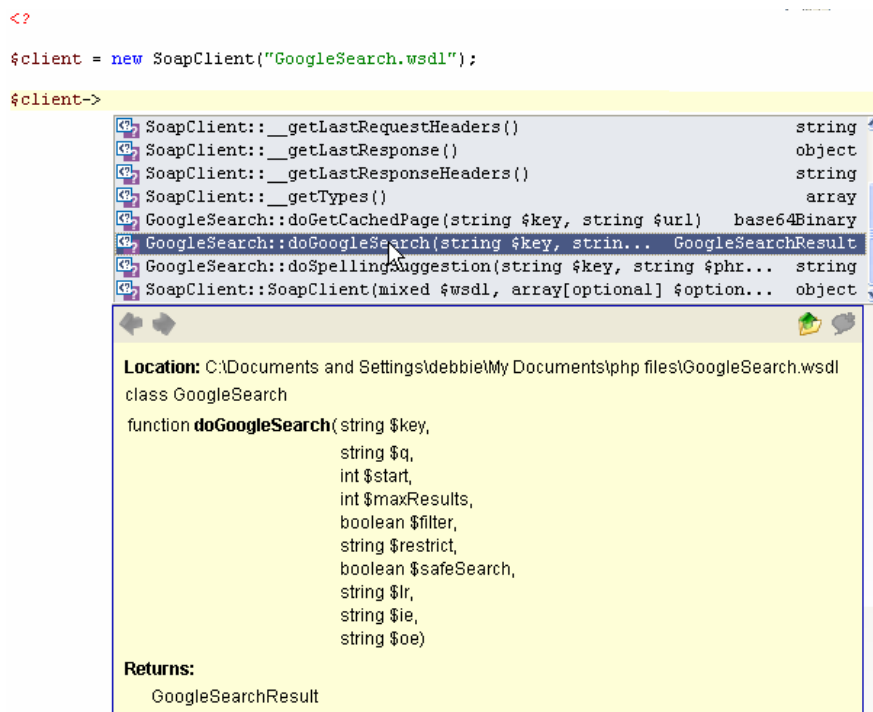


Figure: 19 - Inspector View

- **Code Completion**

Code completion is automatically updated with all the functions included in the referenced WSDL file. (Type the variable and press Ctrl + Space).



- **Auto Link to WSDL files**

Transform the name of the referenced WSDL file into a link by hovering over the file's name and pressing CTRL. Clicking the link (while CTRL is still pressed) will jump to the WSDL file if it is already open in the editor. Auto Link to Files exists for every string containing a file name in the editor.

PHP version compatibility note:

Code completion for Web Services is supported for PHP 5.

Generating WSDL Files - WSDL Generator

Generating a WSDL File is the process of selecting classes and non-class functions to be included in a WSDL file.

The WSDL Generator creates WSDL files based on user defined Configuration sets that are created in the wizard. This means that the wizard remembers configuration settings for each WSDL file.

Note:

Generated WSDL files contain the public functions of a class only.

To generate a WSDL file after creating a configuration set:

1. Open the Wizard by selecting Tools | WSDL Generator.
2. Select a configuration set from the list.
3. Click Finish.

Creating a Configuration Set

To create a configuration set for a WSDL file:

1. Open the Wizard by selecting Tools | WSDL Generator.
2. The Configuration List is a list of different configuration options that can be applied to generate a WSDL file. Use [+] to add configuration sets to the list and [-] to permanently remove configuration sets from the list.
3. Configuration Options:
 - a. **Configuration Name** - The unique name given to a WSDL configuration set.
 - b. **WSDL File Name** - The name that will be given to a WSDL file once it is generated.
 - c. **WSDL File Location** - The location of the WSDL file once it is generated. If you are regenerating an existing WSDL file, use the browse button to navigate to the WSDL file's location. The old WSDL file will be overwritten with the new file.
 - d. **Global Settings** - Additional advanced settings for creating and transferring information:
 - **Naming Convention**: defining a Namespace provides a method of avoiding element name conflicts.
 - **Binding Options**: Choose the format for transferring information. The binding style can be RPC oriented or Document oriented.
 - **Encoding**: If you choose to use encoding, make sure you specify the URL of the proper encoding file to determine the encoding type applied to the WSDL file.

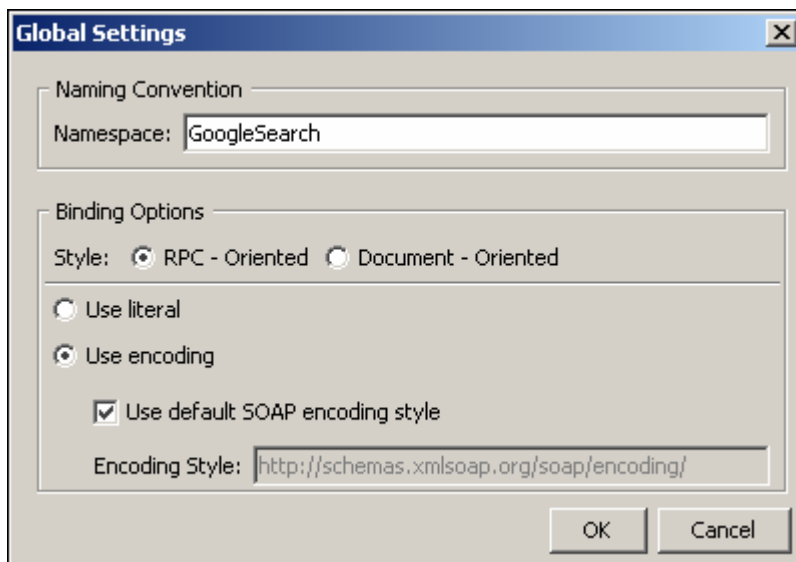


Figure: 20 - The WSDL Wizard, Global Settings Dialog

4. Once all these settings are configured, click Next to add classes and non-class functions. The add file dialog provides a way to apply content to the WSDL file. The wizard individually scans each file added to the list and displays its classes and non-class functions.
5. Add files to the list (the files should be either project files or files open in the editor).
6. The wizard scans the attached file/s to locate all the classes that should be mapped to a unique port for each class. Specify a port URL, for each class or non-class function located in the attached file/s.

Note:

All Non-Class functions in a file are considered a single class and are mapped to a single port. However, non-class functions from different files in the same WSDL configuration are given a different port URL.

7. Once all the added files have been defined, click next to view the configuration summary, dialog.
8. Click Finish to approve the configuration settings and generate the WSDL file or Previous to go back and modify settings.

Notes:

If the selected files are not part of a project or open in the editor, a dialog will appear instructing you to choose files from within a project.

If some of the parameters or return values of a function are of types not found in your project, an additional dialog will appear instructing you to add the classes to the project and restart the Wizard, or enter the URL to the XML file containing the class schema

Chapter 7 - phpDoc Support

IN THIS CHAPTER...

PHPDOC BLOCK

ADD PHPDOC DESCRIPTIONS

PHPDOCUMENTOR SUPPORT

CREATE A NEW CONFIGURATION

phpDocs support and PHPDocumentor integration pertain to two different but related features that enable developers to generate professional documentation directly from the PHP project source code.

phpDocs delivers with the Zend Studio editor a preset means for adding phpDoc comments to files by providing an input line when including statements, classes, class variables, and constants to the code. Developers are prompted to immediately add a description ensuring that the added elements are documented in their context and in real-time. This feature also includes, code completion in comments for phpDoc tags and variables and new syntax highlighting for phpDoc.

PHPDocumentor⁶ has been integrated into Zend Studio to provide easy and efficient generation of PHPDocs directly from the Zend Studio interface, eliminating the need to seek alternatives outside Zend Studio to generate PHPDocs.

The PHPDocumentor comes with an easy to use built-in PHPDocumentor wizard.

phpDoc comments are used to:

- Describe how the element is used in the application code
- Generate API documentation for the application via phpDocumentor
- Generate structured descriptions in Zend Studio's Code Completion library

phpDoc Block

phpDoc blocks are descriptive comments that are part of the application code. They are used to describe the PHP element in the exact location in the code where the element appears. The block consists of a short description, long description, and phpDoc tags. phpDoc blocks also serve as the input for phpDocumentor and/or Zend Studio's structured descriptions that are shown attached to the Code Completion window.

⁶ PHPDocumentor is courtesy of www.phpdoc.org. PHPDocumentor is the world standard auto-documentation tool for PHP, written in PHP.

Add phpDoc Descriptions

Descriptions are added to the code for the distinct purpose of documenting the code to assist other people in understanding the code's logic.

When phpDoc descriptions are added to the code, developers' gain two additional distinct benefits, not only can API documentation be generated via the phpDocumentor, they can also benefit from Code Completion assistance.

There are two ways to add phpDoc comments to the code.

1. Directly to the code – manually
2. To a PHP element – File Manager

To manually add a phpDoc comment to a PHP element:

1. Add an element to the code in the editor
2. Position the cursor above the code element and type `/**` and press enter.

This will automatically add the phpDoc comment area in the code:

```
/**
 *
 */
```

If you add the same phpDoc comment above code elements such as a functions and variables, the phpDoc comment automatically adds the appropriate tag/s for documenting the code element.

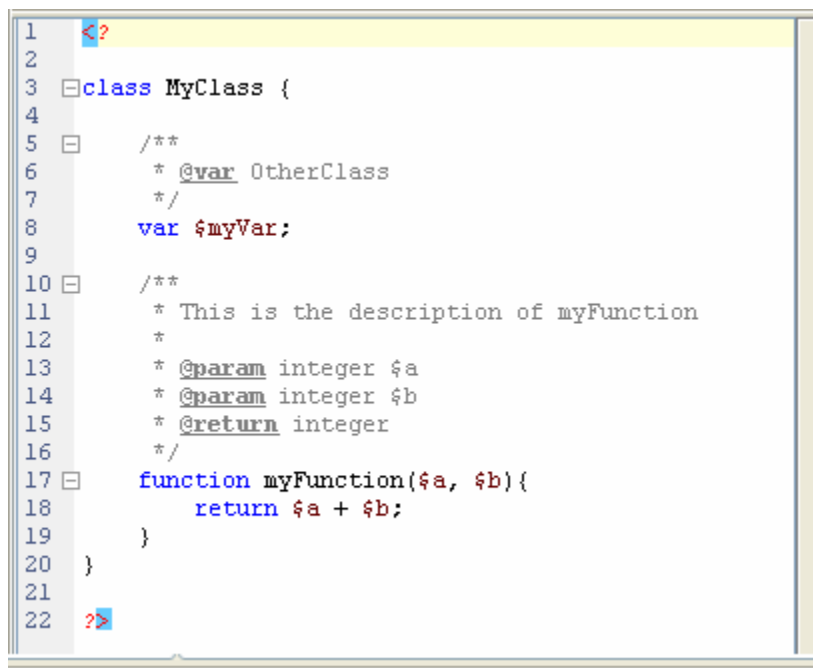


Figure: 21 - Auto Insert phpDoc Comment Tags

Notice the code folding function that expands and collapses the relative context of code elements using the `[+]` and `[-]` signs to the left of the code.

Descriptions that are added above a code element the code are also automatically added to the Code Completion bank so that the next time the code element is used it is readily available from the code completion.

To add a phpDoc description to a PHP element through the File Manager:

1. Open the Inspectors tab (under the File Manager) to display a list of the classes, functions and constants in the current file/project.
2. Right-click on an element to add to the phpDoc block and choose Add Description.
3. An option menu will open select, Add Description. A phpDoc block will be added to the selected element with a short description. This description is completely editable, so the description can be easily changed.

Note:

If you wish to add a comment to the code but you do not want the comment to be visible in the API documentation use standard comments `/*`.

The Code Completion display for an element opens along with the element's description. When this happens, click the Add Description icon in the right-hand corner of the description window.

Zend Studio will add a phpDoc block to the selected element with a short editable description.

Note:

To activate, deactivate, or configure phpDoc comment Insertion, use the preferences menu to control the settings: Tools | preferences | editing.

phpDocumentor Support

phpDocumentor uses an extensive template system to change phpDoc (blocks) comments into readable, useful formats.

PHPDocumentor Wizard is Zend Studio's interface with phpDocumentor. Via the PHPDocumentor Wizard, you can generate API documentation in readable format.

Use the phpDocumentor Wizard to:

- **Create a New Configuration** - creates a new configuration file that includes all required settings, e.g., files and directories to parse, files and directories to ignore, Output directory, and Documentation title. The new configuration will be saved for future use.
- **Load a pre-defined config file** - loads a stored configuration file.

Create a New Configuration

To create a new configuration:

1. Select phpDocumentor from the Tools menu.
Zend Studio's phpDocumentor Wizard opens.
2. Make sure that the Create new configuration radio button is selected and then click Next.
phpDocumentor prompts you to define: Files to parse, Directories to parse, Files to ignore, Packages to parse, and Patterns to ignore.
3. Click Next.
phpDocumentor prompts you to define: Output directory and Converter type.
4. Click Next.

phpDocumentor prompts you to assign:

- Documentation title
- Package name
- Category name
- Custom tags.

You can also enable any of the following settings:

- Parse@access private and @internal/{ @internal} }
 - Generated highlighted source code
 - JavaDoc-compliant description parsing
 - PEAR package repository parsing. (Refer to the phpDoc Manual online at www.phpdoc.org for complete descriptions.)
5. Click Next.
phpDocumentor prompts you to save the current configuration.
 6. Click Finish to generate the file.
Your configurations will be saved and available for the next time you run the phpDocumentor Wizard.

Chapter 8 - Searching

IN THIS CHAPTER...

SEARCHING ACTIVE FILES

SEARCHING MULTIPLE FILES

USING REGULAR EXPRESSIONS

Zend Studio provides a powerful search engine for locating text strings. The array of search options facilitates the development process.

Searching Active Files

Zend Studio features four types of search options for locating text in an active file:

- **Find** - Search for individual occurrences in the currently edited file.
- **Find and Replace** - Search for individual occurrences in the active document with the option to replace.
- **Find Next/Previous** - Search the next/previous individual search result occurrence in the active document.
- **Find in Files** – Search a selection of files per directory and/or Project.

Use these various options to locate, open, reference and edit code within the active file or across multiple files.

Searching in Multiple Files

The Find in Files function allows you to search multiple files returning a list references to all occurrences. This search option allows you to search for text in file(s) in a specified directory or FTP location.

To search text in multiple files (Find in Files):

1. From the main menu go to: Search | Find in Files.
The Find in Files dialog appears.
2. Enter the search subject in the Find What box.
3. If you want to limit the search by file type, select or type a file type in the File Mask field.
4. Browse for the directory you wish to search or check the entire project check box. In this case, the “find in” field is disabled.
5. Check the options and the advanced areas on the dialog to fine tune the search:
 - a. **Match Case** - Search will be case sensitive
 - b. **Match Whole Words** - Search will skip matches that are part of a larger word
 - c. **Regex Search** - Uses regular expressions in search criteria⁷

⁷ This option can also be accessed from “find and replace” and “find in files”

- d. **With Subdirectories** - Also searches in the subdirectories below current location
 - e. **Open in a New Tab** - Displays the search results in a new tab in the Message window
 - f. **Entire Project** - Searches in all the files in current project. In this case, the "find in" field is disabled.
6. Click Find. The search results are listed in the Messages window with a tab indicating the search subject.
 7. To edit a file appearing in the occurrences list, double-click the relevant path and the file will open in the Editing window.

Caution:

To avoid errors, Linux users should remove recursive links from files being searched.

Searching with Regular Expressions

Zend Studio allows you to perform searches using a regex (regular expression).

To search using a regex:

1. Open the Find Dialog from the Search menu.
2. Select Find.
3. Select the option, Regex Search.
4. In the "Find What" field, enter the regular expression.

To display the search results in the Messages window with a tab indicating the search subject use the Find in Files option.

Use the following syntax to search and replace with Regex:

- Search: "(.*?)"
- Replace: _("\$1")

Chapter 9 - Code Inspection

IN THIS CHAPTER...

INSPECTING FILES

INSPECTING PROJECTS

VIEWING PHP FUNCTIONS

The Inspector pane is used to graphically map code elements. Code elements include: class names, members, functions, class constants, class functions, constants, include-files and soap clients.

There are three inspector tabs for viewing different levels of code mapping:

- **File Tab** - Displays in a tree directory all Code Elements declared in the current active file.
- **Project Tab** - Displays in a tree directory all code elements which are declared in the files belonging to the project or open in the Editor.
- **PHP Tab** - Displays a tree directory of all PHP Classes and Functions. (Selecting a function and pressing F1 opens the PHP manual in the Internal browser).

Inspecting Files

The File Inspector displays information pertaining to the active file. Toggling to another open file automatically updates the File Inspector display to reflect the information in the new active file.

The File Inspector displays the following active file components:

- Constants
- Class; constants, variables, functions, Super Classes (Extends) and the interfaces implements
- Functions included in the file
- Include files
- Soap Clients

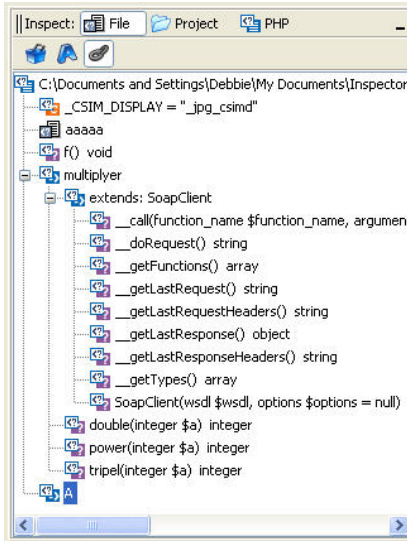


Figure: 22 - File Inspector

The File Inspector lists components in a hierarchical tree. This tree can be expanded and collapsed to help view information. Inspector icons appear next to each item to indicate if it is a constant, class member, variable, member function or an included file.

Inspector Toolbar Options:

- **Grouped** - Group by type (class, Function.)
- **Sort by Name** - List items alphabetically (as opposed to their order in the file)
- **Link to Editor** - activates and disables the option to highlight the appropriate key in the inspector when placing the cursor on an item in the code.

Basic file inspection operations:

- **Go to source** - Double-click on the tree node to automatically navigate to the statement in the active file.
- **Add description** - to automatically insert a DocBlock, right click the tree node and select "Add Description" from the right click menu or manually insert a DocBlock using `/**`.
- **Soap Client Support** - Double-click on the tree node to automatically navigate to the Soap Client in the active file.
- **Goto WSDL file** - Use the right click menu to open the linked WSDL file in the editor.
- **List all include-files** - View all "include" files and open them in the editor using the right click menu.

Note:

DocBlocks are used for adding descriptions into the code. DocBlocks also adds the DocBlock's short description to a tooltip that lists the items listed in the file inspector.

Inspecting Projects

The Project Inspector displays information pertaining to active files and projects (currently open in the editor). Toggling to another project automatically updates the Project Inspector display to reflect the information in the new active project.

The project Inspector displays the following file and project components:

- Constants
- Class; constants, variables and functions
- Functions included in the file
- Soap Clients

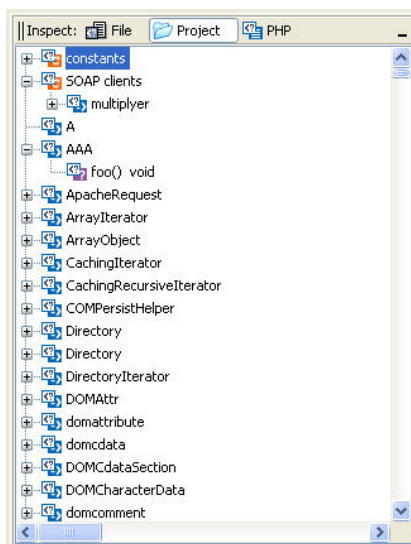


Figure: 23 - Project Inspector

Basic project inspection operations

- **Go to source** - Double-click on the tree node to automatically navigate to the statement in the active file or select the option from the right-click menu.
- **Add description** - to automatically insert a DocBlock, right click the tree node and select "Add Description" from the right click menu or manually insert a DocBlock using `/**`.
- **Soap Clients support** - Double-click on the tree node to automatically navigate to the Soap Client in the active file.
- **Goto WSDL file** - Use the right click menu to open the linked WSDL file in the editor.

To view the constants, classes and members for the current file only, use the File Inspector.

Note:

DocBlocks are used for adding descriptions into the code. DocBlocks also adds the DocBlock's short description to a tooltip that lists the items listed in the file inspector.

Viewing PHP Functions

Get additional reference information on specific PHP Functions and classes by accessing the online PHP Manual from inside Zend Studio.

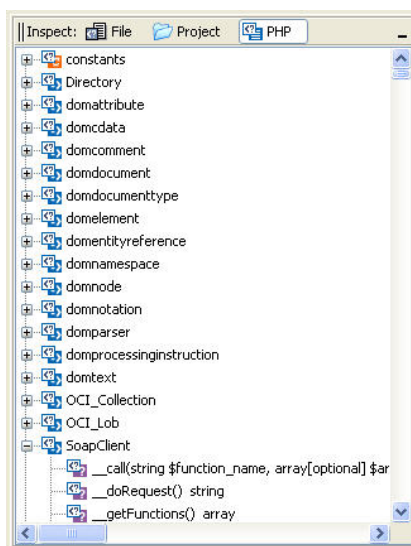


Figure: 24 - PHP Function Inspector

The Inspectors PHP tab lists all existing PHP functions constants and classes in alphabetical order. Each one of these functions can be easily investigated directly from inside the IDE to obtain function class and constant descriptions and usability options.

To ensure the highest level of accuracy, information is directly viewed from the online PHP manual, which opens in the IDE's Internal browser.

To open the PHP Manual in the internal Browser, select a function from the list of PHP functions and press F1 (or use the right-click menu).

Chapter 10 - Debugging and Analyzing Code

IN THIS CHAPTER...

INTERNAL DEBUGGER
SERVER DEBUGGING
CONTROLLING PROGRAM FLOW
USING BREAKPOINTS
CONDITIONAL BREAKPOINTS
MONITORING PROGRAM STATE
CREATING AND MONITORING WATCHES
TRACKING THE STACK
REVIEWING VARIABLES/ASSIGNING VALUES TO VARIABLES
ASSIGNING A NEW VALUE TO A VARIABLE
VIEWING OUTPUT IN THE OUTPUT BUFFER
COMMUNICATION TUNNEL
HTTP AUTHENTICATION
TROUBLESHOOTING THE COMMUNICATION TUNNEL
ANALYZING CODE

Zend Studio supports two debugging capabilities:

1. **Internal** – For local debugging and uses a local copy of PHP 4 or PHP 5.
2. **Server** - For debugging remote servers, that are accessed online from the Studio application using Zend Platform's Studio Server Component on the server/PHP side. When debugging files on a remote server the debugger uses the PHP installed on the remote server. Additional options for debugging are to using Zend Core or separately downloading the debugger component and install them on the server/PHP side.

Zend Core is available for download from: <http://www.zend.com/downloads>

The debugger component is available for download from:

http://www.zend.com/free_download/studio, (Supported Operating Systems are: Windows x86, OS X Mac and Linux x86).

Note:

Do not add the internal debug server which comes with the Internal PHP to the Web server PHP extensions, since it is incompatible with the SAPI module and will NOT work.

The internal debugger is a suite of debugging tools that allows the developer to debug applications as they are being developed-before they are uploaded to the web server.

Zend Platform Studio Server Component enables the developer to debug files where they sit-on the content server.

Setting Debug Options:

To view a list and descriptions of the configurable debug properties go to: Tools | Preferences | Debug.

- These settings can be configured to suit your requirements.

- Many of these settings were setup during the installation process.

Internal Debugger

The Internal Debugger enables developers to locally validate freshly developed code before deploying to a web server. The internal option means that only files located in local directories can be debugged. When debugging internal files the Zend Studio Internal Debugger uses its own PHP version that was installed together with Zend Studio in the installation process. This PHP version is compatible for PHP4 and PHP 5.

Configuring Internal Debugging

1. Open the Preferences window from the main menu, select Tools | Preferences.
2. Select the Debug tab.
3. From the Debug Server Configuration area of the Debug tab, select a Debug Mode (Server/Internal).
The debug mode setting determines the debug toolbar settings (Internal or remote debugging)
4. Click OK
These settings will be applied when debugging.

The following is a list of tools that can be used in the debugging process:

- **Debug Messages Window** - Locate and define errors using the messages generated and displayed in the Debug Messages window.
- **Variable Window and Watches Window** - Watch and reference variables, functions, classes, and expressions.
- **Stack Window** - Monitor the call stack and passed variables.
- **Debug both the Calling and Called Functions** - Using Step in, Step out, Step over, and Breakpoints.
- **Control the Debugging Session** - Use complete, or line-by-line debugging options using tools such as Breakpoints and Go to Cursor.
- **View and Render Standard Output** - Using the content generated to the Output window.
- **View Buffer** - Using the content buffered in the Buffer Window.

To expand your internal debugger functionality you'll need to add extensions to it. You can add some extension directives to the php.ini file.

The syntax follows:

```
extension_dir="(EXTENSION DIRECTORY)"
```

```
extension=(EXTENSION NAME)
```

Where {EXTENSION DIRECTORY} is the directory containing the extension file(s).

For example, on Windows, you can use:

```
C:\ProgramFiles\Zend\ZendStudioClient-5.1.0\bin\php4
```

You can obtain more information from:

http://www.zend.com/support/knowledgebase.php?kbid=14&view_only=1

Remote (Server) Debugging

The Remote (server) Debugger enables developers locally validate code that has already been deployed on a remote server. The remote option means files located on a remote server can be debugged using the Zend Studio Debugger. Another optional addition is to debug local and remote files using the remote debugger's "Local Copy if available..." option (Tools | Debug | Debug URL. When debugging remote files the Zend Studio Debugger uses the PHP version that is installed on the Remote server.

The PHP/Zend Engine and Zend Platform Studio Server Component can typically reside together in one of two locations, either on your local drive or on your hosting server.

With remote server debugging:

- Code can be debugged internally or on a remote server.
- Debug sessions can be run online or off-line.
- Online-browser navigation can be interlinked with debugging to allow the debugger to follow a specified browser sequence.

Server debugging must be enabled to debug using the server PHP/Zend Engine. Disable server debugging to use the local PHP/Zend Engine.

Configuring Remote Server Debugging

1. Open the Preferences window from the main menu, select Tools | Preferences.
2. Select the Debug tab.
3. From the Debug Server Configuration area of the Debug tab, select a Debug Mode (Server/Internal).
The debug mode setting determines the debug toolbar settings (Internal or remote debugging)
4. Click OK
These settings will be applied when debugging.

Note:

You can also enable/disable Server Debugging from the Project Properties window. Typically, this is done at the time the project is created. See Setting Project Properties.

Debug URL

Debug URL allows you to run the debug procedure on pages currently deployed on a production server.

Here are some of the advantages of this method of debugging:

- Use Zend Studio and your browser in conjunction with the Zend Platform's Studio Server Component.
- Interact with a live Website, while debugging the pages. You can use the flow of the site to load and debug files. This helps to debug pages, such as post-login pages, which are normally difficult to debug.
- Output from the debugger is rendered in the browser and displayed in the Output window of the Zend Studio as it is generated.
- Load the called page into Zend Studio. The loaded page is run up until the first PHP line and then stops.
- Debug code using, Variables, Stacks, Watches and Debug Messages.

To setup the Debug URL procedure for server debugging:

1. Install Zend Platform on the web server.
2. Access Zend Platform's Studio Server Component from Zend Studio through: Tools | Zend Platform.
Zend Platform will open on the Studio Server tab in your browser.

Note:

Make sure the remote server's settings are properly defined in Zend Studio. This can be done as a system default or at a project level.

To define the system default, go to Tools | Properties | Debug and manually define the remote server's URL.

To define a specific project's settings, go to Project | Project Properties, disable the "Use system Defaults" option and manually enter the remote server's URL.

3. Once the Studio Server tab is opened go to: Studio Server | Settings.
4. Add Zend Studio's Host name to the allowed host list.
5. You may receive a "Restart Server" message.
The new settings will be applied after the server is restarted.

The Studio Server Component has an option to give the files you are working on first priority when debugging using the “Local Copy” option.

In order to achieve this, the Server application follows this hierarchy when it requests files:

1. Checks if the file called is currently open in the Zend Studio, if found it uses this file.
2. Searches for the file in the open project's path; if found it uses this file.
3. Searches for the file in the server path; if found it uses this file.

This hierarchy prevents having to upload latest revisions.

For example: if a project-file with same name as a server file prevents you from debugging the server file, you may wish to temporarily rename the local file or temporarily remove it from the project. This allows the server resident file to be debugged.

Running Debug URL

To run Debug URL:

1. From the main menu select: Debug | Debug URL to open the Debug URL dialog and configure the following settings:
 - **Open Browser at** – the remote server's URL. If a file is not specified the browser will open the URL but remain blank.
 - **Debug First Page Only** - Only the first page is debugged. As soon as you click OK, the browser opens and waits for the debugger. When complete, the page opens in the browser.
 - **Debug All Pages** - The specified page and all the pages linked to it are debugged. The browser waits for the debug of each page before displaying it.
 - **Start Debug from** - Only the specified page is debugged.
 - **Continue Debug from this Page** - Selecting this option continues to debug all the pages linked to the URL.
2. Specify the location of the files to run the debug session.
Debug files on:
 - a. The Server
 - b. Use a Local Copy if available. If this option is chosen but there is no available local copy the debugger will use the copy on the server.

Note:

The browser displays little if anything during the debugging session, because the HTTP output almost always occurs at the end of the debug process. It is normal for the browser's status-bar to display a message such as "Waiting for Debugger Response".

Controlling Program Flow

Zend Studio provides tools for controlling the execution flow of an application. Specifically, it supports two types of breakpoints that force the execution to stop at preset points.

These are:

- **Breakpoints** - a normal breakpoint stops the program unconditionally when the application execution encounters it.
- **Conditional Breakpoints** - a conditional breakpoint stops the program only when a user-defined condition is met.

Using Breakpoints

Breakpoints are markers that instruct the Zend Platform's Studio Server Component to stop the PHP/Zend Engine at a specified line of code. This feature helps isolate areas of an application that require testing.

- Use breakpoints to stop an application from running to allow you to review information such as Variables, Watches, Stack, and Debugging Messages.
- Use breakpoints to section the program to allow you to test smaller, more manageable blocks of code.

Every breakpoint created appears in the Breakpoint tab. The Breakpoint tab lists the breakpoint file location. This includes file path, file name and line number. In addition, the icons indicate whether the breakpoint is Enabled or Disabled.

To add a Breakpoint:

1. Stand on the relevant line of code and click the line number.
2. The line will get a pink background indicating that a Breakpoint has been added.
You will also notice the breakpoint has been now added to the Debug Window's Breakpoints tab.

Breakpoint status can be understood as follows:

- **Enabled** - An enabled breakpoint interrupts the execution of the application at the point in which it is placed.
- **Disabled** - A disabled breakpoint remains in the project and displays in the Edit Window and Breakpoint tab, however during debugging a disabled breakpoint will not cause a break in execution.

To Enable/Disable All Breakpoints:

1. Go to the Breakpoints Tab
2. Make sure that no breakpoint is selected and open the right-Click menu.

The menu options are:

- Remove All - Permanently delete all bookmarks.
- Enable All – Activate all Bookmarks.
- Disable All – Deactivate All Bookmarks

To edit and modify a single Bookmark:

1. Go to the Breakpoints Tab
2. Select a Bookmark and open the right-Click menu

The menu options are:

- Remove - Permanently deletes the bookmark.
- Enable – Activates the Bookmark.
- Disable – Deactivates the Bookmark
- Goto Source - Jumps to the Bookmark in the code
- Edit Condition

You can also select multiple Bookmarks from the list using CTRL, however the right-click menu changes to allow the remove, disable and enable options.

Conditional Breakpoints

Zend Studio supports conditional breakpoints as well as normal breakpoints. Conditional breakpoints cause a break in the execution of the program when a user-defined condition is met. For example, a normal breakpoint stops the application when the breakpoint is reached. A conditional breakpoint can be defined to stop the application only after a certain number of iterations in a recursive loop.

To define a conditional breakpoint:

1. In the Editing window, insert a standard breakpoint at the appropriate point in the application code. The breakpoint line appears highlighted.
2. Click Breakpoints from the Debug window's Toolbar .A list of breakpoints currently defined in the application is displayed.
3. Select the breakpoint for which you want to define a condition and right-click to open the Edit menu.
4. Select Edit Condition to open the Edit Condition dialog.
5. Insert a breakpoint condition.
6. Click OK.

The breakpoint will now exit to the debugger only when the breakpoint condition is met.

Monitoring Program State

Zend Studio provides a number of tools for monitoring program state.

These include:

- Stack Tracking
- Monitoring Watches
- Reviewing Variables
- Assigning Values to Variables
- Viewing the Output Buffer

Creating and Monitoring Watches

Debugging often includes watching variables. Watches help you narrow the scope of evaluated expressions in the debug process.

In the Watches area you can permanently place the items you want to watch per project, such as variables, objects, and expressions.

With Zend Studio you can:

- Add Watches
- Remove Watches (Delete)
- Edit Watches

To add a watch:

There are several ways to add watches:

1. Go to the Debug Menu and select Add Watch
2. In the Editor, select an expression in the code, open the right-click menu and select the option "Add Watch" from the menu.
3. In the Debug window's watches tab, open the right click menu and select the option "Add Watch" from the menu.
If an expression in the Editor is selected, it will automatically appear in the Add Watch dialog, otherwise expressions can be manually added to the watches list

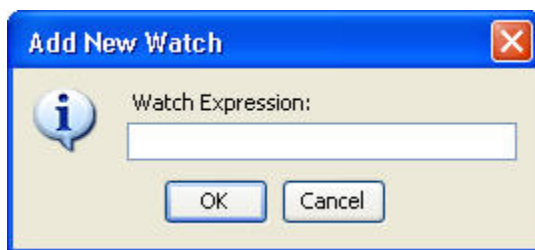


Figure: 25 - Add New Watch Dialog

Watches display the watch expression, type and current value (results).

In the following image, you can see both arrays and primitive variables.

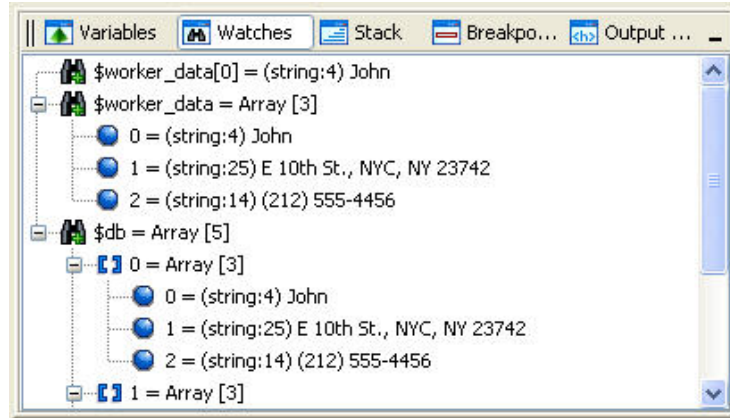


Figure: 26 - Watches List

Whenever content is setup in a certain hierarchy, the watches tab will display the content in accordance to the hierarchy in a tree type display.

Removing Watches:

To remove a single watch from the Debug window's watches tab:

Select an item from the watches list, open the right-click menu and select the option "Remove Watch" from the menu.

To delete all watches from the Debug window's watches tab:

Make sure no items are selected in the watches list, open the right-click menu and select the option "Remove Watch" from the menu.

Expressions can also be edited from the Debug window's watches tab. Select an item from the watches list, open the right-click menu and select the option "Edit expression" from the menu.

Tracking the Stack

When a function is called, the application execution routine temporarily leaves the current function in order to execute the called function. Once it executes the called function, the execution routine returns to the original calling function and continues with the next line of code.

Note:

Main() represents the set of code statements from your script's main section. Think of it as a discrete function.

The following information can be gathered from the Stack Window when debugging has stopped:

- **Called Functions and Parameters.**
The called functions and parameters appear in the stack as written in the line or code.
- **Called Function's Location.**
This can include a full path or it can be a reference line to a function declaration in the code
- **The Main Calling Line of Code.**
This refers to the line number in which the calling statement occurred in the main()
- **Parameter Types.**
Displays an icon, which identifies the parameter as a primitive or arrays type variable.
- **Parameter Values.**
Shows the parameter values that were passed in the function call. Based on the stack, you can now see what transactions existed and you can use the Stack window to jump to the line where a function call occurred. This can be done by right clicking on the stack item and selecting Goto Source.

Reviewing Variables/Assigning Values to Variables

Zend Studio allows you to review variables on the fly when a breakpoint is reached. You can view a variable value by simply placing the cursor over the variable. The variable value is displayed in a tooltip box.

Tooltips can also be displayed for expressions. In order to display a tooltip for an expression, the expression must be selected. Once selected, pass the cursor over the highlighted expression and the tooltip appears with the value of the expression.

Values can display Null when a Debug Session is inactive or when a particular expression(s) is not available to the script.

The tooltip can be customized to display the variable in serialized form. For example, a variable `$myArray` displays the same as the return value of the PHP function `serialize($myArray)`.

Assigning a New Value to a Variable

Zend Studio allows you to manually add values to variables or watches, as needed. You can use this functionality to reduce iterations, force an error or test a value.

To Assign a New Value to a Variable:

1. In the Variable tab, right click on a variable item and select Assign Value from the menu. The Assign Value dialog box appears.
2. Enter the new variable into the New Value box and click OK.

To Assign a New Value to a Watch:

1. In the Watches tab, right click on a variable item and select Assign Value from the menu. The Assign Value dialog box appears.
2. Enter the new variable into the New Value box and click OK.

Note:

Values can also determine the debugger's behavior by adding variables in debugger stopping points (such as in a breakpoint).

Viewing Output in the Output Buffer

This Tab displays the output of your PHP scripts. Use this feature to check scripts that are written to take advantage of the output buffer. Using the output buffer can help reduce the packet size by allowing more information to be sent in a packet.

Buffering output can also allow you to use various parsing options, such as taking an XML script and parsing it for a CGI.

Communication Tunnel

Zend Studio supports a communication tunnel with Zend Platform's Studio Server Component. The communication tunnel solves the communication problem, which occurs when Studio or the Server Component is behind a Firewall or NAT. (to learn more about configuring Zend Platform's Studio Server Component go to: Chapter 15 - Setting Server Components on page, 161.

This feature enables Zend Platform users to also view and edit Event source code in Zend Studio; conversely, Zend Studio users can access the Server debugger via the same communication tunnel that routes full duplex traffic over HTTP.

Configuring the Communication Tunnel

To configure Tunneling Settings in Zend Studio:

1. Open the Tunneling Settings dialog from Tools | Tunneling Settings.

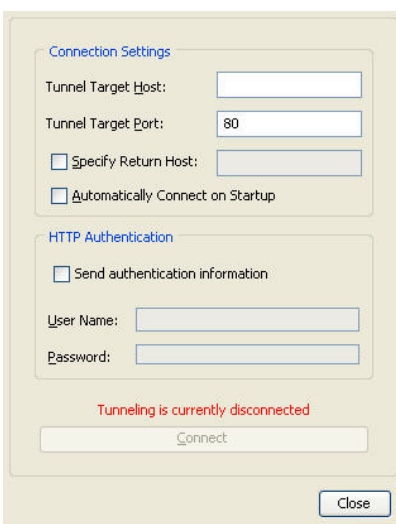


Figure: 27 - Tunneling Settings Dialog

2. Enter values for the following settings:

Setting	Description
Tunnel Target Host	Address of Web server where the debugger resides.
Tunnel Target Port	Port of the Web server on which the debugger resides.
Specify Return Host	When enabled, this is the address of the main server in the cluster.
Automatically Connect on Startup	Enables the communication tunnel when Zend Studio starts up.
Send Authentication Information	Use this option when working with a Web server that requires HTTP authentication. Studio sends the authentication information in the header. Note: This assumes that the user account is set up on the Web server.
User Name:	User name as defined on the Web server.
Password	User password as defined on the Web server. Note: Whenever you use the debugger, the server will use the User Name and Password specified here.

3. Click Connect to connect the Tunnel Target Host over the specified port.

Notes:

Information in: Preferences | Debug must match the information in the Tunneling Settings for tunneling to work.

Tunneling is not suitable for Windows Web-servers.

Broadcasting Port

Studio's Communication Tunnel is implemented via a persistent broadcasting port that broadcasts information about tunneling to Zend Platform and to the Toolbar.

The broadcasting port is configured from: Tools | Preferences | Debug.

HTTP Authentication

Zend Studio supports HTTP authentication. This enables user to send HTTP authentication information (user name, password) together with the header sent to the server. Therefore, you can specify that tunneling to a server requires authentication, thereby improving security.

Troubleshooting the Communication Tunnel

If Studio is unable to connect to the target server, you will get an error message with the response from the server. The table below describes the most likely causes and a recommended action for successfully establishing a connection with the target server.

Possible Cause	Recommended Action
The server address or the port you entered is incorrect	Enter the correct server information in the Tunneling Settings dialog.
HTTP authentication is required	Enter authentication information in the Tunneling Settings dialog box; then click the 'Send authentication information' checkbox.
The dummy file content or location on the server is incorrect	The dummy file on the server side was either changed or does not exist. You will need to insure that the correct dummy file with the correct content is placed in the correct directory on the target server. The correct dummy file is created and located properly as part of the Installation procedure. The problem here is post-installation.
You are not allowed to connect with the server via the communication tunnel	You must have tunneling permissions in the php.ini file. Make sure that the zend_debugger.allow_tunnel variable is properly configured.

For any other cause, or additional information, use one of our support options.

Analyzing Code

The Zend Studio code analyzer helps developers to analyze static source code to enforce good coding practices and scan PHP code. The Code Analyzer achieves this functionality by attempting to reconcile problematic code and locating unreachable code (code that has been defined but is not used or with empty variables).

The Code Analyzer supplies users with a detailed error log while focusing on the error location in the file that is open in the Editing Window. In addition, it supplies you with practical suggestions for improving the code.

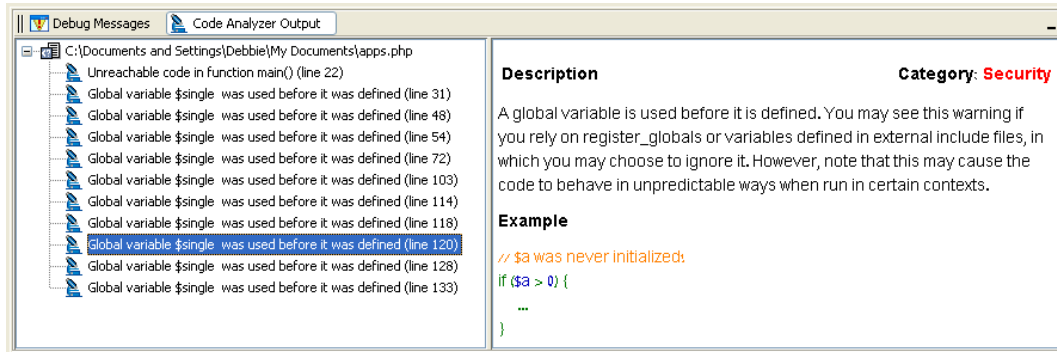



Figure: 28 - Code Analyzer

To run the Code Analyzer:

1. The Code Analyzer can be run on an active file at any time, from the main menu by pressing  or selecting analyze Code from the right-click menu.
2. Double-click the code error in the Code Analyzer Output tab in the Messages window to move the cursor to the exact location in the file.
3. From the Tools menu, select Analyze Code. The Code Analyzer Output window opens with the analysis.

Platform Integration

Studio 5.5 enables events collected by Platform to be collected and displayed in a separate Studio pane. This enables:

- Presentation of all of the Events listed on the Zend Platform server.
- Debugging and profiling of all the events listed on the Zend Platform server.

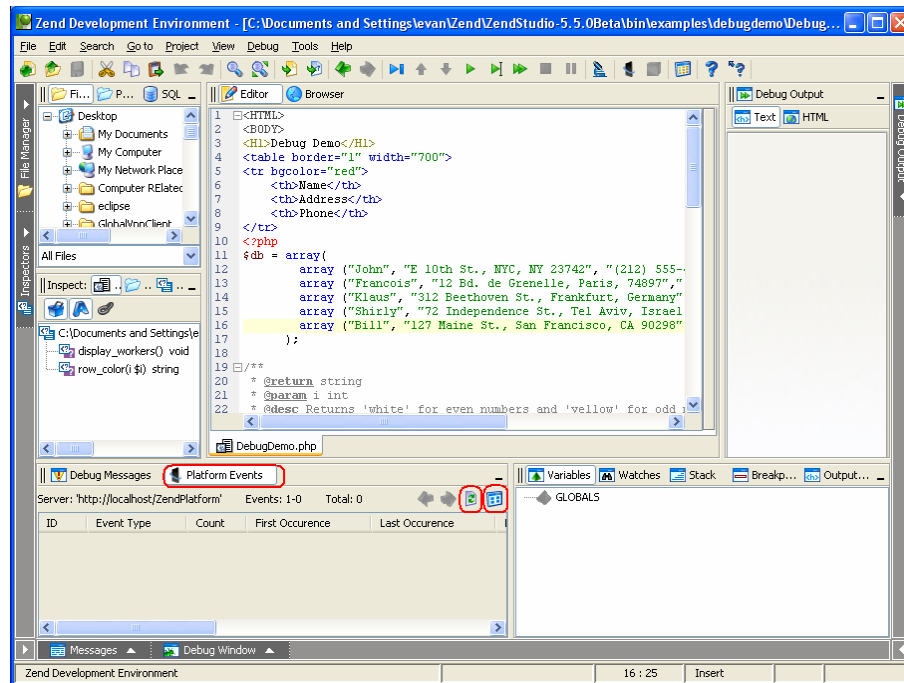


Figure: 29 - Platform Integration

Enabling Platform

1. Open the Preferences dialog (click **Tools | Preferences**).
2. Select the **Platform** tab.
3. Select the desired Platform GUI ("**Default**" —or— enter the **URL** of your Platform server).
4. Enter your Platform **User Name** and **Password**, click **Apply/OK** and return to the Studio 5.5 GUI.

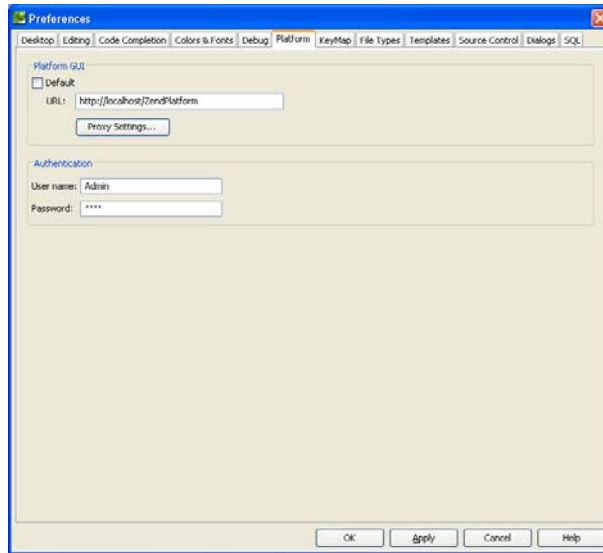



Figure: 30 - Enabling Platform - Preferences

5. Click the **Settings** icon () at the far right. The **Event List Settings** dialog will open. Select the required **Refresh** settings, **Limits**, **Event Sorting** and **Event Filtering**.

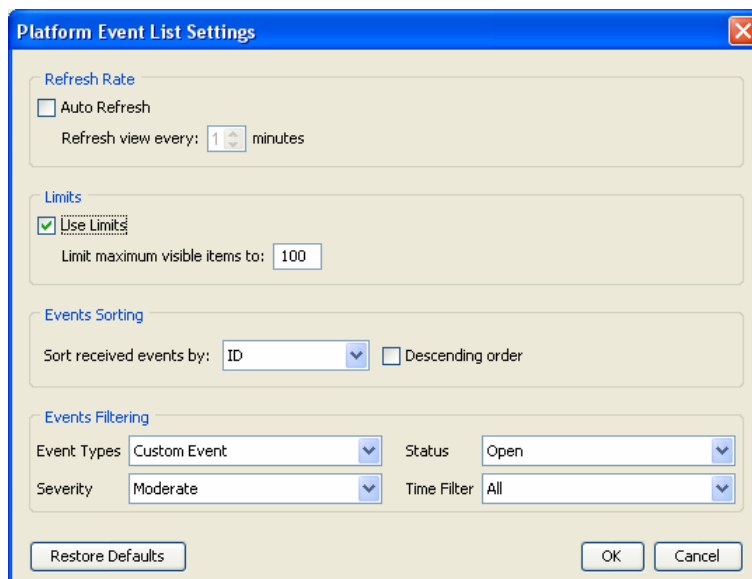



Figure: 31 - Event Settings

Note:

Auto Refresh must be enabled in order for Events to be updated. However you can use the **Refresh** icon  to receive events immediately.

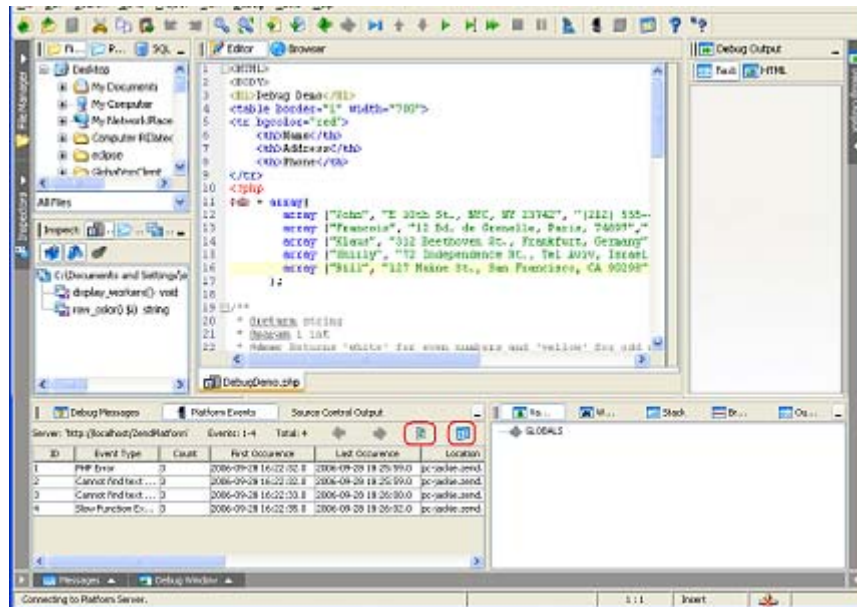


Figure: 32 - Events Listed

- Click on the desired column to sort.
- Expand the view to full length if desired.

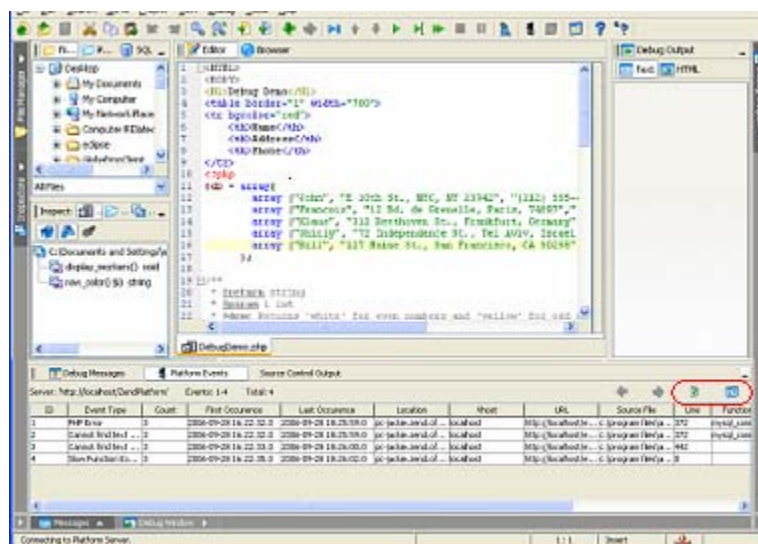


Figure: 33 - Expanded Event View

Chapter 11 - Profiling

IN THIS CHAPTER...

PROFILER INFORMATION TAB

PROFILER FUNCTION STATISTICS TAB

PROFILER CALL TRACE TAB

Profiling is an essential tool in improving PHP application performance. Profiling summarizes the data that makes up the PHP application and represents it in the form of a graph. The graph sets out the important features of the application. By placing timers within the code and running them over and over, the profiling tool is able to build a "profile" of how fast or slow specific areas of the application will run. Zend Studio provides a powerful profiling tool designed to help discover bottlenecks and other areas that need to be optimized to improve the program's performance. An extensive library of profiling benchmarks is included with Zend Studio.

To Run the Profiler:

1. Open the Profile URL window select, Tools | Profile URL.
2. Accept the default URL or change and click OK.
3. The browser presents the requested page. After a few seconds, while the Profiler accumulates information, the Profiler Information window appears.

Note:

The Profiler automatically detects the application's URL. However, the URL field is editable to another URL can be typed and used for profiling.

The Profiler user interface contains 3 tabs:

- **Profiler Information** - provides general information on the profiling duration and date, number of files constructing the requested URL and more. In addition, it displays a Time Division Pie Chart for the files in the URL.
- **Function Statistics** - provides you with the list of files constructing the URL and detailed information on functions in the files.
- **Call Trace** - provides an hierarchical display of functions according to process order, enabling you to jump to the function, view the function call, function declaration, details and more. At any time, you can dock the entire Profiler interface in the Debug Messages workspace.

Profiler Information Tab

The Profiler Information tab provides general information on the profiling duration and date, number of files constructing the requested URL and more. In addition, it displays a Time Division Pie Chart for the files in the URL. The Profiler Information tab is divided into two areas.

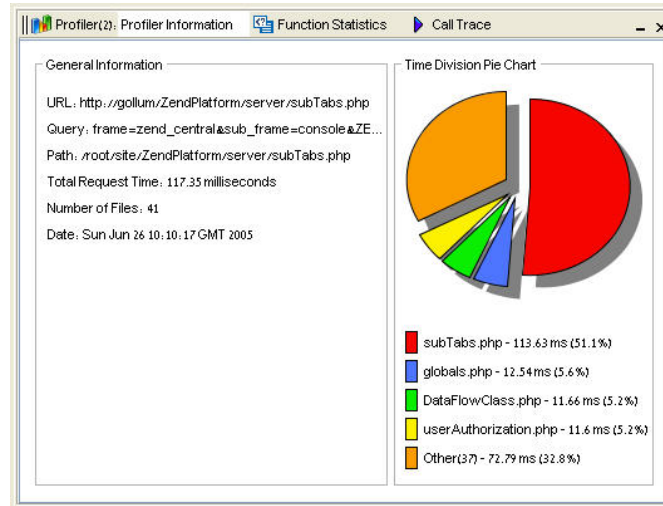


Figure: 34 - Profiler Information Tab

The right side displays time division in a pie chart and the left side provides the following information:

- **URL** - The URL analyzed
- **Query** - The specific query parameters
- **Path** - The exact location of the first file called
- **Total Request Time** - Total process time of the entire page
- **Number of Files** - Number of files composing the page
- **Date** - Date and time that the profiling took place

Profiler Function Statistics Tab

The Function Statistics tab displays the list of files constructing the URL and detailed information on functions in the files.

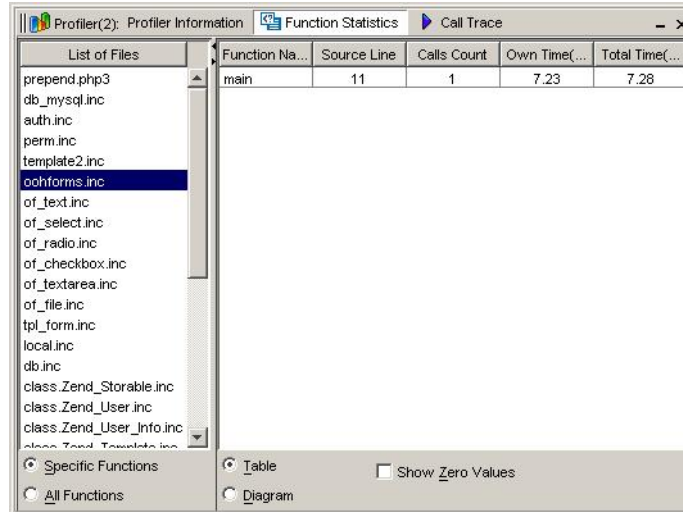


Figure: 35 - Function Statistics Tab

The right pane contains statistics relevant to each function. Initially, all the functions associated with the file appear in the list. To view statistics for all the functions in all files, click All Functions. In this case, the left pane remains blank and the right pane presents the profiling information relevant to all functions as follows:

- **Function Name** - The name of the function.
- **Source Line** - The exact location in the file of the function declaration.
- **Calls Count** - The number of the times that the function was called.
- **Own Time(s)** - The net process duration without internal calls.
- **Total Time(s)** - The total process duration including internal calls.

By default, the function list is sorted according to the calling order. However, you can change the sorting order by clicking on the title of each column in the table. The Profiler has two view modes, Table view and Diagram view. Table view presents the information in a table format; Diagram view presents information in a graphical chart reflecting the "Total" process time and the "Own" time of each function.

To display a function's statistics (in Graph view):

1. In the List of Files view, place the cursor on the file you wish to analyze.
2. A list of all the functions in the selected file appears in a table on the right.
3. Review the Functions information on the right and change the sorting key if necessary.
4. To view the graphical diagram, select Diagram.

Profiler Call Trace Tab

The Call Trace tab provides an hierarchical display of functions according to process order, enabling you to jump to the function, view the function call, function declaration, details and more. To view the hierarchical structure of the functions involved in executing the script, open the Call Trace tab to reveal the following window:

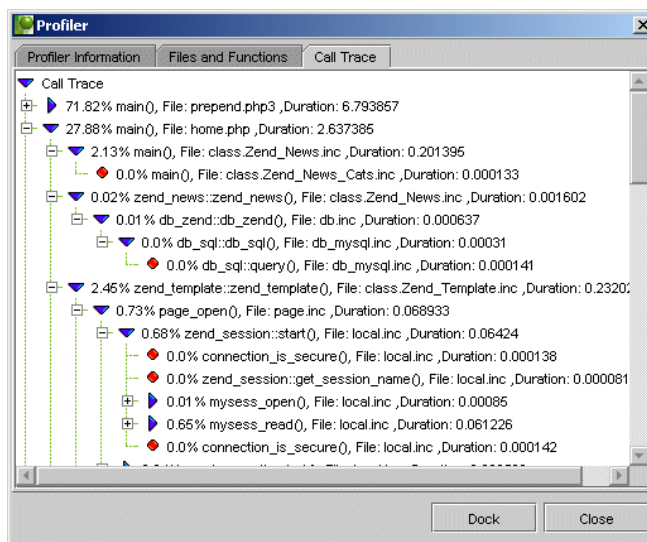




Figure: 36 - Call Trace Tab

Functions that have internal calls (parents) appear in the list next to a blue arrow facing right when collapsed and facing down when expanded. A child function or a function that does not activate internal calls (children) appears next to a red circle. The list of functions is sorted according to the order of process.

Each line details the time out of the total time in percentage, function name and process duration. Placing the cursor on a function displays a tool tip with the exact location of the file as well. The main() function, stands for a fixed function that represents a general call to a file.

For example: `include_once("db_mysql.inc");` means that in this case a main() line will appear in the Call Trace tree .

Use the  and  arrows to expand and collapse the list.

The Call Trace tab supports two global display options:

- **Show File Name** - displays the file name of the function call
- **Show Duration Time** - displays the duration time of the function call

In addition, the Profiler enables you to reach the following functions from the right-click menu.

If you select a parent:

- **Sort By Time** - sorts the function calls within the parent node by duration.
- **Sort By Original Order** - sorts the function calls within the parent node in the order in which they were received.
- **Collapse All** - closes the entire parent node.
- **Expand All** - expands the entire parent node.
- **View Function Call** - opens the relevant file in the editing workspace and highlights the function selected, in the exact location where it was called.
- **View Function Declaration** - opens the relevant file in the editing workspace and highlights the beginning function declaration.
- **View Function Statistics** - focuses on the function in the Files and Functions tab.

If you select a child:

- **View Function Call** - opens the relevant file in the editing workspace and highlights the function selected, in the exact location where it was called.
- **View Function Declaration** - opens the relevant file in the editing workspace and highlights the beginning function declaration.
- **View Function Statistics** - focuses on the function in the Files and Functions tab.

Chapter 12 - Source Control

IN THIS CHAPTER...

USING SOURCE CONTROL DIFF

CONFIGURING CVS

CONFIGURING SUBVERSION

CONFIGURING THE CVS COMMUNICATION TUNNEL

Zend Studio is integrated with two open-source version control systems. This integration enables users to develop PHP applications in Zend Studio and manage versions with industry-standard source control software.

Zend Studio supports the following source control software:

- CVS - <http://www.nongnu.org/cvs/>
- Subversion- <http://subversion.tigris.org/>

Setting Source Control Default

Users can select to work with either CVS or Subversion depending on user or organizational preferences. CVS and Subversion are interchangeable and are defined from the preferences menu (Tools | Preferences | Source Control). Menu options and settings are set to the selected source control application once one is selected.

To set the source control software:

1. Go to Preferences and open the Source Control tab.
2. Choose a Source Control Tool: - CVS- Subversion. This will cause the Source Control tab's options to change according to the selected source control software.
3. Define the, general settings, messages and other settings for the selected source control software and select Apply to set the selected source control software as your user default.
4. Select OK to close the Preferences menu.

The following menus show the selected source control software as the default option:

- Tools menu
- Editor right-click menu

Note:

To change the source control default, return to the Source Control tab and select a different software type from the list. Redo steps 1-4 to configure the source control software.

Using Source Control DIFF

Zend Studio's DIFF viewer allows you to view comparisons of recently saved versions of a PHP file with a version of the file held in the version control repository.

The DIFF viewer supports both supported source control software, CVS and Subversion.

The DIFF viewer's appearance can be customized to reflect user-defined colors to indicate the DIFF categories: Changed, Appended and Deleted.

To customize the DIFF viewer's appearance:

1. Go to Tools | Preferences | Colors & Fonts.
2. Choose a Scheme or select a scheme on which to base your new scheme.
3. Go to the Diff viewer tab and define the element setting preferences.
4. If you are creating a new scheme, select Save As and enter the new color scheme's name.
5. If you are changing an existing scheme's settings, select Apply and OK to close the preferences menu.

To perform a DIFF comparison:

1. Edit the PHP file you are working on and save the changes.
2. Right-click to open the available menus.
3. From the version control menu, select DIFF.

The DIFF viewer opens showing the Repository version of the file on the left, and the Current version of the file on the right.

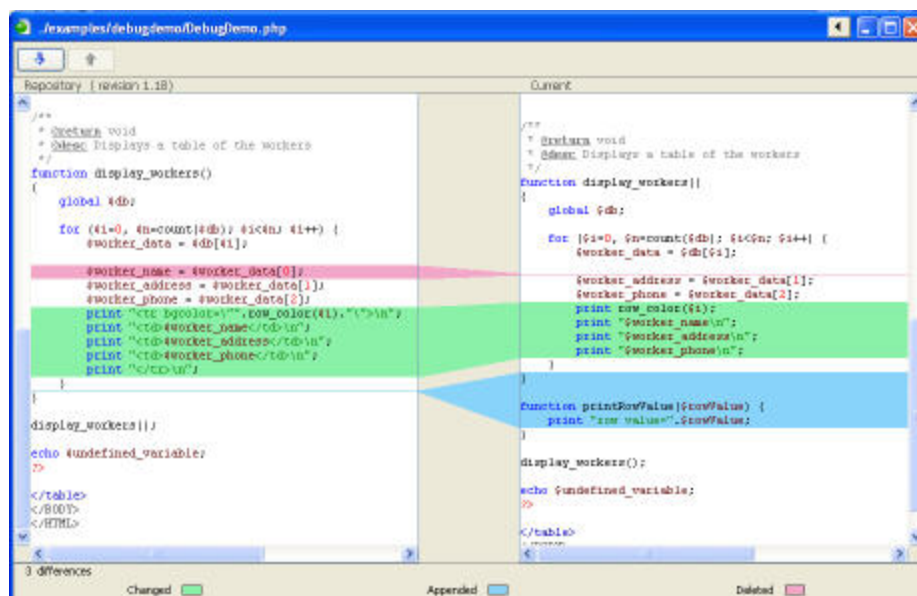


Figure: 37 - Source Control Diff Viewer

The DIFF viewer supports two important features that allow you to identify and navigate to differences between the versions you are comparing.

- **Smart Scrolling** - the application scrolls automatically between DIFFs , i.e., sections of code that have changed
- **View Changes** - the application attempts to align similar unchanged lines

The DIFF viewer supports the following navigational and informative features:

Setting	Description
Arrows	Navigate to the Next and Previous DIFF within the same window
Colors	User-defined color scheme for describing a DIFF (Changed, Appended, Deleted)
Statistical	Number of differences in the version
Revision	Revision number

Configuring Zend Studio for CVS

To configure Zend Studio for CVS, follow these steps:

If you have already checked out a module:

1. Open a Project.
2. Add the file/directories to the Project.
3. Perform CVS actions from the Project tab.

If you have not checked out a module:

1. Go to Tools | CVS | Checkout.
2. Enter the details in the Checkout dialog and click OK.
3. Make sure that the Checkout operation was successful.
4. Open a Project.
5. Add the files/directories to the Project.
6. In the Project tab, perform the necessary CVS actions.

To display the following CVS commands, right-click on a Project file/directory and select CVS. Perform this operation either from the Project Manager window, or in the Editing window.

Setting	Description
Update	Get the most recent copy of the file from the CVS repository, and merges it with the local version.
Commit	Commit changes to the CVS server repository.
Add	Specifies files to be added to the CVS repository. The files will be added only after you commit them.
Status	Show the current status of the files as defined in the CVS server.
Diff	Compares between your local copy of the file and the one located on the CVS server repository, and lists the difference between them.
Log	Prints out a log containing all the information relevant to the file or the directory, as appears in the CVS.
Checkout	Checkout a new module on your disk. You can Checkout a module using the File System window as well as the Project system.

The output any of the above actions appears in the CVS Output tab of the Messages window.

Configuring the Zend Studio - CVS Communication Tunnel

Zend Studio allows you to use a communication tunnel between Studio's integrated CVS client and the CVS server over SSH. This is accomplished by configuring the CVS_RSH environment variable to SSH.

To configure the CVS_RSH environment variable:

1. Open the CVS tab in Preferences: Tools | Preferences | Source Control.
2. Select CVS as the source control tool.
3. In the CVS_RSH Environment Variable value field, enter: 'SSH' and click Apply.

Studio's integrated CVS client and the CVS server will now communicate over SSH.

Note:

The configuration procedure described above toggles the communication tunnel on. To toggle the communication tunnel off, erase 'SSH' from the CVS_RSH Environment Variable value field and click Apply.

Configuring Zend Studio for Subversion

To configure Zend Studio for Subversion, follow these steps:

If you have already checked out a module:

1. Open a Project.
2. Add the file/directories to the Project.
3. Perform Subversion actions from the Project tab.

If you have not checked out a module:

1. Go to Tools | Subversion | Checkout.
2. Enter the details in the Checkout dialog and click OK.
3. Make sure that the Checkout operation was successful.
4. Open a Project.
5. Add the files/directories to the Project.
6. In the Project tab, perform the necessary Subversion actions.

To display the following Subversion commands, click the right mouse on a Project file/directory and select Subversion. Perform this operation either from the Project Manager window, or in the Editing window.

Command	Description
Update	Get most recent copy of the file from the Subversion repository, and merges it with the local version.
Commit	Commit changes to the Subversion server repository.
Add	Specifies files to be added to the Subversion repository. The files will be added only after you commit them.
Delete	Immediately Delete files or links from your working copy. If it is a directory, it is not deleted, but Subversion schedules it for deletion. When you commit your changes, the directory will be removed from your working copy and the repository.
Revert	Reverts the file to its pre-modified state by overwriting it with the cached "pristine" copy from the .svn area
Resolve	Once conflicts between files have been resolved, use Resolve to let Subversion know. This removes three temporary files (filename.mine filename.OLDREV filename.NEWREV) and Subversion no longer considers the file to be in a state of conflict
Status	Show the current status of the files as defined in the Subversion server.
Diff	Compares between your local copy of the file and the one located on the Subversion server repository, and lists the difference between them.
Log	Prints out a log containing all the information relevant to the file or the directory, as appears in the Subversion.
Checkout	Checkout a new module on your disk. You can Checkout a module using the File System window as well as the Project system.

The output any of the above actions appears in the Subversion Output tab of the Messages window.

Source Control File Status

The **Source Control File Status** feature provides visual representation of a file's status in both **CVS** and in **Subversion** (either one [*but only one*] can be active). The status of each file is represented by a different color in Studio's Project Explorer.

Source Control File Status applies to files that are:

- Newly added.
- Modified.
- Merged with conflicts.
- Not under the Version Control System.
- Up-To-Date (unmodified)

By default, Source Control File Status highlighting is enabled for both **Subversion** and **CVS** sources; the highlighting default colors for each of the products are different.

Configuring Source Control File Status

1. Open Studio's Preferences (click **Tools | Preferences**).
2. Select the **Source Control** tab.
3. Go to the section named **File Status Decorations**.

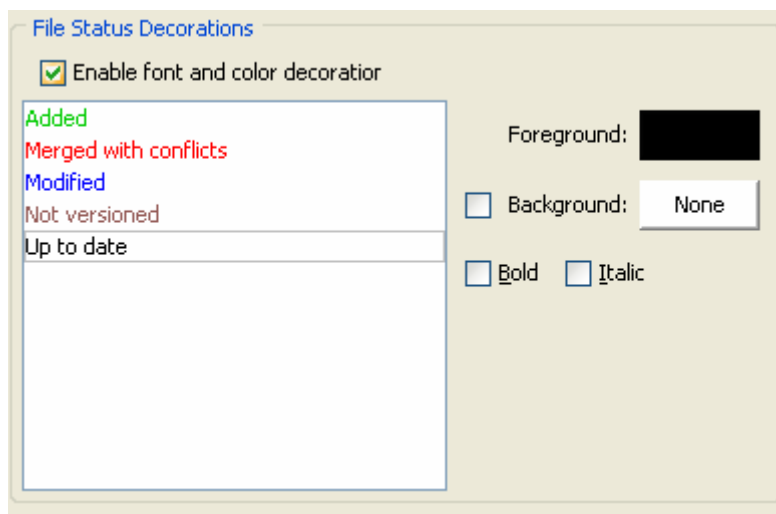


Figure: 38 - File Status

4. Enable/Disable the Source Control File Status as required.
5. Enable/Disable the Background color(s) as required (foreground color cannot be disabled).
6. Change the Foreground / Background color(s) by clicking the respective color rectangle. A dialog to choose color will appear.
7. Change the desired parameter(s) as required.
8. Click **Apply** / **OK**.
9. The default settings are as follows:

- Newly Added - Green
- Modified - Blue
- Merged with conflicts - Red
- Not under the Version Control System - Brown
- Up to date (unmodified) - Black

Chapter 13 - SQL Support

IN THIS CHAPTER...

SQL SETTINGS

FILE MANAGER: SQL

MAIN WORKSPACE - DATA DISPLAY

SQL QUERY CONTROL

Zend Studio's SQL Support creates a convenient interface between Studio's development environment and SQL databases. The SQL feature set will be of enormous help to developers writing business and database applications and/or working in the team environment.

About

Zend Studio's SQL Tool allows you to add and/or configure a connection to an SQL server. This feature set then allows you to view the structure of an SQL database from within the Studio development environment. You can view or edit the table data or procedures stored in the database. Alternatively, if your PHP application is designed to create a new SQL relational database, Studio's SQL Tool enables you to view the new database and its contents. Most importantly, you can do all this without having to rely on 3rd-party SQL utilities.

List of Functions

SQL Support, provides the following functions:

- **SQL Server Configuration** - Allows you to configure the SQL Server Settings
- **SQL Server Tree** - Allows you to view the database structure that is composed with schemas, tables, stored procedures, indices etc.
- **SQL Query Functions** - Allows you to run SQL query (statements) on a selected location.
- **Messages** - Returns an error message if the query script fails to execute properly.
- **Results (Edit mode)** - Allows you to edit the contents of the table.
- **Text Viewer, Hex Viewer and Image Viewer** - Allows you to view the contents of a table cell that contains textual data or binary data in a form of Blobs and Clobs.

SQL Settings

Zend Studio allows you to add an SQL server to the development environment and to configure the server settings.

Zend Studio supports the most popular SQL servers, with support for additional servers forthcoming. Currently, the following databases are supported: Oracle, MySQL, SQLite, Microsoft SQL Server, PostgreSQL, IBM Cloudscape and IBM DB2.

To add an SQL Server to Studio:

1. From Studio's File Manager, click the SQL tab.
2. Right-click to open the right-click menu.
3. Select the Add SQL Server option to open the Add SQL Server dialog.

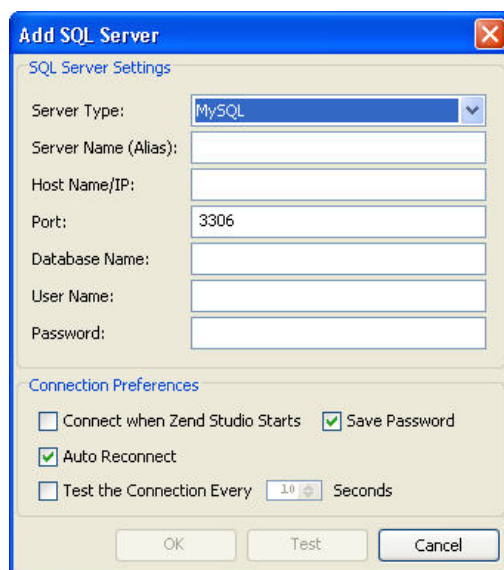


Figure: 39 - Add SQL Server

4. Enter the SQL Server Settings in the fields.

The settings include:

- a. **Server Type** - A list of Server Types supported in the current version of Studio.
- b. **Server Name (Alias)** - The name of the server that will appear on the SQL Tree (of the File Manager).
- c. **Host Name/IP** - Address of the server.
- d. **Port** - When you choose a Server Type, the default port appears in this field automatically.
- e. **Database Name** - The name of the database that the server connects to. This field is named as SID in Oracle's connections and as Database File in SQLite "flat-file" database connections.
- f. **User Name** - Your approved User Name for the server.

- g. **Password** - The correct Password for the User Name.
5. Enter the Connection Preferences using the checkboxes provided.

The preferences include:

- a. **Save Password** - Saves the Password.
 - b. **Connect When Zend Studio Starts** - Connects to the server whenever Zend Studio starts.
 - c. **Auto Reconnect** - If the connection to the server is broken, Auto Reconnect attempts 3 times to reconnect. If it is unable to reconnect, it then registers a "disconnect" and Auto Reconnect will stop working.
 - d. **Test the Connection Every** - Checks every n seconds to see if the server is responsive. Use the Seconds field to enter your preference.
6. Click Test to attempt to connect to the SQL Server using the settings and connections currently entered in the Add SQL Server dialog box.
7. Click OK to accept the settings and preferences you have entered. Studio records your settings and automatically tests the connection to the server.

Note:

If you made a mistake in either the User Name or Password, e.g., if the server does not recognize them as being valid, you will receive a pop-up that allows you to attempt to re-enter the information.

Editing Server Settings

To edit Server Settings for an SQL Server:

1. From Studio's File Manager, click the SQL tab.
2. From the list of SQL Servers currently defined for Studio, select a server, right-click and select the Settings option. The SQL Server Settings dialog box opens for that server.
3. Enter the SQL Server Settings in the fields provided.

The settings include:

- a. **Server Type** – When updating a server the list of server types is disabled (Create a new server definitions if the server type needs to be changed).
 - b. **Server Name (Alias)** - The name of the server that will appear on the SQL Tree (of the File Manager).
 - c. **Host Name/IP** - Address of the server.
 - d. **Port** - The default port can be manually edited.
 - e. **Database Name** - The name of the database that the server connects to. This field is named as SID in Oracle's connections and as Database File in SQLite "flat-file" database connections.
 - f. **User Name** - Your approved User Name for the server.
 - g. **Password** - The correct Password for the User Name.
4. Enter the Connection Preferences using the checkboxes provided.

The preferences include:

- a. **Save Password** - Saves the Password.
 - b. **Connect When Zend Studio Starts** - Connects to the server whenever Zend Studio starts.
 - c. **Auto Reconnect** - If the connection to the server is broken, Auto Reconnect attempts 3 times to reconnect. If it is unable to reconnect, it then registers a "disconnect" and Auto Reconnect will stop working.
 - d. **Test the Connection Every** - Checks every n seconds to see if the server is responsive. Use the Seconds field to enter your preference.
5. Click Test to attempt to connect to the SQL Server using the settings and connections currently entered in the Add SQL Server dialog box.
 6. Click OK to accept the settings and preferences you have entered.

Connecting to an SQL Server

This section describes the procedure for connecting to an SQL Server that you have not configured to "Connect When Zend Studio Starts". Unconnected Servers appear in red in the SQL Servers Tree.

To connect to an SQL Server:

1. From the Server Tree of the SQL tab, double-click the Server you want to connect to. Studio connects to the selected server.
2. Alternatively, select the server you wish to connect to and right-click to open the server menu.
3. Select Connect to Server and click. Studio will attempt to connect to the selected server.

File Manager: SQL

Zend Studio allows you to view the database structure of an SQL server.

To view the database structure of an SQL server:

1. Open the SQL tab of the File Manager. The SQL server tree displays a list of all SQL servers currently listed.

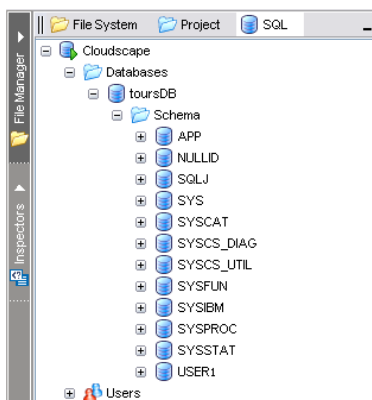


Figure: 40 - SQL Server Tree

2. Click the server whose database structure you wish to view.
3. Expand the server node to see a list of the databases stored on that server.

Viewing the Schema Structure of the Database

Some of the supported servers display schemas as part of their relational model.

To view the schema structure of a database stored on an SQL server:

1. From the SQL tab, click the server that contains the database whose schema you wish to view.
2. Expand Databases. Studio displays a list of all databases currently stored on the selected server.
3. Double-click the database whose schema structure you wish to view. Studio displays the schema structure for the selected database.

Main Workspace: Data Display

Zend Studio allows you to view data stored in a database table. The contents of the database table are displayed in table form in the Data Display area of the SQL view.

About the Data Display

Studio's procedure for viewing the contents of a database table is an assisted query of the SQL database. When you select a table from the SQL tab's database tree, Studio executes an SQL query that returns the contents of the table. How much of the table data you see in the Data Display depends on the display parameters you define.

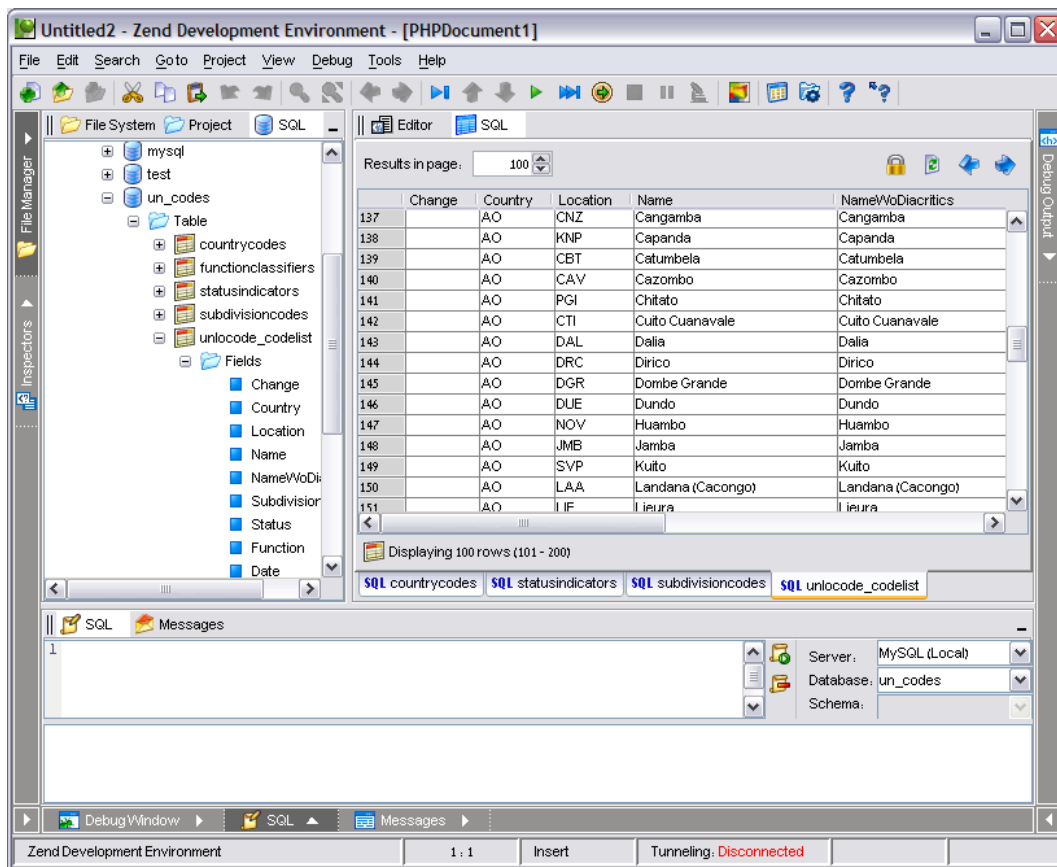


Figure: 41 - SQL Data Display

Functions, Shortcuts and the Right-click Menu

Right clicking on any of the tree elements in the SQL file tree opens a menu of functions that include:

- **Refresh** - Refreshes the SQL tree display. The refreshed elements are the ones that lie inside the refreshed node.
- **Query** - Displays the SQL query editor.
- **Metadata** - Displays the selected element's metadata.
- **Table Data** - Displays the contents of a table.
- **Global Settings** - Displays the Studio preferences for the SQL tool.

Viewing the Contents of a Table

To view the contents of a database table:

1. From the SQL tab of the File Manager, select the server that contains the database table whose contents you wish to view.
2. Expand the server tree until you can see the complete schema structure of the database.
3. Select the database table whose contents you wish to view.
4. To display the data contained in the table, do one of the following:
 - a. Double-click the table; or
 - b. Select the table and click Enter; or
 - c. Right-click the table and select Show Table Data from the menu.

Note:

The first two options' behavior is controlled by the "Global Settings" of the SQL tool.

5. In the Results in Page selector at the top of the Data Display, select the number of results you wish to display per screen.

The Data Display will show the contents of the table according to the selected resolution.


Data Display

Zend Studio allows you to view, edit and/or sort the data currently displayed onscreen.

The contents of a table cell can be viewed and edited depending according to the type of content and the mode (edit mode or view mode).

View Mode:

View mode is the default mode that appears when going an element from the SQL tree and selecting "Table Data" from the right-click menu. The Table Data is automatically displayed in the Editor area as a new tab called SQL.

While in view mode the Lock/Unlock button is Yellow  and the data display will change according to the data type as follows:

- **Textual Elements** - Double-clicking on a textual element will open the element in the embedded text viewer.
- **Numbers** - no additional options for viewing numbers.
- **Blobs (Binary Large Objects) and Clobs (Character Large Objects)** - Double-clicking on Blobs or Clobs will open the Hex editor by default. The right-click menu provides additional viewing options: Plain text, Hex and Image (for Blobs only).

Edit Mode:

Edit Mode is reached from the View mode stage by enabling the Edit option. When enabled, table data can be edited in different ways according the data type.

In Edit mode the Lock/Unlock button is Green  and the data display will change according to the data type as follows:

- **Textual Elements** - can be edited directly in the field display or double-click to open textual element and edit it in the embedded text viewer.
- **Numbers** – can be edited directly in the field display.
- **Blobs (Binary Large Objects) and Clobs (Character Large Objects)** – Cannot be edited
- **Date and Time Formats** – are edited in embedded date and time pickers according to the number format.

Sorting Visible Data

Studio allows you to re-order the data that is currently shown in the Data Display. To sort the visible data, click the active column headings.

The following sort options are supported:

- **Original order** - this is the unsorted order of the data as the SQL server returned it.
- **Ascending** - click an active column heading; the column sorts in ascending alphabetical order.
- **Descending** - click the same column heading a second time; the column sorts in descending alphabetical order.

- **Return to original order** - click the same column a third time; the column returns to the original order as returned by the SQL server.
- **Secondary sort** - CTRL + Click a column; Studio performs a secondary sort of associated data within the selected column.

Data Display: Large Number of Results

Zend Studio supports a variety of functions for managing queries that return a large number of results.

Setting the Number of Results Displayed on a Page

Studio allows you to define the number of results displayed per online page. This feature is helpful in viewing large table contents.

To set the number of results in a page:

1. In the Results in Page selector at the top of the Data Display, select the number of results you wish to display per page. (Acceptable values range from 1 to 3,000).
2. The Data Display will show the contents of the table according to the selected resolution.

Navigating to Results Not Shown in the Initial Display

Studio displays the results of the structured query according to the user-defined 'Results in Page' value. However, when the returned results number exceeds the page limit, not all the results will be shown in the initial onscreen display table. In this case, Next / Previous arrows are enabled to navigate to the next or to the previous page.

Refreshing Displayed Results

Studio allows you to refresh the displayed results by clicking the Refresh button at the top of the Data Display area. By doing so, you will get the most updated data from the server and you will be able to view any changes made by the embedded query editor or ones that were made externally.

Clicking the Refresh button sends the same query to the server and then updates the data display with the fresh results.

Organizational Objects

Studio allows you to view the structure of the SQL database. Specifically, you can expand the tree to display the organizational objects, e.g., tables, fields, etc. These organizational objects make up the database elements.

Viewing Table Organizational Objects

In the illustration below, the organizational elements of the SQL table "columns_priv" are represented in the SQL tree.

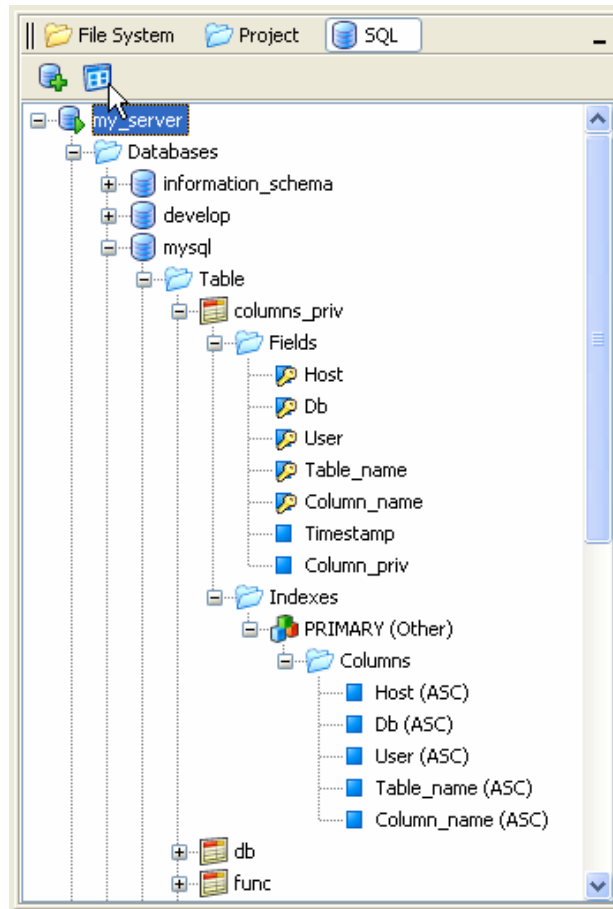


Figure: 42 - SQL Tree

The illustration can be understood as follows:

- Each entry in the Fields organizational element corresponds to a column in the database table.
- A Field entry that is marked with a key image corresponds to a Primary key defined on that table column. For example, see the field "Host".

Viewing Table Indexes

Studio allows you to view the indexes defined for an SQL table. Specifically, you can expand Table elements to view indexes that were predefined.

In Figure: 42 - above, there are currently five indexes defined for the "columns_priv" table. These correspond to the columns "Host", "Db", "User", "Table_name", and "Colum_name."

An Index is displayed by its name and is followed by its type in parentheses.

The Index type can be Clustered, Hushed or Other. An Index Column has an ASC or DECS that defines its sorting order as Ascending or Descending.

Stored Procedures

A stored procedure is a set of pre-defined Transact-SQL statements used to perform a specific task. There can be multiple statements in a stored procedure; all the multiple statements are clubbed into one database object.

The basic building blocks of a stored procedure include:

- A CREATE PROC (CREATE PROCEDURE) statement
- The procedure name
- The parameter list
- The SQL statements

Browsing in a Stored Procedure

Studio allows you to browse in a stored procedure directly from the tree structure. Specifically, you can expand the collapsed stored procedure to reach its components.

In the illustration below, you can see the stored procedures in the un-collapsed Stored Procedure folder in the database. The procedure "SQLCOLUMNS" is un-collapsed, allowing you to view its parameters and return values.

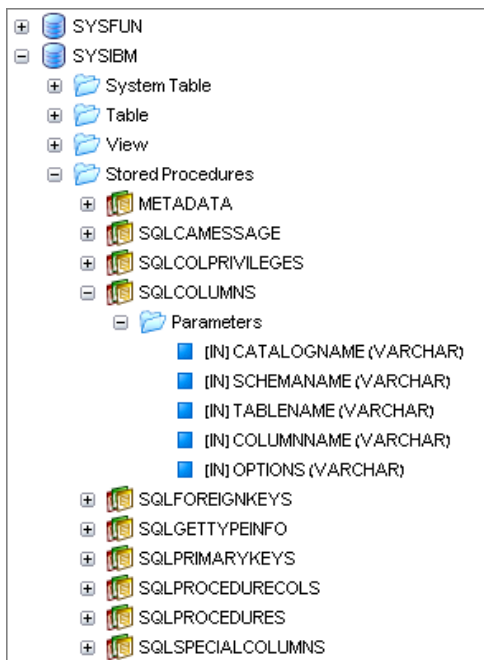


Figure: 43 - SQL Tree

Server Metadata

SQL Server stores information about databases and database objects-the metadata-in system tables within databases. Studio's SQL Support allows you to view the metadata tables.

The metadata you can view includes:

- Server metadata
- Schema metadata Table metadata

To view metadata for a particular server:

1. In the SQL Servers tree, select the server whose metadata you wish to view.
2. Right-click to open a menu of functions.
3. Select Metadata to view Metadata for the selected server onscreen.

Note:

There are three tabs that make up the Metadata: Status, Variables, and Process List. However, the content and quantity of tabs depends on the type of server and the specific user's authorizations. If the user does not have the appropriate authorizations a message will appear in the Messages window.

Using the Kill Process

Some servers support a "Kill Process" function, which allows you to stop a specific process currently running on the server.

To use the Kill Process function:

1. In the Metadata screen dialog, open the Process List tab by selecting a server and selecting the Metadata option from the right-click menu.
2. Select the process you wish to stop.
3. Right-click to open a menu of functions.
4. Select Kill Process to stop the selected process.

Note:

If the selected server does not support the Kill Process function, Kill Process will be grayed out on the Right-click menu.

Schema Metadata

To view metadata for a database's schema (assuming that the database supports SQL schemas):

1. In the SQL Servers tree, select the database node whose metadata you wish to view.
2. Right-click to open a menu of functions.
3. Select Metadata to display the database node's Metadata onscreen.

Table Metadata

To view the metadata for a database table:

1. In the SQL Servers tree, select the table whose metadata you wish to view.
2. Right-click to open a menu of functions.
3. Select Metadata to display the Metadata for the database table onscreen.

Field Metadata

To view table cell metadata:

1. In the SQL Servers tree, select the table cell whose metadata you wish to view.
2. Right-click to open a menu of functions.
3. Select Metadata to display the selected cells Metadata onscreen.

Note:


The field metadata presentation is a filtered presentation of the table's metadata.

Editing the Contents of a Table

Studio allows users with the correct permissions to edit the content of an SQL database table.

Not everyone has permission to edit the content of the database table!

To edit a table the following conditions must be met:

- The table must be accessed from the SQL Tree (not from an SQL query).
- The table must have at least one primary key.
- The display must be changed to edit mode .

If one of these conditions is not met, a message in the SQL Messages window will appear detailing which condition was not met.

SQL Query results displayed in the Editor's SQL display are not editable.

Note:

The primary key of a table is the column whose values are different in every row. Because they are different, they make each row unique. By extension, this makes it possible to edit individual rows without impacting the entire table.

Primary Keys

If the table contains a Primary Key (PRI), the field node will be marked with a key icon in the Table tree. Therefore, you can identify an editable table from the Table tree by the key icon marking it.

When data is displayed from the SQL tree, columns that are defined as primary keys will display the letters [PRI] along with their names.

Unlock Function

The Unlock function , unlocks the table for editing.

In order to unlock the table the following conditions must be met:

- You must have permission to edit the table.
- You must view the data via a table selected from the SQL tree.
- The table must have at least one primary key.

Setting the Table Cell to a Null Value

SQL uses a special value, called the null value, to mean that a particular value is not known. As part of the editing procedure, Studio allows you to set a table cell as a null value.

To set a table cell to a null value:

1. Verify that the table is unlocked.
2. Select the table cell you wish to set to a null value.
3. Right-click to open the Edit menu; then select Set to Null. Alternatively, type (null) in the cell.
4. Click Enter to update the selected cell in the database to contain a null value.

Editing a Table Field

There are two methods for editing a table field:

- Editing the field in the table itself
- Using the Edit dialog

To edit the field in the table itself:

1. Verify the table is unlocked.
2. Double-click the cell you wish to edit.
3. Enter your changes
4. Exit the cell, i.e., record the changes, by clicking outside the cell area.

To edit a table field from the Edit dialog:

1. Verify the table is unlocked.
2. Double-click the cell you wish to edit.
3. Select the Edit button that appears to the right of the cell. The Edit dialog opens displaying the cell's contents.
4. Enter your changes.
5. Click OK to record the changes to the database.

Note:

The Edit dialog is enabled only for cells with textual data. Also, note that line breaks can be inserted only when editing the data with the Edit dialog.

Data Display: Large Objects

Studio supports multiple view options for the SQL-99 data types "Blobs" and "Clobs".

There are three embedded viewers: Plain Text, Hex, and Image.

"Blobs" (Binary Large Object) and "Clobs" (Character Large Object) are large data types supported in SQL. These are of special interest to developers of web applications (PHP)- particularly where images or other large data files are going to be recovered from the database.

For example, in a web site designed for a Real Estate company, you may want to show high-resolution pictures of houses for sale. These image files can be huge. SQL allows for efficient storage of images in the database - Blobs and Clobs - in a way that makes it possible for the application or user to access them efficiently. This impacts directly on the user experience of the web site.

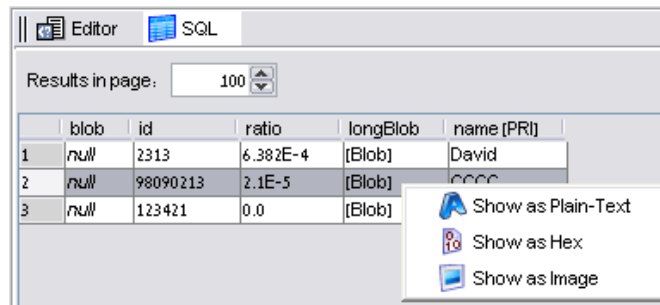
View Options: Plain Text, Hex, and Image

Studio provides three embedded viewers for Blob/Clob data types:

1. **Plain-Text** - for viewing large character files.
2. **Hex** - view file binary data as if you were looking at a common hex viewer. Bytes are displayed as hex characters-character data is displayed on the right. Starting positions of each row are displayed on the left. The Hex viewer allows you to select a number from the Hex display. The selection appears below the main display area. There are also several display options for viewing the selected number:
 - a. **Little Endian** - low-order byte of the number is stored in memory at the lowest address, and the high-order byte is stored at the highest address.
 - b. **Big Endian** - high-order byte of the number is stored in the lowest address, and the low-order byte is stored at the highest address.
 - c. **Decimal** - displays the decimal value of the number.
 - d. **Hex** - displays the number in hex characters.
3. **Image** - for viewing the data as an image (GIF and JPEG are supported).

To view Blob / Clob data:

1. In the query results table, verify that the data you wish to view is a marked with [Blob] or [Clob].
2. Select the cell that you wish to view.
3. Double-click the selected cell to view the data in the embedded Hex viewer. Alternatively, right-click to select from the menu of supported viewers.



Results in page: 100

	blob	id	ratio	longBlob	name (PRI)
1	null	2313	6.382E-4	[Blob]	David
2	null	98090213	2.1E-5	[Blob]	cccc
3	null	123421	0.0	[Blob]	

Context Menu Options:

- Show as Plain-Text
- Show as Hex
- Show as Image

Figure: 44 - Query Results Table

- Then choose to show the cell data as Plain, Text, Hex, or Image.

SQL Query Control

Studio includes a number of functions that aid in running SQL queries. These are found in the SQL Query area at the bottom of the Studio workspace.

These include two displays:

1. **Query Area** - displays the query currently edited; includes an apparatus for selecting a Server, Database, and Schema; also includes function buttons for controlling the query. When navigating through the tree the Server, Database and Schema fields will change according to the selected Server's structure.
2. **History of Queries** - displays a history of queries run.

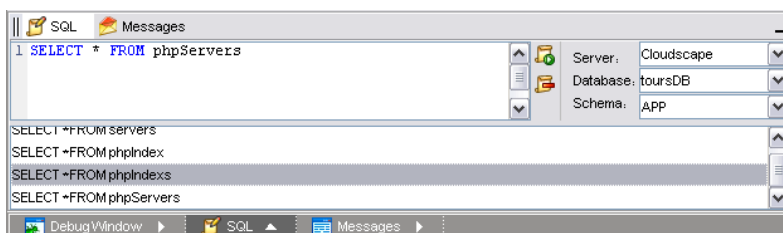



Figure: 45 - SQL Query Control

To run an SQL query:

1. In the query area, enter the query you wish to run using correct SQL syntax.
2. Use the controls to the right of the query area to select the Server, Database, and Schema (if applicable) you wish to query.
3. Select  or press CTRL+ Enter. The query runs on the selected location and displays the results in the query area. Queries that do not return result sets, like Update and Insert, will display the number of affected rows in the SQL Messages tab.
4. Query history can also be accessed by selecting a query and executing it from the right-click menu, and pressing the 'Go' option or by pressing CTRL+ Enter.

Note:

Double-clicking a history item will place the history query text into the query editor area for editing and re-executing.

Controls: Server, Database, and Schema

Studio provides controls for defining the target Server, Database, and Schema.

These controls are located to the right of the query area.

Use these controls to select the Server, Database, and Schema (if applicable) you wish to query.



Before running a query:

- Use the display fields of the three controls to verify that you running the query on the correct server.
- When traversing the SQL tree the query control values will automatically be set to the values of the current location in the tree.

Functions: Go and Clear

Studio provides two function buttons for running an SQL query.

The function buttons are located to the left of the query area.

- Use  to run the query on a selected location.
- Use  to clear the query area.

Re-running a Query from the History Area

To run a query from the history area:

1. Select a history element from the history display.
2. Right-click to open the right-click menu.

The menu includes:

- a. **Go** - re-run the query immediately, from the history area
- b. **Remove** - removes the query from the history area
- c. **Remove All** - removes all history elements from the history area

3. Click Go to re-run the SQL query.

(See below for an Alternative Method for Re-running a Query)

Note:

History elements currently in the history display are saved when you exit Studio, and restored when you start Studio. The number of history elements saved by the system is defined in "# of History Elements Saved" parameter of the Global Settings tab. Default=100. (See Global Settings) .

Alternative Method

The following instructions describe how to use an alternative method for re-running a query:

1. Select a history element from the history display.
2. Double-click the selected element. Studio moves the selected element from

the history display to the Query Editor.

3. Click the Go icon inside the Query Editor. The SQL Query re-runs.

SQL Messages

Studio's SQL support includes a message area. In the event of an error in executing a query script, the message area displays a message reporting the error. In cases where a query type of Insert, Update or Delete was made, any information regarding the number of affected rows will appear in the messages area.

Error Message Example: the example below shows a query has been run for the 'PHPSERVERS' database object.

No such object exists. Consequently, the message area has become active and displays a message reporting the error.

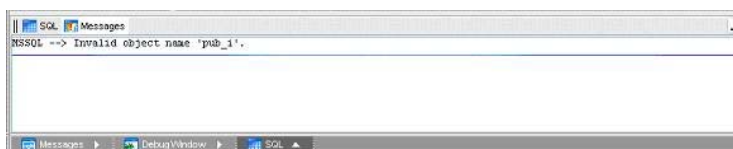


Figure: 46 - SQL Error Messages

Chapter 14 - Preferences

IN THIS CHAPTER...

DESKTOP PREFERENCES
EDITING PREFERENCES
CODE COMPLETION PREFERENCES
DEBUG PREFERENCES
COLORS AND FONTS
KEYMAP CONFIGURATION
FILE TYPE PREFERENCES
TEMPLATES
SOURCE CONTROL PREFERENCES
DIALOGS PREFERENCES
GLOBAL SQL SETTINGS

Zend Studio allows you to control and customize a variety of settings and options. These options include general settings that apply to the desktop, functions, navigational tools, onscreen appearance, feature settings, etc.

The Preferences window is accessed from the main menu: Tools | Preferences.

Note:

Restore all default settings by right clicking in the Preferences Window and selecting Restore all Defaults from the shortcut menu. This will restore the active tabs defaults.

The Preferences Window is divided into tabs representing different feature's preferences as follows:

- **Desktop** - Contains options, which allow customizing the desktop icons, fonts, background colors, language and more.
- **Editing** - Contains customizable settings and options, which effect editing tools and appearance.
- **Code Completion** - Contains settings for content and control of completion windows.
- **Colors & Fonts** - Contains color assignments for Syntax Highlighting (General, PHP, HTML, and Javascript) and for Highlighting DIFF elements.
- **Debug** - Contains settings and customization features for the debugging process.
- **Keymap** - Contains settings for customizable shortcuts.
- **File Types** - Enables customization of the list of file types and associated file extensions.
- **Templates** - Allows you to Add, Edit or Remove templates.
- **Source Control Options:**
 - CVS** - Enables CVS integration for version control management.
 - Subversion** - Enables Subversion integration for version control management.
- **Dialogs** – show or hide the various Studio dialogs.

- **SQL** - Contains global settings for SQL Support.

Setting Desktop Preferences

Desktop preferences are accessed from the Preferences menu (Tools | Preferences). These preferences define the Zend Studio's system look and feel.

The following Preferences can be set:

Property	Description
Use OS Look and Feel	When enabled, Zend Studio will appear in a format similar to the OS in which it is installed, e.g. Windows XP, or Mac (Linux users do not have this option).
Language	Choose a system language for the display and menus: English, German, French, Dutch, or Korean
Font	Defines the font, size and style for the Desktop and menu text.
Encoding	Setting for type of Encoding. Choose an option from the list or manually define an encoding type by entering the new name into the Encoding field and pressing Enter.
PHP Version	Sets default version of PHP; this affects the internal debugger used, code analyzer and code completion.
Support Asp Tags	Enables/Disables the automatic PHP/HTML Code Completion to respond to ASP tags in the same way as it responds to PHP tags.
Use Internal Browser	Enables/Disables the internal browser. Disabling the browser will remove the Browser tab from the User Interface.
Browser Activation Command	Command for activating your preferred Browser, as if it were activating from a command line.
PHP Manual URL	Defines the path of the PHP online manual.
Zend Guard Path	Defines the path to Zend Guard.
Show Hidden Files and Folders	Enables/Disables the Show Hidden Files and Folders function.
Check for External File Modification	When enabled, Studio checks the file system to see whether open files were externally edited. Note: this does not check files externally modified over FTP.
External Modification Check Frequency (ms)	Defines the period in ms between checks for External File Modification; default=1000.
Automatically Reload Externally Modified Files	Enables/Disables the automatic reloading of externally modified files. Controls the method in which files that have been changed outside of the Zend Studio will reload.
Maximum Entries in Search Results Display	Defines the maximum number of entries that are displayed in the Search Results window.

Setting Editing Preferences

Editing preferences are accessed from the Preferences menu (Tools | Preferences). These preferences define the editor's behavior.

The following Preferences can be set:

Property	Description
Code Style	
Tab Size	Assigns the size of a tab in spaces. If Tabs as Spaces is selected, this will be the input number of space characters for one TAB keystroke.
Tab as Spaces	Controls the actual character input as a result of entering a TAB. When selected, one tab keyboard input will not input a tab character but will result in the equivalent assigned number of spaces (space characters). When not selected the input of a tab will result a Tab character.
Auto Indentation	Enables/Disables the automatic indentation that occurs as you type code in the editing window.
Wrap Lines	Controls the manner in which the editing window displays lines of text greater in length than the editing window.
Wrap Words	Enables/Disables word wrapping in the editing window.
Strip Trailing Spaces on Save	Enables/Disables the strip trailing spaces on save feature
Show Print Margin at Column	Define the pages print margin according to printer definitions to make sure the code is inside the printed page.
Line Ending Style	Controls the ending style of the line. You can choose between Windows, Unix or As Is.
Smart Keys	
Smart Home	When Enabled, pressing on the Home key, moves the cursor to the first character in the line (ignores white spaces). Pressing Home again will bring the cursor to the beginning of the line. If Disabled normal Home key logic is applied and the cursor, jumps to the beginning of the line.
Auto Insert Pair Bracket	Enables/Disables the auto insertion feature.
Auto Insert Pair '}'	Enables/Disables the auto insertion feature.
Auto Insert Pair Quote	Enables/Disables the auto insertion feature.
Auto Insert HTML Closing Tag	Enables/Disables the auto insertion feature.
Auto Complete phpDoc Block	Enables/Disables the auto completion feature. When enabled entering /** and pressing enter will automatically create a phpDoc Block.
Generate phpDoc Stub	Enables/Disables the auto insertion feature. When enabled new phpDoc Blocks are generated with function parameters, return values, classes etc inside the phpDoc Block.
Highlighting	
Highlight Matched Bracket	Display of highlighting for quote-mark characters (beginning and ending), which are paired (matched) correctly. (){}[]<>
Highlight Matched Quote	Display of highlighting for enclosing characters (beginning and ending), which are paired (matched) correctly. ' ' " "
Highlight Active Line	Display of highlighting on the line, which is active (cursor line).

Property	Description
Matching Bracket Line tooltip	To view the header of an opening bracket, place the cursor of the ending bracket and a tooltip containing the header appears.
Real-Time Errors	Enables/Disables the Real-Time Errors function. Maximum Real-Time Errors - Number of errors that can be handled by Real-Time Errors function; Default= 100
Code Folding	
Enable Code Folding	Enables/Disables code folding for items selected in the preferences field: Fold by Default (below).
Fold by Default	- All Non PHP Code- PHPDoc Blocks- Classes- Functions

Setting Code Completion Preferences

Code Completion preferences are accessed from the Preferences menu (Tools | Preferences).

The following table lists and describes the properties contained in the Code Completion tab. Enable or disable any of the features, to customize Code Completion to your needs. In the Editing Window, Code Completion underlines any error made in the code and marks the entire line. To view a description of the error, place the cursor on the error marking and a tool tip appears.

The following Preferences can be set:

Property	Description
PHP Code Completion	
Automatically show Code Completion List	<p>Turns on the automatic pop-up of the PHP Code Completion window (affects only the starting function listing, pop-up will occur for parameters if the function is identified).</p> <p>You can condition the display by checking the following variables: Only if there are less than (# of) options - Conditions the display of the list by maximum number of options. A list containing more than the value entered, will not open automatically yet you can always display the list by pressing Ctrl-space.</p> <p>Popup after (# of) ms - Time value that affects the amount of time (in ms) between the entry of triggering text and the pop-up of the Code Completion window.</p> <p>Class Names - Auto pop-up of the Class Name Completion list of the available Classes. The Class Names Code Completion window appears after the new, extends or implements is entered in a PHP section of code.</p> <p>Variables - Turns on/off the Variable Names Code Completion. If disabled the Variable Names Code Completion window will not appear after the \$ character is entered in a PHP section of code.</p> <p>Functions, Keywords and Constants - This list includes member functions and variables, keywords and constants of the Classes declared in the active file. This completion lists, member names and function syntax.</p>
Show Variables from other files	<p>Checked - Variable Names Code Completion shows variables being used in all open and project files.</p> <p>Unchecked - Variable Names Code Completion shows only the variables being used in the active file.</p>
Determine Object Type from Other Files	Recognize Object types not only from the current file.
Disable Constants Completion	Enable/Disable Constant Code Completion.
Case Sensitive Completion for Constants	Enable/Disable case sensitive code completion. When disabled code completion will not differentiate between uppercase and lowercase items.
Show Class Names in Global Completion List	Enable/Disable the addition of class names to the code completion list. When enabled class names will be included in the code completion list.
Auto Popup Function Arguments	Turns on the automatic pop-up of the PHP Functions Arguments window.

Property	Description
Show Function Prototype in Arguments Completion	Shows full function prototype in Function Argument Window
Remove PHP Trailing Characters	Removes the characters to the right of the code completion insertion point.
Display Description tooltip after	Enable/Disable the Code Completion Description Windows and the amount of time (in ms) between the display of the Code Completion Window and the Description Window.
HTML Code Completion	
Automatically show Code Completion List	<p>Turns on or off the automatic pop-up of the HTML Code Completion window. You can condition the display by checking the following variables:</p> <p>Only if there are less than (# of) options - Conditions the display of the list by maximum number of options. A list containing more than the value entered, will not open automatically yet you can always display the list by pressing Ctrl-space.</p> <p>Popup after (# of) ms - Time value that affects the amount of time (in ms) between the entry of triggering text and the pop-up of the Code Completion window.</p> <p>Tags - Auto pop-up of the HTML Code Completion List of HTML tags.</p> <p>Attributes - Auto pop-up of the HTML Code Completion List of attributes for the tag.</p> <p>Attribute Values - Auto pop-up of the HTML Code Completion List of attributes' values for the tag.</p>
Remove HTML Trailing Characters	Removes the characters to the right of the code completion insertion point.
Always Complete HTML tags as Uppercase	Affects the inserted HTML code. When enabled, inserted HTML tags will be all uppercase characters.
Use XHTML Tag Style	Inserts typical XHTML tags.

Setting Debug Preferences

Debugger configurations are done from the Debug tab. The following table describes the user-configurable debugging features.

The following Preferences can be set:

Property	Description
Connection to Debug Server	
Debug Mode	Sets the Debugger toolbar settings to Internal or Server. (Server debugging - uses the PHP/Zend Engine and Zend Platform's Studio Server Component on the remote server. Internal debugging - uses the PHP/Zend Engine and internal debugger on the local drive.
Debugger Server URL	The IP or URL of the host that runs the Zend Debug Server. To check the connection to the debug server, select the Check Debug Server Connection, from the Debug menu. If the test fails, check the list of common problems as appears in the pop-up window.
Client IP	Sets the IP address of Studio's host machine.
Dummy File	This is a file, which is utilized by Zend Studio.
Client Debug Port	Sets the port number for communication with the Zend Debug Server
Broadcasting Port	The port for sending debug information for the Zend Toolbar and Zend Platform. This port number should be synchronized with the Toolbar and platform preferences.
Server Response Timeout	The amount of time allowed for a server response. If no response is received within this time, a notification will be generated to inform you that the Server is not responding.
Encrypt Communications using SSL	Enables encryption option.
Proxy Settings	Define HTTP user and authentication information when, encrypting communications using SSL.
Debug Options	
Temporary Output Files Location	Determines the temporary location of the files.
Show HTTP Header	Output of a HTTP content type header to the Output Window.
First Line Breakpoint at Debug URL	When enabled, sets an automatic Breakpoint in the first line of code. When disabled, will stop at the first breakpoint in the file.
Detailed Variable tooltip	Displays a serialized variable tooltip during a debug session.
Define Server Output Encoding	Choose an encoding type for debugger output encoding.
Messages	
Show Start End Messages	Controls the output of Show Start End Messages to the debug messages window during a debug session.
Show Notice Messages	Controls the output of Show Notice Messages to the debug messages window during a debug session.
Show Warning Messages	Controls the output of Show Warning Messages to the debug messages window during a debug session.

Property	Description
Show Error Messages	Controls the output of Show Error Messages to the debug messages window during a debug session.
Show Strict Messages	Controls the output of Show Strict Messages to the debug messages window during a debug session.
Windows	
Grab focus when Session Starts	When active the IDE will grab focus when a debug session begins.
Automatically Open Debug Window	Controls the automatic opening of the Debug window upon starting a debug session.
Automatically Open Debug Messages Window	Controls the automatic opening of the Debug Messages window upon starting a debug session.
Automatically Open Output Window	Controls the automatic opening of the Output window upon starting a debug session.

Colors and Fonts

Define and control the appearance of code in the editor. Define different colors for different languages and customize the views to suit personal preferences. Store preferences in a scheme or have several schemes if necessary.

To change a currently applied scheme:

1. Open the Colors & Fonts tab in Preferences: Tools | Preferences | Colors & Fonts.
2. Select a scheme from the Scheme Name dropdown menu.
3. Click Apply to apply the selected scheme to the active code displayed in the main workspace.

Note:

This scheme will be applied to the code display after restart, as well.

To create a new scheme:

1. Open the Colors & Fonts tab in Preferences: Tools | Preferences | Colors & Fonts.
2. Select an existing scheme.
3. Customize the appearance by navigating through the tabs and defining appearance preferences.
4. Select Save As and enter the new scheme's name into the Scheme Name field.

The customization options are as follows:

Option	Description
General	Define the general appearance of elements in the editor window.
PHP	Customize PHP colors and appearance.
HTML	Customize HTML colors and appearance.
Java Script	Customize Java Script colors and appearance.
CSS	Customize CSS and appearance.
XML	Customize XML colors and appearance.
SQL	Customize SQL colors and appearance.
Diff Viewer	Customize the Diff Viewer's appearance for Changed, Appended and Deleted code.

Colors and Fonts - General Tab

Define colors for the following elements:

- **Background** - Controls the background color of the editing window.
- **Matched Bracket/Quote Highlight Color** - Controls the color of the highlighting of matched quote-marks and enclosing characters in the editing window. ' ' " (){}[]<>
- **Mismatched Bracket Highlight Color** - Controls the color of the highlighting of enclosing characters that do not match. (){}[]<> ' ' " "
- **Search in Selected Text Highlight Color** - Controls the highlight color of text found in a selected text search.
- **Active Line Color** - Controls the highlight colors of the line in which the cursor is active.
- **Breakpoint** - Controls the highlight line color of code marked as breakpoints.
- **Debug Step** - Controls the highlight colors of the line in which the debug execution has stopped.

Colors and Fonts - Java Script Tab

Define formatting and colors for the following elements:

- **Java Script Tag** - Controls the highlight color of Java Script tags.
- **Java Script Text** - Controls the highlight color of Java Script text.
- **Reserved Word** - Controls the highlight color of Java Script reserved words.
- **Number** - Controls the highlight color of Java Script constants numbers.
- **Constant Quoted String** - Controls the highlight color of Java Script constants.
- **Java Script Comment** - Controls the highlight color of Java Script Comments.

Colors and Fonts - HTML Tab

Define formatting and colors for the following elements:

- **Comment** - Controls the highlight color of a comment.
- **Table Tags** - Controls the highlight color of HTML table tags.
- **Heading Tags** - Controls the highlight color of HTML heading tags.
- **All Other Tags** - Controls the highlight color of HTML tags other than table and heading tags.
- **Text** - Controls the highlight color of HTML text.
- **Attribute Value** - Controls the highlight color of HTML attribute values.
- **Syntax Error** - Controls the highlight color of HTML syntax errors.
- **Special Char** - Controls the highlight color of escape characters.

Colors and Fonts - PHP Tab

Define formatting and colors for the following elements:

- **PHP Tag** - Controls the highlight color of PHP tags.
- **PHP Text** - Controls the highlight color of PHP text.
- **PHPDoc** - controls the highlight color of PHPDoc comments.
- **Reserved Word** - Controls the highlight color of PHP reserved word.
- **Variable** - Controls the highlight color of PHP variables.
- **Number** - Controls the highlight color of PHP constants numbers.
- **Constant Quoted String** - Controls the highlight color of PHP constants.
- **Heredoc** - Controls the highlight color of PHP hHeredoc.
- **Comment** - Controls the highlight color of PHP comments.

Colors and Fonts - CSS Tab

Studio supports CSS Highlighting. This is the user-defined color scheme that assigns colors automatically when you open a file with the CSS extension. Through this preferences tab, you can define unique settings for CSS files.

Define formatting and colors for the following elements:

- **Comment** - Controls the highlight color of CSS comments
- **Constant Quoted String** - Controls the highlight color of CSS constants
- **Property** - Controls the highlight color of CSS property
- **Selector** - Controls the highlight color of CSS selector
- **Text** - Controls the highlight color of CSS text
- **Value** - Controls the highlight color of CSS values
- **CSS Tag** - Controls the highlight color of CSS tags
- **Reserved Word** - Controls the highlight color of CSS reserved words

Colors and Fonts - XML Tab

Studio supports XML Highlighting. This is the user-defined color scheme that assigns colors to XML elements within the XML code.

Define formatting and colors for the following elements:

- **Tag** - Controls the highlight color of XML tag
- **Tag Name** - Controls the highlight color of XML tag name
- **Attribute** - Controls the highlight color of XML attribute
- **Attribute Value** - Controls the highlight color of attribute value
- **Text** - Controls the highlight color of XML text
- **Reserved Word** - Controls the highlight color of XML reserved word
- **Comment** - Controls the highlight color of XML comments

Colors and Fonts - SQL Tab

Studio supports SQL Highlighting. This is the user-defined color scheme that assigns colors to SQL elements within the SQL code.

Define formatting and colors for the following elements:

- **Comment** - Controls the highlight color of SQL comments
- **Constant Quoted String** - Controls the highlight color of SQL constants
- **Number** - Controls the highlight color of numbers in SQL
- **Reserved Word** - Controls the highlight color of SQL reserved words
- **Text** - Controls the highlight color of SQL text

Colors and Fonts - DIFF Tab

Studio supports DIFF Highlighting. This is the user-defined color scheme that assigns colors to the different elements that have changed within to different versions of the same code.

Define formatting and colors for the following elements:

- **Changed** - Controls the highlight color of Changed lines
- **Appended** - Controls the highlight color of Appended lines
- **Deleted** - Controls the highlight color of Deleted lines

Configuring the KeyMap

There are three default KeyMaps containing keyboard shortcut settings for the main menu options. These KeyMaps reflect commonly known keyboard shortcuts as they appear in Visual Studio, Mac and Emacs. Selecting one of these options applies these keyboard shortcut configurations to the IDE.

New KeyMaps can be created based on one of the three default KeyMap options and subsequently new KeyMaps can be created based on user defined KeyMaps.

Customizing KeyMaps

You can use shortcut keys to activate commands from the keyboard rather than the menus and toolbars. You can also change which shortcut keys executes these commands.

To customize KeyMaps:

1. Open the Preferences Window from the Main Menu, select Tools | Preferences.
2. Select the KeyMap tab.
3. In the upper part of the KeyMap tab, select a KeyMap configuration. By default, Zend Studio contains three popular sets: Visual_Studio, Emacs and Mac. You can customize any of them or create a set of your own.
 - a. To customize one of the default sets, select a set and Activate.
 - b. To create a personalized set, duplicate one of the default sets and enter a new configuration name.

Note:

At any point, selecting “Restore Defaults” can restore default configurations.

4. To change a shortcut for a key, or add another shortcut, select the key's category from the Category dropdown list. The Actions list relevant to the selected Category is displayed.

Note:

Each action can be assigned multiple shortcuts.

5. In the Actions window, select the action you wish to modify. The current shortcut keystroke appears on the right.
 - a. To add a shortcut, type it in the Add Keystroke box and click Add Value.
 - b. To delete a shortcut, select and click Remove.
 - c. To restore the default value, click Default.
6. Click Apply and OK.

Note:

Each shortcut can only be used for one action.

Print and download Zend Studio's Official Keymap from:
<http://www.zend.com/store/products/zend-studio/>

KeyMap Properties

The following lists the properties contained in the KeyMap tab along with their description.

Editor

- **Open Next Entry** - Controls the KeyMap assignment for the Open Next Entry function.
- **Show Completion List** - Controls the KeyMap assignment for the Show Completion List function. Displays the appropriate Code Completion List for the active cursor position (code section.) Note: Code completion must be enabled in the Preferences settings.
- **Show Function Arguments** - Controls the KeyMap assignment for the Show Function Arguments function. Displays the appropriate Function Arguments for the selected function.
- **Go to Line Beginning** - Controls the KeyMap assignment for the Go to Line Beginning shortcut. Moves cursor to beginning of line.
- **Go to Line Ending** - Controls the KeyMap assignment for the Go to Line Ending shortcut. Moves cursor to end of line.
- **Show Recent Files** - Displays a window containing a list of all the Recent Files, enabling you to open a selected file.

File

- **New File** - Controls the KeyMap assignment for the New File shortcut. Opens a new blank file to the Editing window.
- **Open File** - Controls the KeyMap assignment for the Open File. Opens the selected file.
- **Close File** - Controls the KeyMap assignment for the Close File shortcut. Closes the active file.
- **Close All** - Controls the KeyMap assignment for the Close All shortcut. Closes all open files.
- **Save** - Controls the KeyMap assignment for the Save shortcut. Executes save for the active file.
- **Save As** - Controls the KeyMap assignment for the Save As shortcut. Displays the Save As dialog.
- **Save All** - Controls the KeyMap assignment for the Save All shortcut. Saves all open files.
- **Open Project** - Controls the KeyMap assignment for the Open Project

shortcut. Displays the Open Project dialog.

- **Print** - Controls the KeyMap assignment for the Print shortcut. Displays the print dialog.

Edit

- **Cut** - Controls the KeyMap assignment for the Cut shortcut. Removes the selected text from the active document and places it on the clipboard.
- **Copy** - Controls the KeyMap code assignment for the Copy shortcut. Places a copy of the selected text on the clipboard.
- **Paste** - Controls the KeyMap assignment for the Paste shortcut. Inserts the clipboard text to the active selection or cursor position in the active file.
- **Select All** - Controls the KeyMap assignment for the Select All shortcut. Selects all lines of code in the Editing window.
- **Indent Code** - Controls the KeyMap assignment for the Indent shortcut. Indents selected lines of code.
- **Undo** - Controls the KeyMap assignment for the Undo shortcut. Reverses the previously executed action (when possible).
- **Redo** - Controls the KeyMap assignment for the Redo shortcut. Re-applies the last executed action (when possible).
- **To Lowercase** - Controls the KeyMap assignment for the lowercase shortcut. Converts the selected text to all lowercase characters.
- **To Uppercase** - Controls the KeyMap assignment for the uppercase shortcut. Converts the selected text to all uppercase characters.
- **Duplicate Line/Selection** - Controls the KeyMap assignment for the Duplicate Line/Selection shortcut. Inserts a duplicate the line or selected text immediately following the selection.
- **Erase Line** - Controls the KeyMap assignment for the Erase line shortcut. Erases the active selection lines or the active line.
- **Add / Remove Line Comment** - Controls the KeyMap assignment for the Add / Remove Line Comment shortcut. Adds/Removes Comment Line marks from the active selection or line.
- **Add / Remove Block Comment** - Controls the KeyMap assignment for the Add / Remove Block Comment shortcut. Adds/Removes Block Comment marks from the active selection or line.
- **Toggle Line Wrapping**
- **Add / Remove Bookmark** - Controls the KeyMap assignment for the Add / Remove Bookmark shortcut. Adds/Removes the bookmark at the insertion point.
- **Remove All Bookmarks** - Controls the KeyMap assignment for the Remove All Bookmarks shortcut. Removes all bookmarks from the active file.
- **Show Bookmark Dialog** - Controls the KeyMap assignment for the Show Bookmark dialog shortcut. Opens the Bookmarks dialog.
- **Break** - Controls the KeyMap assignment for the Break shortcut. Inserts Break html tag.

- **Space** - Controls the KeyMap assignment for the Space shortcut. Inserts Space html tag.
- **Bold** - Controls the KeyMap assignment for the bold shortcut. Converts the selected text to all bold character formatting.
- **Italic** - Controls the KeyMap assignment for the Italic shortcut. Converts the selected text to all italic character formatting.
- **Heading 1** - Controls the KeyMap assignment for the Heading 1 shortcut. Applies beginning and ending Heading 1 HTML tags at the beginning and ending of the active selection.
- **Heading 2** - Controls the KeyMap assignment for the Heading 2 shortcut. Applies beginning and ending Heading 2 HTML tags at the beginning and ending of the active selection.
- **Heading 3** - Controls the KeyMap assignment for the Heading 3 shortcut. Applies beginning and ending Heading 3 HTML tags at the beginning and ending of the active selection.
- **Close Message Window** - Controls the KeyMap assignment for closing the system messages window
- **Expand all Folds** - Controls the KeyMap assignment for code folding for all folded elements.
- **Collapse all Folds** - Controls the KeyMap assignment for code folding for all folded elements.
- **Collapse all Non-PHP** - Controls the KeyMap assignment for code folding for non-PHP elements.
- **Collapse all Classes** - Controls the KeyMap assignment for code folding for classes.
- **Collapse all Functions** - Controls the KeyMap assignment for code folding for functions.
- **Collapse all DocBlocks** - Controls the KeyMap assignment for code folding for DocBlock comments.
- **Collapse Fold in Scope** - Controls the KeyMap assignment for code folding to collapse elements in scope.

Search

- **Find** - Controls the KeyMap assignment for the Find shortcut. Displays the Find dialog.
- **Find and Replace** - Controls the KeyMap assignment for the Find and Replace shortcut. Displays the Find and Replace dialog.
- **Find Next** - Controls the KeyMap assignment for the Find Next shortcut. Executes the find command with the Find What text, searching cyclically. If no text exists, the Find dialog displays.
- **Find Previous** - Controls the KeyMap assignment for the Find Previous shortcut. Executes the find command with the Find What text, searching backwards. If no text exists, the Find dialog displays.
- **Find in Files** - Controls the KeyMap assignment for the Find in Files shortcut.

Displays the Find in Files dialog.

GoTo

- **Go to File** - Controls the KeyMap assignment for the Go To File shortcut. Opens the Go To File dialog.
- **Go to Resource** - Controls the KeyMap assignment for the Go To Resource shortcut. Opens the Go To Resource dialog.
- **Go to Line** - Controls the keycap assignment for the Go To shortcut. Opens Go To Line dialog.
- **Go To Matching Bracket** - Controls the KeyMap assignment for the Go To Matching Bracket shortcut. Moves the cursor to the companion bracket.
- **Go To Next Bookmark** - Controls the KeyMap assignment for the Go To Next Bookmark shortcut. Moves the cursor to the next bookmark in the active file.
- **Go To Next Project Bookmark** - Controls the KeyMap assignment for the Go To Next Project Bookmark shortcut. Moves the cursor to the next bookmark in another file.
- **Back** - Controls the KeyMap assignment for the Back shortcut. Moves the back to the previous line edited.
- **Forward** - Controls the KeyMap assignment for the Forward shortcut. Moves the forward, following a Back command.

Project

- **New Project** - Controls the KeyMap assignment for the New Project shortcut. Displays the New Project dialog.
- **Open Project** - Controls the KeyMap assignment for the Save Project shortcut. Saves the active project.
- **Save Project** - Controls the KeyMap assignment for the Save Project shortcut. Saves the active project.
- **Check Included Files** - Controls the KeyMap assignment for the Check Included Files shortcut.
- **Remove from Project** - Controls the KeyMap assignment for the Remove from Project right-click menu option.

Debug

- **Add/Remove Breakpoint** - Controls the KeyMap assignment for the Add / Remove Breakpoint shortcut. Adds/Removes the breakpoint for the active line.
- **Add Watch** - Controls the KeyMap assignment for the Add watch shortcut. Displays the Add Watch dialog.
- **Step Over** - Controls the KeyMap assignment for the Step-Over (debug) shortcut. Initiates debug to fully execute the current line of script and any procedures called within it.
- **Step Out** - Controls the KeyMap assignment for the Step out shortcut. Initiates debug for the active line, if within a called script or function continues until the called script is fully executed and returns to the calling script.
- **Step Into** - Controls the KeyMap assignment for the Step Into (debug) shortcut. Initiates debug to the next line to be executed, whether in the current script or called from another script.
- **Go** - Controls the KeyMap assignment for the Go (debug) shortcut. Initiates debug to the next breakpoint or to the end of the script, whichever occurs first.
- **Go To Cursor** - Controls the KeyMap assignment for the Go To Cursor (debug) shortcut. Initiates debug to the current cursor position.
- **Run** - Controls the KeyMap assignment for the Run (debug) shortcut. Initiates debug to the end of the script.
- **Check Debug Server Connection** - Controls the KeyMap assignment for the Debug Server Connection shortcut. Opens Check Debug Server Connection dialog.
- **Debug URL** - Controls the KeyMap assignment for the Debug URL shortcut. Displays the Debug URL dialog.
- **Profile URL** - Controls the KeyMap assignment for the Profile URL shortcut. Displays the Profile URL dialog.
- **Show in Browser** - Controls the KeyMap assignment for the Show in Browser shortcut. Displays debug output in browser window.
- **Stop Debugger** - Controls the KeyMap assignment for the Stop Debugger (debug) shortcut. Stops the active debug session.

Tools

- **Analyze Code** - Controls the KeyMap assignment for the Analyze Code shortcut. Runs the Code Analyzer.

Source Control - CVS and Subversion

- **Update** - Controls the KeyMap assignment for the Update shortcut. Gets the most recent copy of the file from the CVS repository, and merges it with the local version.
- **Commit** - Controls the KeyMap assignment for the Commit shortcut. Commits your changes into the CVS server repository.
- **Add** - Controls the KeyMap assignment for the Add shortcut. Specifies files to be added to the CVS repository. The files will be added only after you commit them.
- **Status** - Controls the KeyMap assignment for the Status shortcut. Shows the current status of the files as defined in the CVS server.
- **Diff** - Controls the KeyMap assignment for the Diff shortcut. Compares between your local copy of the file and the one located on the CVS server repository, and lists the difference between them.
- **Log** - Controls the KeyMap assignment for the Log shortcut. Displays the revision History of a file.
- **Checkout** - Controls the KeyMap assignment for the Checkout shortcut. Checks out a new module on your disk.

Help

- **PHP Function Help** - Controls the KeyMap assignment for the PHP Function Help feature. Displays the PHP Function help.

Setting File Type Preferences

Zend Studio allows you to modify the default list of file types and the associated file extensions.

To customize the list of File Types and File Extensions:

1. Open the preferences window from the Main Menu, select Tools | Preferences.
2. Select the File Types tab.
3. To add an entirely new file type, click Add in the Known file types area. The Add New File Type dialog appears.
4. Enter the file name and a description and check the relevant type of file box. Click OK. The new file type is added to the current list.
5. To add file extensions relevant to the new file type, click Add in the File types extension area. The File Extension dialog appears.
6. Add the relevant file extension and click OK. The new extension is added to the list.
7. You may add as many file extensions associated with any file type. To do so, select the file type first and add file extensions.
8. To remove any custom file type or extension from the list, simply select the type or extension and click Remove.
9. When complete click apply and OK.

Note:

Default file types cannot be removed.

Setting Template Preferences

The Templates feature helps you to write code quickly and accurately. Templates are shortcuts used to insert a framework for the segment of code you are about to write. Once you have inserted a template, you can then compose the code using a combination of manual and automated (see Code Completion) code entry methods.

Template Features:

- Customize existing templates or create new templates with the Add and Edit options.
- Share templates using the import/export option.

The template list is a sortable list that can be sorted by Key, Context and Description:

- **Key** - The key combination typed to insert the template into the Editing window.
- **Context** - The templates functional context, e.g. HTML, PHP, or PHP doc
- **Description** - A textual description of the template.

Adding, Editing, and Removing Templates Zend Studio allows you to Add, Edit, or Remove templates from the list of templates maintained by Zend Studio.

To add a template:

1. Open the Preferences Window from the Main Menu, select Tools | Preferences..
2. Select the Templates tab. The Templates tab opens.
3. To add a new template, click Add. The Add a New Template dialog appears.
4. Enter an Abbreviation, Context, Description, and Template Code.
5. To add a variable from the list of variables maintained by Studio, use the Add Var function.
6. Click OK to add the new template to the list of templates in the Templates tab.
7. Permanently add the new Template to the list of templates maintained by Studio, click Apply or OK on the Templates tab.

To edit an existing template:

1. From the Templates tab, select the template you wish to modify.
2. Click Edit to open the Edit Template dialog.
3. Edit the template code (using standard text entry methods and/or the Add Variable feature).
4. Click OK. The edited version of the template appears in the Preview pane of the Templates tab.
5. To permanently add the edited version of the template to the list of templates maintained by Studio, click Apply or OK on the Templates tab. The edited version of the template is saved to the list of templates on the Templates tab.

To remove an existing template:

1. From the Templates tab, select the template you wish to modify.
2. Click Remove to remove the template from the list of templates in the Templates tab.
3. To permanently remove the template (non-default) from the list of templates maintained by Studio, click Apply or OK.

Note:

OK saves your additions and exits the Template tab; Apply saves your additions, but allows you to continue working on the Templates tab. To exit the Template tab without saving your additions, click X in the upper right-hand corner of the Templates tab dialog.

Setting Source Control Preferences

Use the Source Control tab to enable and customize the Source Control integration. Zend Studio currently integrates with two leading source control applications, CVS and Subversion. The source control tool list at the top of the Source Control tab determines which source control application will be used. The source control options in the Source Control tab change according to the chosen application (CVS or Subversion).

General CVS Settings

Path to CVS - The location of CVS's executable file. Browse to select the path where it is installed.

CVS_RSH Environment Variable - Studio allows you to use a communication tunnel between Studio's integrated CVS client and the CVS server over SSH. This is accomplished by configuring the CVS_RSH environment variable to 'SSH'. To toggle the communication tunnel off, erase 'SSH' from the CVS_RSH Environment Variable value field and click Apply.

Messages

On each of the following CVS-related dialogs listed below, you can specify parameters, such as revision number, comment, etc. If you prefer to skip these dialogs and use the default values, you may set them as Disabled in the Source Control tab.

- Show CVS Update Dialog
- Show CVS Add Dialog
- Show CVS Diff Dialog
- Show CVS Commit Dialog
- Show CVS Status Dialog
- Show CVS Log Dialog

Other Settings

Show Visual Source Control Diff

General Subversion Settings

- Path to SVN - The location of Subversion's executable file. Browse to select the path where it is installed.

Messages

On each of the following Subversion-related dialogs listed below, you can specify parameters, such as revision number, comment, etc. If you prefer to skip these dialogs and use the default values, you may set them as disabled in the Subversion tab.

- Show Subversion Update Dialog
- Show Subversion Add Dialog
- Show Subversion Diff Dialog
- Show Subversion Delete Dialog
- Show Subversion Resolve Dialog
- Show Subversion Commit Dialog
- Show Subversion Status Dialog
- Show Subversion Log Dialog
- Show Subversion Revert Dialog

Other Settings

- Show Visual Source Control Diff

Setting Dialog Preferences

Define which dialogs should appear at all times and which should be disabled. In some cases users disable dialogs directly from the dialogs "don't show this message again". These disabled messages can also be reactivated from the Dialogs tab.

The Zend Studio IDE dialogs are as follows:

- **Show Tips Dialog** - Enables/Disables the display of the Tips Dialog.
- **Show Help Agent Tips** - Enables/Disables the display of the Tips Dialog that appears when performing certain actions.
- **Show Code Analyzer Dialog** - Enables/Disables the display of the Code Analyzer Dialog.
- **Show Save As Project Dialog** - Enables/Disables the display of the Save As Project Dialog.
- **Show Included Files Dialog** - Enables/Disables the display of the Included Files Dialog.
- **Show Auto-Update Dialog** - Enables/Disables the display (at startup) of Auto-Update Dialog.
- **Show Debug Conflict Dialog** - Enables/Disables the display of the Debug Conflict Dialog.

Setting Global SQL Settings

Studio's global SQL settings are configured from the SQL tab of Preferences.

There are three ways to open the SQL tab:

1. Via the Tools menu: Tools | Preferences | SQL
2. Right-clicking on a server in the SQL tree and select Global Settings
3. Right-clicking on an element in the Query Editor and select Global Settings

The configurable settings in the SQL tab can be understood as follows:

- **Use SQL 'LIMIT' when Possible** - LIMIT defines range of results returned by a query. Default=Enabled
- **Open Every Result in a New Tab** - Determines how results are displayed. If enabled each query result is shown in a separate tab.
- **Maximum Entries in Results Table** - Maximum number of rows in a single page of results. Default=100

Note:

Limit is restricted by Paging Size and paging location. Furthermore, LIMIT is not supported in all SQL servers; SQL Lite, PostgreSQL, and MySQL support LIMIT.

- **When Double-Clicking on a Table Node** - Determines what will happen when a table node is double-clicked. Values include: Show Data, Show Metadata, Do Nothing (i.e., expand tree only). Default=Show Data
- **When Double-Clicking on Other Nodes** - Determines what will happen when another node type is double-clicked. Values include: Show Metadata, Do Nothing (i.e., expand tree only). Default=Do Nothing
- **Date Format** - Formats any date field returned as part of the results. Default=yyyy/MM/dd HH:mm:ss
- **History Limit** - Number of history elements shown in history area. Default=100

Proxy Settings

Zend Studio includes an option for setting system proxy definitions

Proxy settings should be used when there are no other options for connecting to an HTTP protocol.

There are four different ways to reach the proxy settings dialog:

1. Go directly to the Proxy Settings Dialog: Tools | Proxy Settings
2. Access the dialog from the Debug Settings Tab: Tools | Preferences | Debug Tab
3. Access the dialog from the "Auto Update Dialog" that appears every time you load Zend Studio.

Note:

Make sure your dialog preferences are set to the option "show auto update" dialog (Tools | Preferences | Dialogs).

4. Access the dialog from the Code Snippets dialog: Edit | Show Snippets.

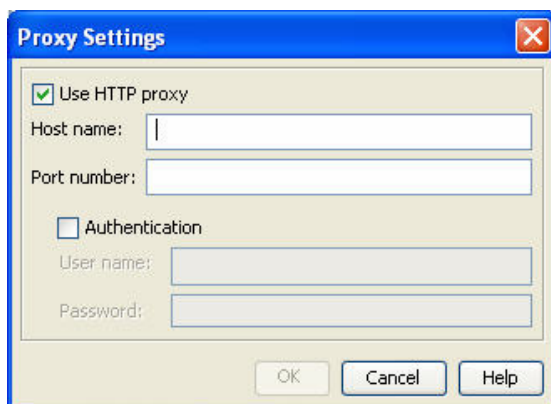


Figure: 47 - Proxy Settings Dialog

The proxy setting options are as follows:

- **Use HTTP proxy** - activates the use of Proxy settings.
 - **Host Name** - Enter the proxy server's host name
 - **Port Number** - Enter the proxy server's port number.
- **Authentication** - Optional setting for gaining secure connection to the proxy server.
 - **User Name** - The user name required by the server.
 - **Password** - The password name required by the server.

Chapter 15 - Setting Server Components

IN THIS CHAPTER...

PHP SETTINGS

CONFIGURING ZEND PLATFORM'S STUDIO SERVER COMPONENT

The configuration of PHP effects the environment used in PHP development and execution of PHP. The Zend Platform Studio Server Component provides a management and information tool to both developers and PHP Administrators. Reducing the need to manually access, search and edit the INI files that effect PHP.

Zend Platform's Studio Server Component provides the ability to:

- View Information and Settings, which effect PHP
- Access Zend Studio components on the Web Server

PHP Settings

The Server Settings (directive settings in the php.ini file) can be viewed at a glance in Studio.

The displayed settings reflect the current php.ini file values, which may or may not have changed since the last time the server was re-started.

In order to display settings in a read-only table you must log-out from the Administrator status.

Configuring Zend Platform's Studio Server

Components such as the Zend Guard, Zend Debug Server, and Zend Tools can be accessed and used by clients.

Security Settings

Zend Platform's Studio Server provides security for the Zend Studio Components, which reside on the server. Only authorized IP addresses can access the Studio Server. All other IP addresses will be denied access.

In the Studio Server, IP addresses that pass the two-stage security check are permitted to view the server settings and use Studio components. However, in order to change many of the settings, you must be logged in as an Administrator.

There are four setting categories:

- **Allowed Hosts** - These are the hosts that are allowed to initiate debugging and profiling sessions.
- **Denied Hosts** - These are the hosts that are not allowed to initiate debugging and profiling sessions, even if they are on the Allowed Hosts list.
- **Allowed Hosts for Tunneling** - These are the hosts that are allowed to use this node for tunneling. The Zend Studio Tunnel is used for debugging PHP code across a firewall to use the integration with the Zend Studio.
- **Other Settings** - These are additional settings supported by the Studio server. Currently, 'Expose Remotely' is the only setting in this category. This

setting determines whether the Debug Server will expose itself to remote clients. This is required if you want the Zend Studio Browser Toolbar to automatically detect pages that can be debugged.

The Settings screen is for Adding, Editing or Removing a host from the Allowed Hosts, Denied Hosts, or Allowed Hosts for Tunneling categories. You can also assign a value (Always, Selective, or Never) to the Expose Remotely setting for the selected node.

IP Permission Management

IP permissions are set in the IP Permissions dialog, which can be accessed using the Administrator password.

Note:

Following installation, access permission is granted to the IP address from which the Studio Server was installed, as well as any IP addresses added during the installation.

The Administrator must login to the server from a permitted IP address, and access the Security Settings.

The security settings for IP access is a two-stage process that checks the IP requesting access that consists of:

1. Allowed Host List check
2. Denied Host List check

Check for Allowed

Checks the IP address against the Allowed Host list to see if it has been allowed.

Any IP not found here will be denied access to the Studio Server and the Zend Debug Server, regardless of the Denied Hosts settings.

An IP address can be specified using wild-card characters to allow a full range of IPs to be given permission.

Checked for Denied

Checks the IP address against the Denied Host list to see if it has been denied.

This second stage allows you to filter out IP addresses from gaining access. For example, if the Allowed Host list contains 10.5.8.*, in order to keep 10.5.8.52 from access it would have to be entered in the Denied Host list. If the IP address passes both stages it can access the Zend Developers Studio including the Studio Server.

IP addresses that pass the two-stage security check are permitted to view the server settings. In order to change the server settings you must be logged-in as an Administrator.

There are two possibilities when changing the IP Access Settings for each IP Host list:

Action	In Allowed Host List	In Denied Host List	Correct/Incorrect
Grant Permission	YES	NO	Correct
Deny Permission	NO	NO	Correct
Deny Permission	YES	YES	Correct
Deny Permission	NO	YES	Incorrect, has no effect. Hosts not in Allowed list are automatically denied access.

Chapter 16 - Java Bridge

IN THIS CHAPTER...

CODE COMPLETION

ASSIGNMENTS

EXCLUSIONS

CHANGING THE JRE

ADDING JAVA OBJECTS

Zend Studio supports **Java Object** instances. *After instantiation of a PHP Java object, the PHP variable it was assigned to assumes the properties and methods of the Java class.*

In addition, **Studio 5.5 uses only one instance of the JVM** to serve all PHP scripts on a given system (as opposed to one per PHP script).

By default, Studio 5.5 uses the default JRE configuration used to load itself.

Code Completion

Code Completion offers the following functions:

- object java_last_exception_get()
- void java_last_exception_clear()
- void java_set_ignore_case(bool ignore)
- array license_info(string encoding)
- void java_throw_exceptions(int throw)

Assignments

Assignments attach the resulting Java object to the PHP variable, e.g.:

- `$a = new java("java.lang.System");`

Exclusions

The following items are excluded: Integer, Double, String, Boolean, Hashtable and Object[], NULL (Java).

They are **mapped** to **int**, **float**, **string**, **boolean**, **array** and **null** (PHP) respectively (see the table below):

Java	PHP
lang.java.String	String
java.lang.Integer / int, Long, Byte, Char, Short	Int
java.lang.Double / double, Float	Float
java.lang.Boolean / Boolean	Boolean
Object[]	Array
Hashtable	Array
all other objects remain as Java objects	Object

Changing the JRE

1. Go to **Preferences | Code Completion**.
2. Go to the section labeled **Java Bridge** (bottom left). Select the **Browse** button. The Installed JRE dialog will open.

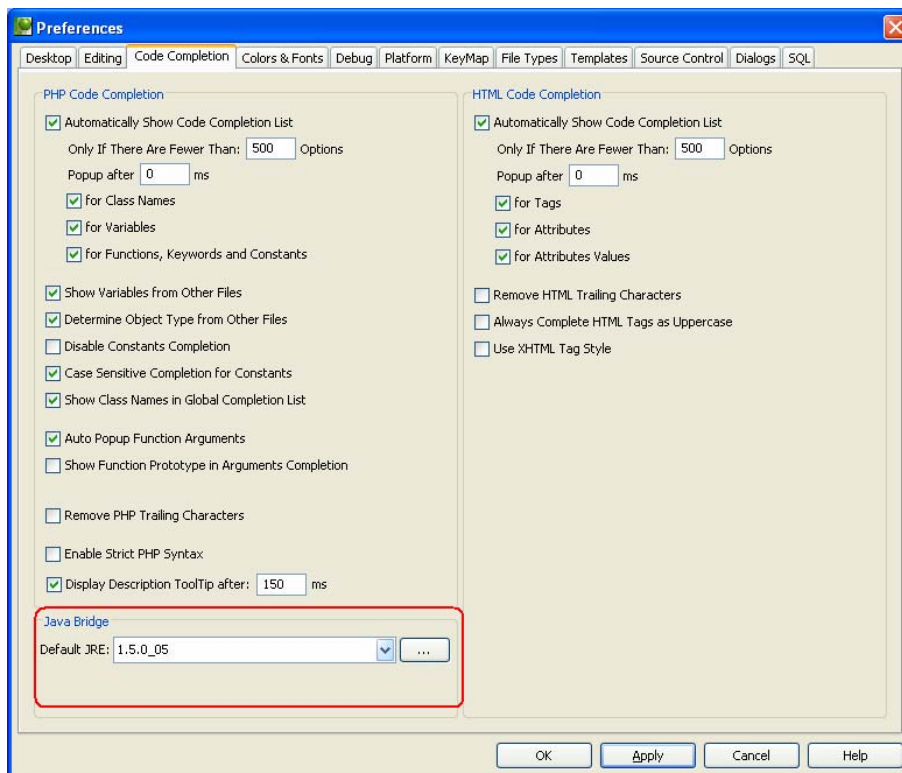


Figure: 48 - Java Bridge

3. Click **Add**.

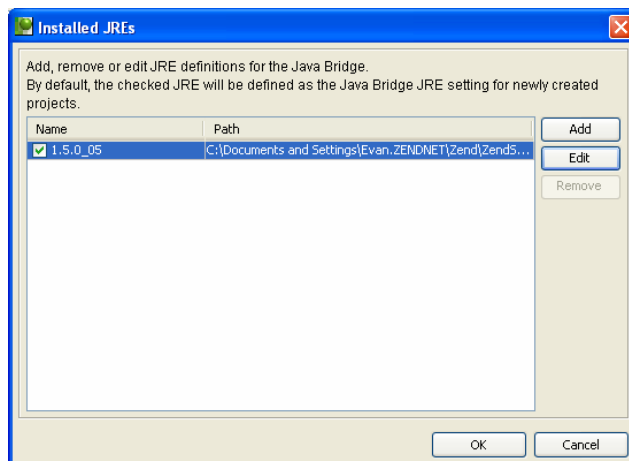


Figure: 49 - Installed JREs

4. Press **Add** or **Edit** to open the **Installed JREs** dialog. Use it to add (or edit) the JRE configuration as required.
5. The **Remove** and the **Edit** buttons are enabled for all defined JREs except for the JRE definition that is used by the Studio itself.

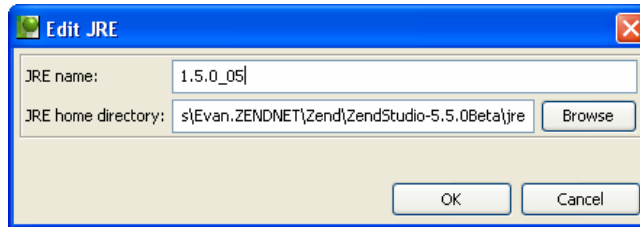


Figure: 50 - Edit JRE

New Project (Classpaths and JRE)

Classpaths and the JRE can be set for a project when creating a new project. This is done using the New Project Wizard.

1. Click **Project | New Project**. The New Project Wizard will appear.

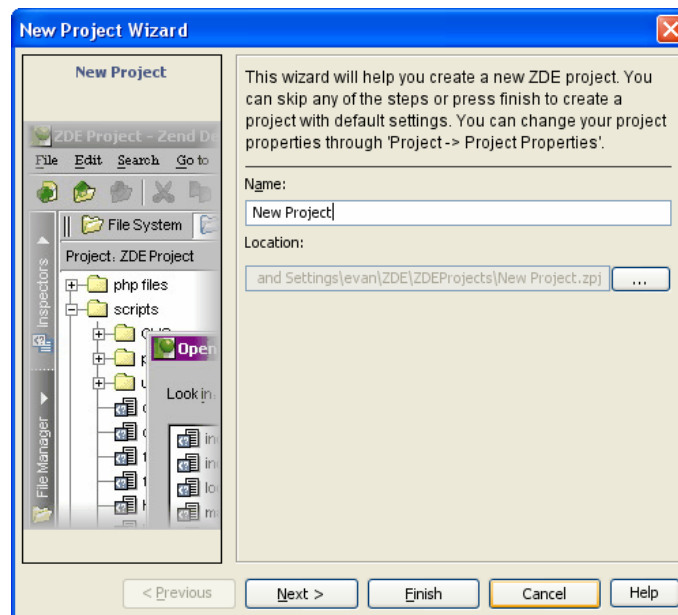


Figure: 51 - New Project Wizard

2. Click **Next**. The Path Entry dialog will appear. Add or delete path entries for your source files.

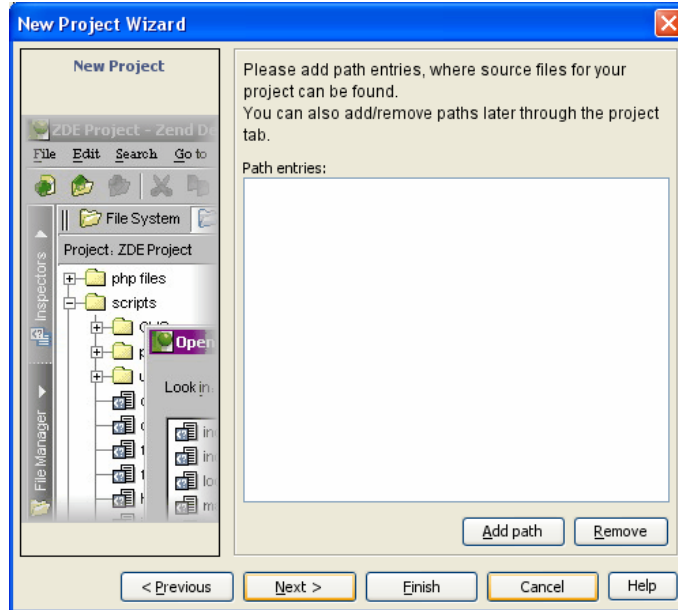


Figure: 52 - Path Entry

3. Click **Add** to add a path to the desired file(s).

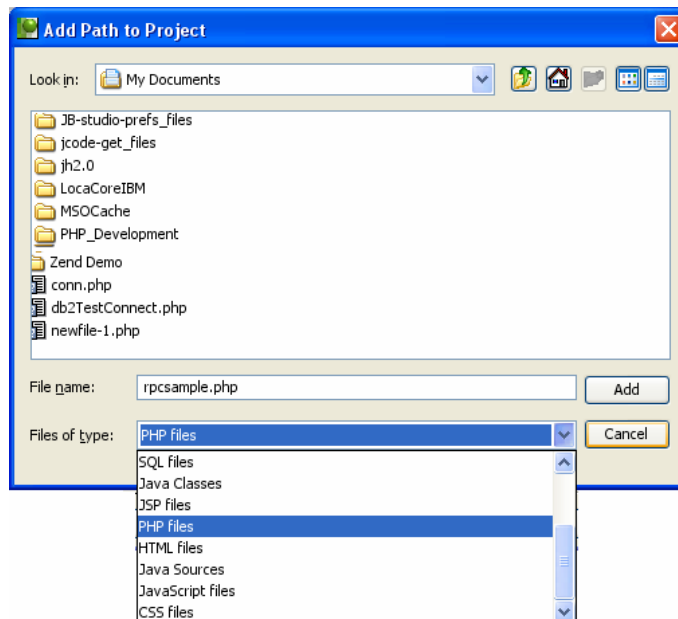


Figure: 53 - Add Path

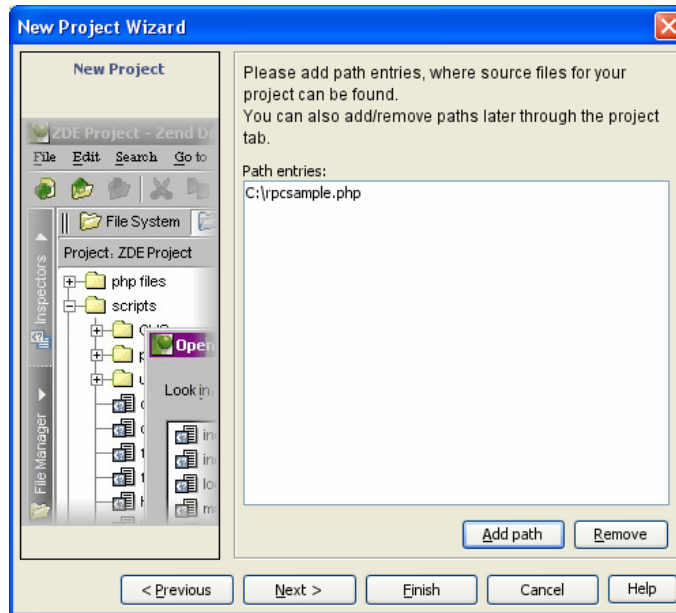


Figure: 54 - Path Added

4. Click **Next** until the dialog to set the JRE, JARS and Classfolders.

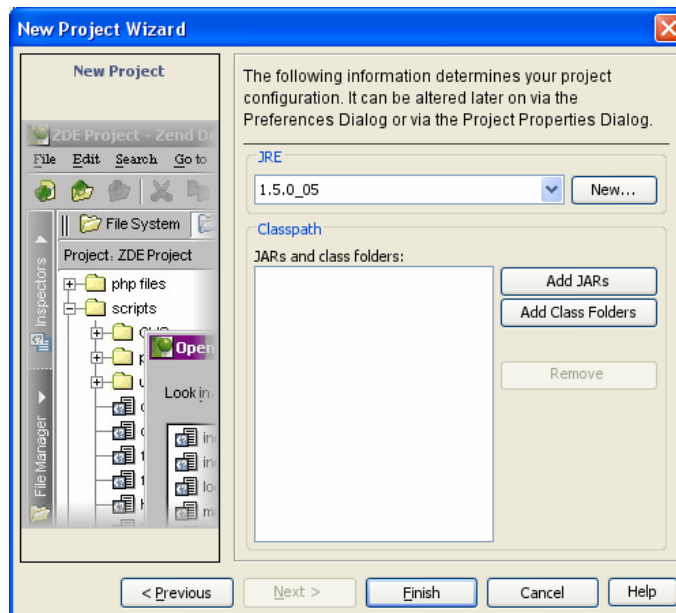


Figure: 55 - Add JRE, JARS, Class Folders

5. Click **New** to select a new JRE if required.
6. Click **Add JARs** and/or **Add Class Folders** to select new JARs and/or Add Class Folders if required.
7. Click **Finish** to create the project and finish.

Adding Java Objects

Follow the instructions below to Add Java Objects.

1. Create a project; open/create a PHP file.
2. Create a new java instance (e.g. `$a= new Java("")`).
3. Put the curser between the quotation marks and activate Code Completion (CTRL+Space). Code completion exists for the default Java packages..

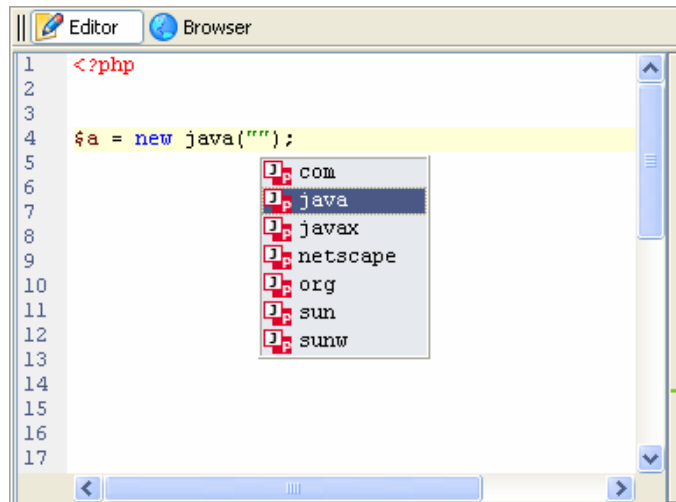


Figure: 56 - Java Code Completion Packages

4. Continue with the Code Completion procedure until the appropriate class has been selected (see sequence below).

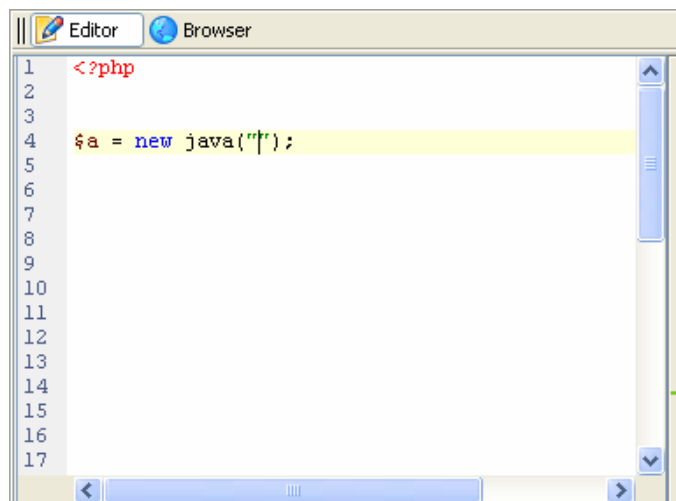


Figure: 57 - Code Completion 1/3

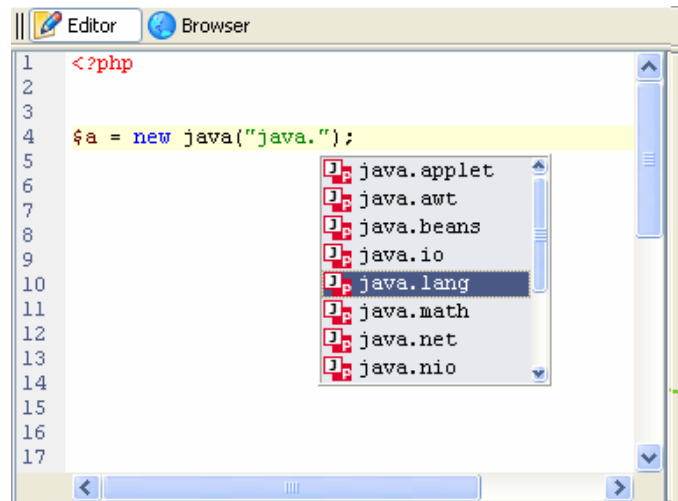


Figure: 58 - Code Completion 2/3

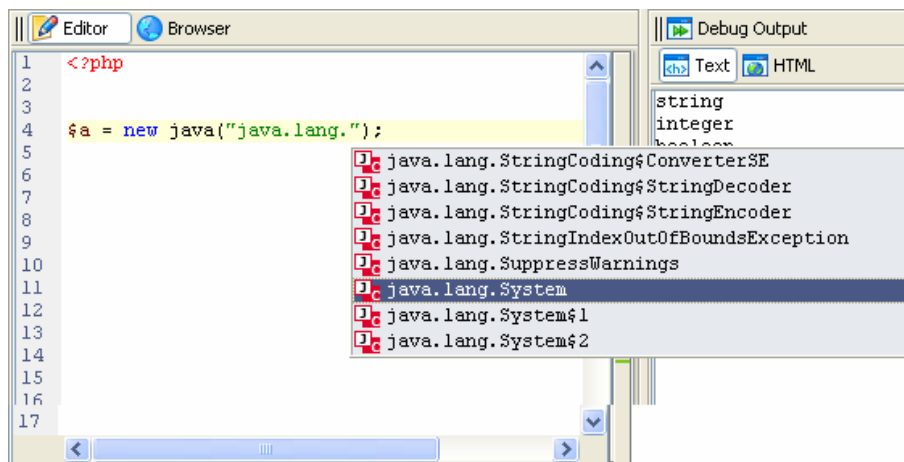


Figure: 59 - Code Completion 3/3

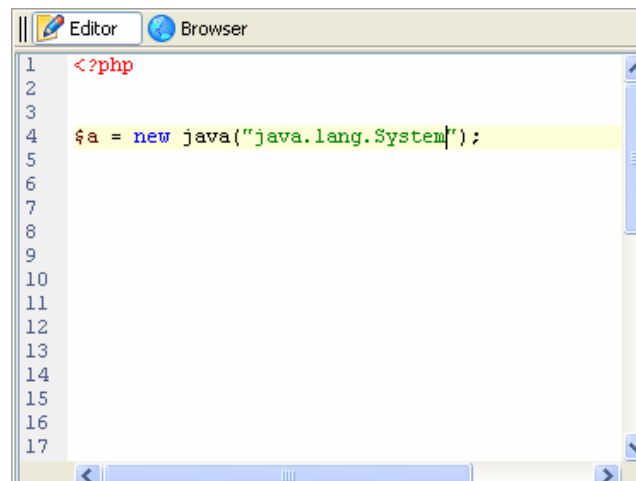
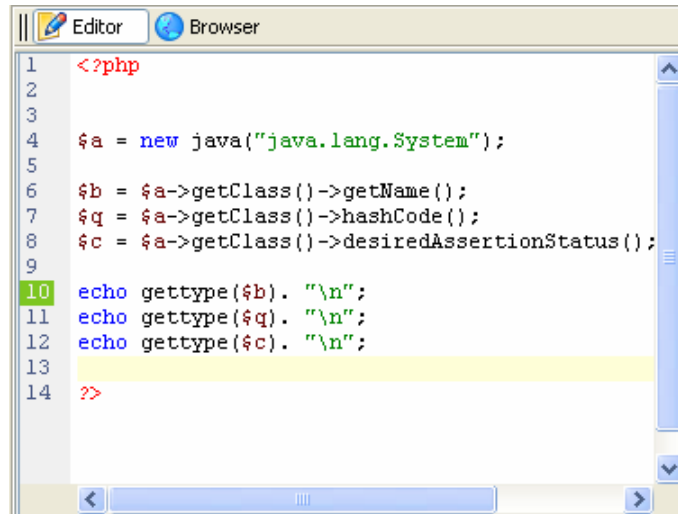


Figure: 60 - New java Object Added

5. The following sequence shows *instantiation of PHP Java objects*



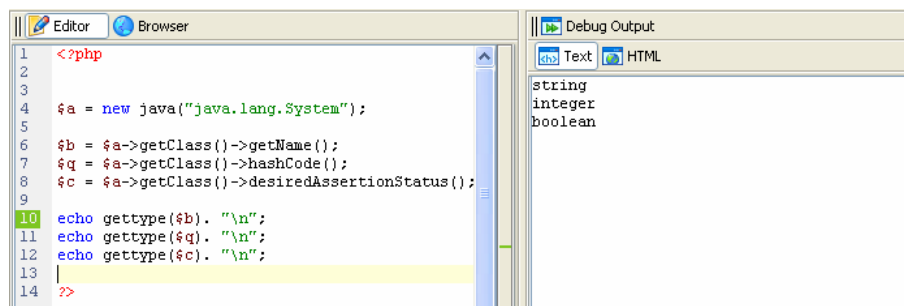
```

1  <?php
2
3
4  $a = new java("java.lang.System");
5
6  $b = $a->getClass()->getName();
7  $q = $a->getClass()->hashCode();
8  $c = $a->getClass()->desiredAssertionStatus();
9
10 echo gettype($b). "\n";
11 echo gettype($q). "\n";
12 echo gettype($c). "\n";
13
14 ?>

```

Figure: 61 - PHP Java Object Instantiation

6. The PHP variables that the Java objects are assigned to assume the properties and methods of the Java class (string, integer and Boolean).



```

1  <?php
2
3
4  $a = new java("java.lang.System");
5
6  $b = $a->getClass()->getName();
7  $q = $a->getClass()->hashCode();
8  $c = $a->getClass()->desiredAssertionStatus();
9
10 echo gettype($b). "\n";
11 echo gettype($q). "\n";
12 echo gettype($c). "\n";
13
14 ?>

```

Debug Output

string
integer
boolean

Figure: 62 - PHP Variables Assume Java Properties

Adding New Packages to the Code Completion Library

You can add Classpaths, JARs and folders containing Java files to the project. Doing this enables *the methods, classes, functions, etc., that are present in the added packages to be available to the current code completion library.*

1. Go to **Project | Project Properties**. The Project Properties dialog will open.

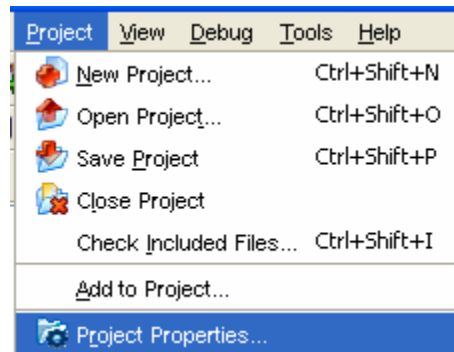


Figure: 63 - Add Packages

2. Select the **Java Bridge** tab.

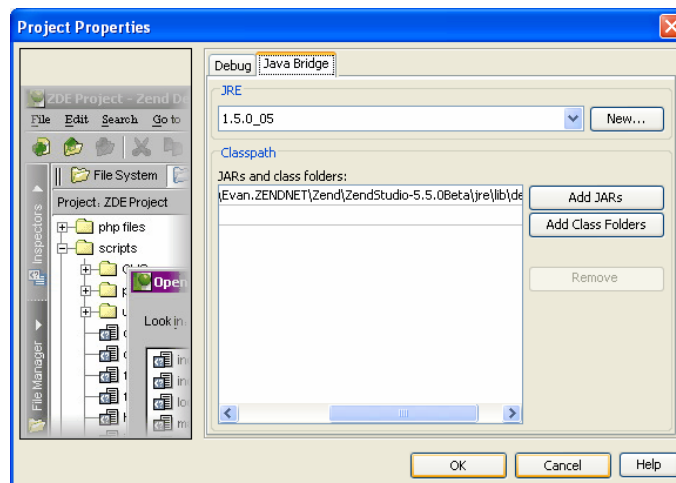


Figure: 64 - Java Bridge Tab

3. Click **Add Jars** or **Add Class Folders**. The appropriate Add dialog will appear.

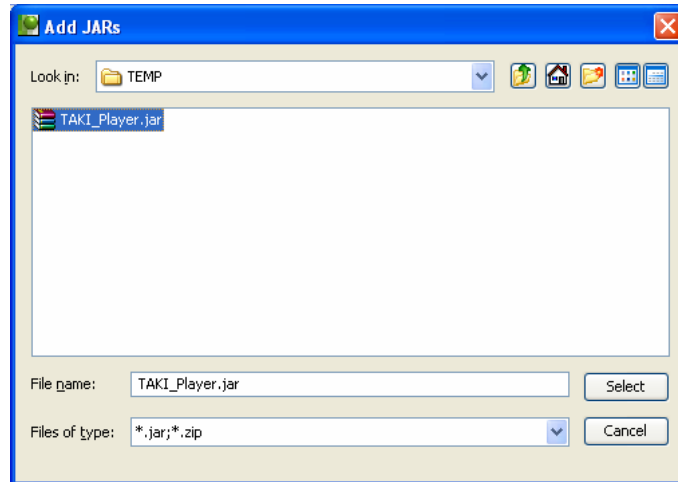


Figure: 65 - Add JARs

4. Locate the JAR / Class folder. Click **Select** and **OK** to add the JAR(s) and return to the IDE. **The added objects now appear in the code completion library.**

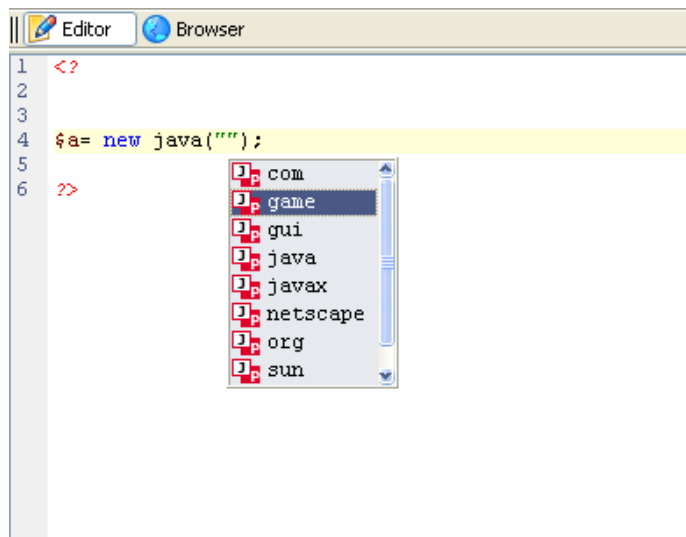


Figure: 66 - Added Java Objects

PHP /Java Integration Code Example

```

<?

// EJB configuration for JBoss. Other servers may need other settings.
// Note that CLASSPATH should contain these classes
$envt = array(
    "java.naming.factory.initial" =>
    "org.jnp.interfaces.NamingContextFactory",
    "java.naming.factory.url.pkgs" =>
    "org.jboss.naming:org.jnp.interfaces",
    "java.naming.provider.url" => " jnp://yourflowers.com:1099"
);
$ctx = new Java("javax.naming.InitialContext", $envt);

// Try to find the object
$obj = $ctx->lookup("YourflowersBean");

// here we find an object - no error handling in this example
$rmi = new Java("javax.rmi.PortableRemoteObject");
$home = $rmi->narrow($obj, new Java("com.yourflowers.StoreHome"));

// $hw is our bean object
$store = $home->create();

// add an order to the bean
$store->place_order($_GET['client_id'], $_GET['item_id']);
print "Order placed. Current shopping cart: ";

// get shopping cart data from the bean
$cart = $store->get_cart($_GET['client_id']);

foreach($cart as $item) {
    print "$item[name]: $item[count] at $item[price]\n";
}

// release the object
$store->remove();

?>

```

Chapter 17 - Zend Guard™

IN THIS CHAPTER...

OBFUSCATION

ZEND LICENSE MANAGER

ENCODING APPLICATIONS

Zend Guard is the first electronic licensing solution for the PHP marketplace combined with the encoding solution that pioneered PHP intellectual property protection.

Zend Guard enables software vendors to maximize profitability with intellectual property protection and software license management.

Zend Guard is the first Electronic Licensing solution for the PHP marketplace, combined with the Encoding solution that pioneered PHP intellectual property protection.

Your unprotected intellectual property, in the form of plain text PHP scripts and software with no license restrictions, can be copied, modified, and maintained by someone else. It is open to your competitors, to hackers, and to well-meaning developers at deployed customer sites.

The Zend Guard is comprised of two components providing a dual layer of protection for PHP applications.

Obfuscation

Zend Guard is the industry standard in PHP intellectual property protection. It protects exclusive and commercial PHP applications by encoding PHP scripts. Developers enjoy the benefits of a leading open-source language, while protecting PHP code when ready to distribute applications. By protecting applications, Independent Software Vendors (ISVs) expand distribution and revenue on maintenance and support.

Code Obfuscation let's Independent Software Vendors (ISVs), developers and businesses improve intellectual property protection from reverse engineering by obfuscating PHP source code.

Three levels of protection are available:

- Encoding only – Full encoding but no obfuscation for full backward compatibility.
- Basic - Obfuscation of local variables names.
- Strong - Obfuscates all function names, function calls, class names, class methods and local variables. (Not including code components added to the Exclude List).

Zend Guard supports the latest industry standard with full support for PHP 5.1.

Zend License Manager

The Zend License Manager enables ISVs to manage the commercial distribution of their PHP applications by generating license keys and creating files that require a license key to operate. Software vendors can easily specify license models without changing their application's source code. Licensing PHP applications enables software vendors to dramatically penetrate their available market, enhance customer loyalty and account penetration, resulting in increased revenue and profitability.

The Zend License Manager extends to developers and ISVs the option to not only protect that code from copyright infringement, but also impose license restrictions on those files by marking such encoded files as ones that require a license to operate. The Zend License Manager generates unique licenses based on defined criteria, manages those licenses and then ensures those files require the licenses to operate. Applications will not run unless the proper software license is found, allowing vendors to invoke license policies that will result in maximized sales.

Encoding Applications

Zend Guard allows code developers to encode their PHP script at any time during the development process, but more importantly prior to distributing or publishing it.

Zend Guard allows you to encode your PHP scripts in several different ways.

The following lists examples of file encoding options:

- **Expedited at Run-time** - Eliminating compilation and optimization at run-time.
- **Unreadable Source Code** - Files are encoded and not in a human readable format.
- **Require Valid License** - Files can be encoded to support or require licensing.
- **Time Limited** - Files can be encoded to expire at a set date.
- **Encoded-Only Mode** - Files can be set only to cooperate with associated encoded files that bear the same encoded signature.

The encoding functions are accessible through an integrated GUI, shown below:

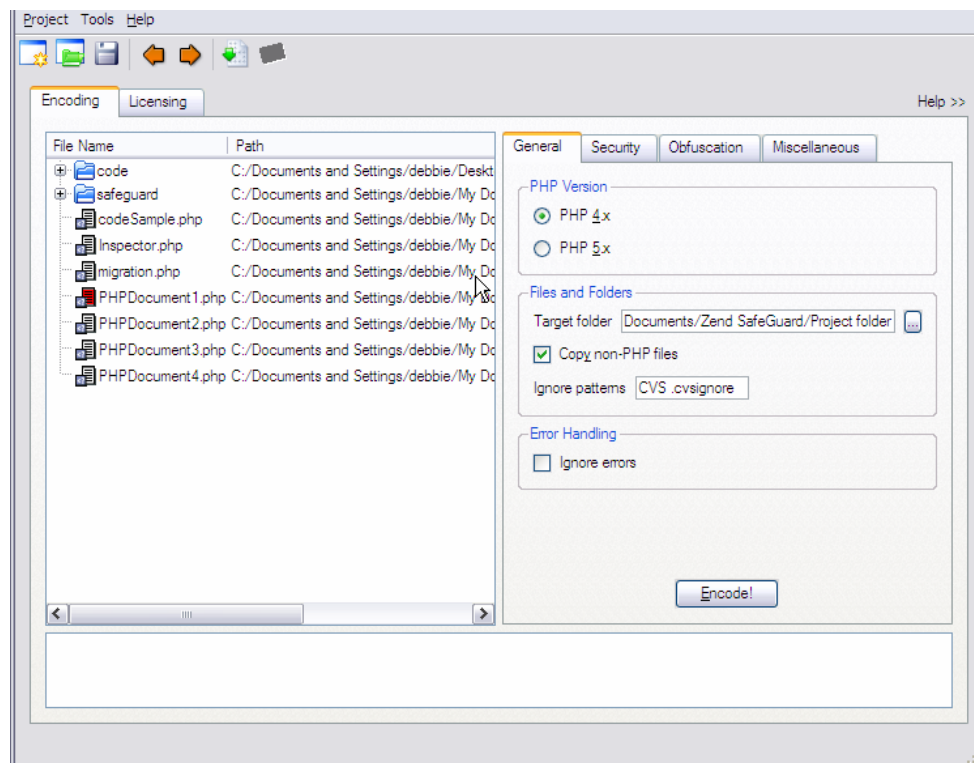


Figure: 67 - Zend Guard

Table of Figures

Figure: 1 - Zend Studio Workflow	12
Figure: 2 - Zend Studio User Interface	20
Figure: 3 - Main Menu	22
Figure: 4 - Main Toolbar	24
Figure: 5 - Browser Toolbar	26
Figure: 6 - File Type Drop-Down Menu	39
Figure: 7 - Desktop Preferences	45
Figure: 8 - Open <File>	47
Figure: 9 - Open <URL>	47
Figure: 10 - Open <URL>	48
Figure: 11 - Templates in the Code Completion Menu	49
Figure: 12 - Anti-Aliasing	55
Figure: 13 - Restart Studio	55
Figure: 14 - Goto PHP Resource	64
Figure: 15 - Goto Project File	65
Figure: 16 - WSDL File Generator Wizard	66
Figure: 17 - SOAP Client	67
Figure: 18 - Additional Actions Added to Menu	67
Figure: 19 - Inspector View	69
Figure: 20 - The WSDL Wizard, Global Settings Dialog	72
Figure: 21 - Auto Insert phpDoc Comment Tags	75
Figure: 22 - File Inspector	81
Figure: 23 - Project Inspector	82
Figure: 24 - PHP Function Inspector	83
Figure: 25 - Add New Watch Dialog	91
Figure: 26 - Watches List	92
Figure: 27 - Tunneling Settings Dialog	95
Figure: 28 - Code Analyzer	98
Figure: 29 - Platform Integration	99
Figure: 30 - Enabling Platform - Preferences	100
Figure: 31 - Event Settings	100
Figure: 32 - Events Listed	101
Figure: 33 - Expanded Event View	101
Figure: 34 - Profiler Information Tab	103
Figure: 35 - Function Statistics Tab	104
Figure: 36 - Call Trace Tab	105
Figure: 37 - Source Control Diff Viewer	108
Figure: 38 - File Status	113
Figure: 39 - Add SQL Server	116
Figure: 40 - SQL Server Tree	119
Figure: 41 - SQL Data Display	120
Figure: 42 - SQL Tree	124
Figure: 43 - SQL Tree	126
Figure: 44 - Query Results Table	132
Figure: 45 - SQL Query Control	133
Figure: 46 - SQL Error Messages	135
Figure: 47 - Proxy Settings Dialog	160
Figure: 48 - Java Bridge	166
Figure: 49 - Installed JREs	166
Figure: 50 - Edit JRE	167
Figure: 51 - New Project Wizard	167

Figure: 52 - Path Entry	168
Figure: 53 - Add Path	168
Figure: 54 - Path Added	169
Figure: 55 - Add JRE, JARs, Class Folders	169
Figure: 56 - Java Code Completion Packages	170
Figure: 57 - Code Completion 1/3	170
Figure: 58 - Code Completion 2/3	171
Figure: 59 - Code Completion 3/3	171
Figure: 60 - New java Object Added	171
Figure: 61 - PHP Java Object Instantiation	172
Figure: 62 - PHP Variables Assume Java Properties	172
Figure: 63 - Add Packages	173
Figure: 64 - Java Bridge Tab	173
Figure: 65 - Add JARs	174
Figure: 66 - Added Java Objects	174
Figure: 67 - Zend Guard	177