**Foreword**

As an extensible tool integration platform, Eclipse has spawned the creation of a large and vibrant open source and commercial software ecosystem. Zend Studio for Eclipse is a significant addition to that ecosystem.

From the perspective of our community, the PDT (PHP Development Tools) project is an important new addition to Eclipse. Although we are best known for our Java IDE, the Eclipse community has long since evolved into something much more interesting. I believe that it is fair to say that today Eclipse has become the leading open source tooling integration platform. Our goals are not to simply create an integrated development environment (IDE), but to create a platform for many IDEs. Even more challenging, our community is creating a tooling platform that supports the creation of all types of software tools, extending well beyond the scope of traditional developer tools. Today at Eclipse, you can find tools that span the complete software development lifecycle from modeling and design to data analysis to development to testing and monitoring.

But amongst many developers, the perception remains that Eclipse is a “Java thing”. While it is true that most of the code built at Eclipse is written in Java, our goals have always been to provide development tools that span as many programming languages and platforms as possible. For example, our C/C++ Development Tools (CDT) project has achieved a lot of success in the Linux and embedded development worlds. The PDT project upon which Zend Studio for Eclipse is based is a further (and critical) evolution of this vision.

As one of the largest and fastest growing web development languages, PHP has quickly grown to a mainstream enterprise development language and platform. In addition to being the web development language of choice for many sites (we use it ourselves at www.eclipse.org!), it is also the technology that underlies many of the Web’s most important wiki and content management systems. As an everyday example, consider the very popular Wikipedia site which is built on MediaWiki which is, in turn, based on PHP. Another very popular PHP-based web property is Facebook. There are too many examples to possibly list here, but the point is that PHP as a language and as a platform is firmly woven into the fabric of our everyday use of the Web. And it has achieved that in just slightly over 10 years of existence (starting from PHP 3). At the time of writing, PHP is currently the fourth most popular programming language on the planet (source: http://www.tiobe.com/tpci.htm). In fact, the now-classic combination of PHP with Linux, Apache and MySQL (commonly referred to as the LAMP stack) is widely credited with the rapid growth of dynamic web properties over the past decade. It is a proven stack which meets the business and technical needs of many of the most important websites that we all use on a day-to-day basis.

The Eclipse community itself may be used as a proof point of PHP’s adoption and success. While the Eclipse platform and projects are implemented in Java, if you take a look at the many web properties run by the Eclipse Foundation, they are all implemented in PHP.
PDT provides you, the PHP developer, with the tools you need to build, debug and deploy PHP applications.

But just as importantly, because PDT is based on the Eclipse platform, it provides you with not only great PHP development tools but with a tools platform that you can use to integrate with other Eclipse-based products and open-source plug-ins. PDT is designed to be an open extensible system, and a large measure of its future success will be growth in the number of its own ecosystem of Eclipse plug-ins, both open-source and commercial. But in addition, there are a great many existing Eclipse plug-ins that you can draw upon to extend your tooling environment. To tap into this world of Eclipse extensions, take a look at our Eclipse Plug-In Central website (also implemented on PHP) which can be found at:

http://www.eclipseplugincentral.com/

Hopefully both the Zend Studio for Eclipse Developers Guide and the PDT toolset will make you a more productive PHP developer. But please remember that Eclipse is all about active community involvement, and we hope to welcome you soon as an active contributor to PDT and other projects at Eclipse. As you work with PDT and the capabilities described in this book, I'd encourage you to communicate your successes back to the community, and perhaps consider contributing any interesting extensions you may develop. The PDT website may be found here:

http://www.eclipse.org/pdt/

It includes pointers to the PDT newsgroup—where you can communicate and share your results with other PDT users and adopters—and pointers to the Eclipse installation of Bugzilla, where you can contribute your bugs, comments and patches.

Mike Milinkovich
Executive Director,
Eclipse Foundation
Introduction

PHP is currently the most widely used programming language on the Web with over 5 million developers, responsible for 40% of existing web applications. The simplicity of PHP has led to more than 20 million domains written in PHP, with growth continuing. When compared with other languages for achieving the development of a web application, PHP has proven to have tremendous advantage with its simplicity, in terms of the amount of work required and the potential complexity of its code.

The need for an editor or a development environment to create web applications with a short “time to market” is obvious, and different possibilities are available today for the PHP developer community. The possibilities can be categorized into three main groups in which each group introduces a different set of features, addresses different needs, and subsequently is tagged with a different pricing.

The first group, generally known as Simple Editors, includes the most basic feature set, such as syntax highlighting as part of the editor. Some of these editors come with the different operating systems, and some are the evolution of those (for example, NotePad and NotePad++). This group of editors usually doesn’t include management tools like debugging or code analyzing tools, and is good for quick pinpoint development rather than large and complex web applications. Most of these editors are free of charge.

Basic Integrated Development Environments (IDEs) are the second group; they include an additional layer of features. These features can include basic debugging, project management, and several analysis tools. Some of these editors are free of charge, and sometimes they are even open source products.

The last group, known as Professional IDEs, includes all-in-one solution products. These development environments generally include development, management, analyzing, debugging, and deployment tools. The complete feature set in these products provides the capability to support full product development life cycles, starting from the development of the code until the deployment to the production server. A Professional IDE is a commercial product and can include an installation wizard and product support as well.

Over the years we can see a marked increase in the number of developers moving to professional IDEs from the basic editors. The need for team support, deployment tools, and quick development has convinced many companies to invest their money in the purchase of development tools with a quick return of both investment and productivity.

The gap between simple and professional IDEs can also enable some companies to provide a product free as a simple, initial solution. There may also be the option to pay for upgrades and thus be entitled to then use a professional IDE, but this is not always the case.
Zend Studio for Eclipse is based on the Eclipse technology in general and the PHP Development Tools (PDT) project in particular. The decision to develop based on the Eclipse technology was made because there are a few million developers who use Eclipse or Eclipse-based products. Many of those developers are looking at PHP as a way of developing rich Internet applications, and they simply wanted PHP support in Eclipse.

Zend has been working on Zend Studio for Eclipse for quite some time parallel to the development of the PDT Eclipse project. The product has been released a few times to a close group of beta testers to ensure the product stability and user interface usability and to gather feedback and bugs.

This book’s authors, Peter MacIntyre and Ian Morse, who have vast experience in the PHP world and have been developing with Zend Studio for Eclipse in the past year, provide a great understanding of Zend Studio for Eclipse and its functionality.

The book provides explanations and instructions on how to use the best professional PHP IDE available today! In this book you also learn to develop web applications in the easiest and most productive way because this book not only introduces you to the many wonders of Zend Studio for Eclipse, but also guides you in developing a small web Customer Relationship Management (CRM) application.

Yossi Leon
Product Manager, Development Tools
Zend Technologies, Inc., the PHP Company
Environmental Settings

This chapter discusses in detail all the options available to you in setting up the overall working environment of Zend Studio for Eclipse. The many options available cover subjects such as text color, tab styles, and server definitions.

First, however, there is a little more detailed coverage of the concepts of perspectives and how they can be employed to great gain in the context of the working environment of Zend Studio for Eclipse.

**The Concept of Perspectives**

As was discussed briefly in Chapter 1, “A First Look at Zend Studio for Eclipse,” the concept of perspectives is helpful when you are using Zend Studio for Eclipse in different contexts and different stages of development. You should learn how easily you can define and utilize perspectives, and over time this ability should become second nature to you.

To begin with, you should look at the views that you want to have in focus. Then you can save those views and name them as your own perspective. To look at the views available, you can select Window, Show View, Other. A dialog appears, showing a list of all available views within Zend Studio for Eclipse. After you select the views that you want and arrange them on your screen, you can save that collection of views under a perspective name of your own choosing. To do this, select Window, Save Perspective As. Figure 3.1 shows the perspective naming dialog with the existing perspective names already listed.

The customization of the perspective does not end there. You can manage some of the toolbar icons and menu items that are connected to the perspective. To do this, select Window, Customize Perspective, or right-click on the perspective in question on its toolbar on the top-right side of the IDE and select Customize. Be sure that you name your new perspective uniquely before you change too many of its details just in case you are changing the options of a “stock” perspective that you don’t want to change permanently. When the customization window opens, as shown in Figure 3.2, you can change any of the settings made available to you.
Figure 3.1 Saving a newly designed perspective.

Figure 3.2 Customizing a newly created perspective.

On this customization window, you can control what menu items show up on each of three submenus: namely, the File, New menu; the Window, Open Perspective menu; and the Window, Show View menu. This is controlled through the Submenu drop-down list on the Shortcuts tab of the Customize Perspective window. When this drop-down is changed, the subcategories that are available adjust to the new context.

By switching to the Commands tab within this same customization window, you can then control both the menu item and toolbar item of other areas within Zend Studio.
for Eclipse (but only for the current perspective that you are customizing). Figure 3.3 shows the options for this tab. For example, if you turn on the HTML Composer Actions selection (as shown in the Figure 3.3), you activate the whole list of menu items under the Modify menu (which appears as a new top-level menu between the Project and Run menus), and you add a series of toolbar items that have their own toolbar at the top of the IDE (see Figure 3.4). Again, this happens only within the current perspective.

![Figure 3.3 Customizing a newly created perspective.](image1)

![Figure 3.4 Newly added HTML Composer Actions menu and toolbar items.](image2)

As you should be aware by now, the perspective feature of Zend Studio for Eclipse alone is extremely valuable to PHP developers. But, as you would expect from a modern computer programming interface, there is much more control at your fingertips. Enter Zend Studio for Eclipse Preferences.
Zend Studio for Eclipse Preferences

As with any major software product, you would expect to be able to have some sort of master control over its performance and look and behavior. Zend Studio for Eclipse is no exception; however, there are some differences between the Zend Studio for Eclipse product layer itself and the Eclipse foundation on which it is built. Under the Window menu, there is an item that opens up this Pandora’s box of options: Preferences, located at the bottom of the menu displayed when you select Window. If you’re not careful, you can get yourself into some trouble using this option. However, it’s not too difficult to back yourself out of any corners that you may find yourself in.

Before we get too close to the details of the preferences, here are just a few words about the nature of the preferences. Some preferences have systemwide ramifications and some are quite specific to a certain topic or process. We guide you through most of these options in this chapter, but we merely mention some of these differences as such. Throughout the remainder of this chapter, we first discuss systemwide preferences and provide some detailed examples of them and then describe some specific preferences that affect how PHP is handled within Zend Studio for Eclipse itself. We deal with all other preference settings that may be adjusted depending on a certain topic inside that specific context.

Systemwide Preferences

In the following sections you will learn about much of the systemwide preferences that can be controlled in this IDE. Some are of a general nature that can affect the overall environment and some are more specific to a certain aspect of the IDE.

General

General preferences are exactly that—preferences to the overall environment of the Zend Studio for Eclipse IDE. With the highlight on the first item, named General, you can see the initial control area, as shown in Figure 3.5. Here, you are offered three checkbox controls and an Open Mode for item selections within the IDE. The three checkboxes are as follows:

- **Always Run in Background** — When this option is checked, the developer can continue working while longer running processes are executed in the background.

- **Keep Next/Previous Editor, View and Perspectives Dialog Open** — When this option is turned on, the Selection dialog for Editor, View, and Perspective Cycling stays open. When it is not selected, the dialog disappears. See the following note for a brief description on cycling.

- **Show Heap Status** — This option turns on an indicator that shows how much Java Heap memory is being used by Zend Studio for Eclipse. This indicator generally shows up in the bottom-right side of the display on the status bar. It also shows an icon that allows you to perform garbage collection.
Note

Zend Studio for Eclipse gives you a relatively quick way to cycle through your open files in the editor with the Ctrl+F6 key combination. When you use this key sequence, a small dialog opens, showing all the currently open files (it stays open only when the Keep Next/Previous Part Dialog Open option is turned on in the General preferences). From this dialog, showing currently open views in Figure 3.6, you can then select one of these files to switch to. Selecting one of these files has the same result as clicking on the tab in the editor that was connected to an open file. The same functionality is available for Views (Ctrl+F7) and Perspectives (Ctrl+F8).
Within the General Preferences window, the section labeled Open Mode in the lower half of the screen includes the options Double Click and Single Click. Choosing one of these options simply means that anywhere within the Zend Studio for Eclipse environment if you are selecting an item, you can either double-click or single-click on it. Also, within the Single Click option, you can control the hover action or the use of the arrow keys within that context.

If you click on the tree listing on the left side of this General Preferences window and select the Appearance item, you are given a host of options that, oddly enough, have an effect on the overall appearance of the Zend Studio for Eclipse IDE. The first option here simply allows you to select what I call a master perspective, or as Zend describes it, a presentation. If you change this to another option, Eclipse 2.1, and apply the changes, you see a totally different layout or presentation to the overall IDE. The perspective toolbar switches to a vertical orientation on the left side of the IDE, and other views are opened for you. If you don’t like this presentation, change it back to the default in the preferences window. As another example, you can set the tabs that are on the top of the code editor to move to the bottom of the view if you prefer by selecting Bottom instead of Top in the Editor tab position. You can control the tab positions on the views as well as the editor if you want, and you can show or hide the descriptive text on the perspectives toolbar from this Appearance section.

Under the Appearance tree item are two subitems: Colors and Fonts and Label Decorations. Here, you can control the colors and font sizes of many of the editors, views, and wizards within Zend Studio for Eclipse. The Label Decorations simply add more information to a particular item. For example, if you turn on the PHP Problem Decorator and apply the changes, a small red X appears on any of the tree items in the PHP Explorer that may have problems. This is an aid in locating code issues within each tree item.

**Note**

Some of the General Preferences window tree items also have multiple tabs within them. For example, the General Appearance Compare/Patch Tree item has two tabs named General and Text Compare.

The next tree item under General Preferences is the Compare/Patch option. There are two tabs here to be concerned with. On the first tab, General, there are a number of checkboxes that you can adjust. Without going into each specific checkbox attribute, we can summarize all these options to have an effect on the comparison of versions of your code, whether with local histories of saved files or with CVS versions of your code. The second tab gives a little more visual clue as to what you will be adjusting. The Text Compare tab, shown in Figure 3.7, shows how a code comparison will look as you make some of the adjustments.
On this tab, you can see that there are options in how the comparison between two files is displayed. If you want to see this in action, close your Preferences window and select a file in the code editor that you have been editing; right-click and then select Compare With, Local History from pop-up menu. If there is a local history, Zend Studio for Eclipse displays that with the comparison options that you have set.

The next option under the General Preferences is called Content Types; here, you simply assign different file types to various code styles. This is done via the file’s extension. Most of these file associations are preset for you and don’t really need to be adjusted, but the option is here for you just the same. For example, if you expand the text option and select the PHP tree item (PHP Content Type), you see a list of file associations that are preset (locked in) for Zend Studio for Eclipse. You have the option of adding new associations if you so choose.

**Editors**

The next major portion of the General Preferences window has to do with the editors that you will be using within Zend Studio for Eclipse. There is another section specific to PHP that directs the editor on a more specific level, but here, as the name implies, you can adjust the editors’ features on a general level. Clicking on the Editors tree item itself reveals only a few options. You can turn on or off the tabs that appear on the top of each open file in the code editor by selecting the Show Multiple Editor Tabs option. Just above that, you can control the number of files that show up on the list of recently used files. The default is four, but I usually increase this number to at least eight when I’m working on a project of any significance.
The next items that you can control here are all related in some degree, and this is a feature that I am growing to like. You can tell Zend Studio for Eclipse to close open files automatically if more than a set number of files is currently open. If this option is turned on, the “first in, first out” rule is put into effect. If another file is opened in the editor, the oldest open file is closed (and you can control what happens to that file as it is being closed) and the newly requested file is opened. This method of file management can be well employed to maintain a “clean” development environment.

The next two tree items that are sublevels to the Editor level are File Associations and Structured Text Editors. In the File Associations area, you can control what happens when certain file types are encountered by Zend Studio for Eclipse. You can open specific editors, select *.sql from the File Types list, for example, and can have Zend Studio for Eclipse open the SQL Source Editor or the SQL Editor automatically. If you have two or more options per file type, it would be wise to select a default editor so that Zend Studio for Eclipse can open the files in the appropriate editor with little fuss.

After you have established some of your preferred file associations, you can look at the next option under the Editors branch called Structured Text Editors. This area of options relates to any editors that Zend Studio for Eclipse uses for structured text. This means any kind of code or HTML style of content that is not merely descriptive text but is more computer language based. In general terms, you can have matching brackets highlighted for you in these editors, you can have errors reported to you, you can be warned when unsupported content appears in those editors, and you can have code folding enabled where applicable. You can also control the color of the highlighted brackets if you want. The next tab within this same tree option is called Hovers; here, you can control how the hover pop-up help is displayed. For example, if you want to see problem help only when you hover over identified problem HTML code while holding down the Control (Ctrl) key, you can select the Problem Description item and add the pressed key modifier to it.

Still under the General Editors tree branch is a section called Text Editors. Without going into all the details here, you can control many features and options in relation to text editors within Zend Studio for Eclipse. For example, you can turn on or off the display of the line numbers, you can highlight the current line that the editor is focused on, and you can alter the width of the inserted spacing of the Tab key. Under the Annotations area, you can alter the colors and appearance of editor icons like the breakpoint, the error signifier (red underline squiggles), and code warnings, to name a few. You can even have your editors do spell checking, but this may not be advisable because a great deal of structured content like HTML and PHP uses short-formed and cryptic words for commands and functions. There would naturally be a lot of spelling errors in this context, so turn on this option with care and understanding of what it will do.

The next item under the General section of the Preferences is the keyboard mapping level. Under Keys, you can remap any keyboard key combination that is preset and make it your own. The neat thing here is that the keys are mapped in context. By that, I mean that you can map a key combination for one context of Zend Studio for Eclipse—Ctrl+B to build all code in a project, for example—and have the same key combination...
defined for another context—to turn on bold in the WYSIWYG editor. You can view these key mappings in the data grid and then alter them in the lower portion of this screen where the bindings are shown, see Figure 3.8.

![Key mapping Preferences window](image)

**Figure 3.8** The Key mapping Preferences window.

The next branch on the General tree is the Perspectives branch. Here, you can control certain aspects of the preferences and how they react in the Zend Studio for Eclipse IDE. You can tell the perspective to open in a new window, open as a new view within an existing perspective, and you can tell Zend Studio for Eclipse to always open a certain perspective when a new project is started.

Below the Perspectives branch is the one called Search. Here, you can control how the Search feature works. You can tell it to reuse existing search windows on subsequent searches so that the desktop is less cluttered, you can colorize your results, and you can give the search view a default perspective in which to reside.

Following the Search branch is the Startup and Shutdown section. There is very little to control here except which Eclipse plug-ins should be active when Zend Studio for Eclipse starts and some workspace controls. One item that is recommended to be always active is the Confirm Exit When Closing Last Window option. This is a good safety feature to enable in Zend Studio for Eclipse because there is such potential to have a lot of views open at any one time, and you would not necessarily want to close down the whole environment when you want to close down only a few views.

The next branch, Web Browser, allows you to control the internal browser that Zend Studio for Eclipse uses. You can merely use the internal browser or select an external
one to use. The three browser options that are initially offered to you are the default system browser (one your environment always launches when you ask for something from the Web), Firefox, and Internet Explorer. You can also define another browser to use if you like (Opera) and mark it as the default. Of course, the internal browser has a few advantages to that of the external one: You don’t have to leave the Zend Studio for Eclipse environment, and it resolves and displays content more quickly.

The Welcome section is the place where you control how Zend Studio for Eclipse starts up and what features are displayed to you when you begin to use the IDE for your day’s work. You can set it up to look almost browserlike with different quadrants for the initial splash screen, or you can simply ignore all this and get right into the IDE with the PHP perspective as your initial starting point.

The last area of interest under the General level of Preferences is labeled Workspace. On this branch you can set some overall workspace features such as when to build your projects (automatically) and whether you should save all open files before a build. There are also some sub-branches here where you can adjust your language settings, determine your project build order (good if one project depends on another one and needs to have one built before it), manage any linked resources that are associated with the workspace, and manage the local saved files history (how long to keep them, how many saves per file to record, and how much space to reserve for the file size).

This has been a rather lengthy review of just the General options that Zend Studio for Eclipse has at its disposal. If you are looking at some of the figures included so far, you will notice that we have gone down only one of the many different branches in the preferences tree. For the sake of space and the overall purpose of this book, we are skipping over all the other main tree branches and now focusing only on the PHP preferences branch.

**PHP Preferences**

Under the PHP branch, as the preference’s name suggests, you adjust and maintain the options available to you in Zend Studio for Eclipse that directly affect the PHP portion of the environment. There is a lot to be done and a great deal that is within your power to control.

When you click on the PHP main branch level of the preferences, you see an option on what will happen when you double-click on something in the PHP Explorer. You can either have Zend Studio for Eclipse go into the selected element or have the element expanded if that is possible. If you have the Link with Editor option also turned on in the IDE, you see duplicating actions because the link activates on the first click of the mouse and then the double-click action takes place.

**Code Analyzer**

The second major section in the PHP preferences branch is labeled Code Analyzer. Here, you can control the severity levels of the anything that the analyzer picks up and give them one of three levels: error, warning, or ignore. You can even do this on a
The code analyzer scans through your open code as you create it or as you import or modify existing code. Any issues that it comes upon are reported in the Problems view and are also displayed in the code editor as errors (in red usually) or warnings (in yellow). The option here is to be able to turn off certain errors that may occur repeatedly or raise lighter issues to full-scale error codes; it is up to you and you can define each code issue to be different.

**Code Coverage**
The Code Coverage portion of the PHP preferences area is merely a representation of what will be displayed when you run the code profiler. The code that has been covered is shown with different shading than code that has not been covered. The colors that can be used are available in the General Preferences section; locate Appearance, Colors and Fonts, and then expand the PHP Debug Tree option.

**Code Gallery**
Code Gallery is the next section available to you for control and management. Here, you can select from your own code library of classes or functions and one supplied online by Zend. You can also include others that may exist by using the Add functionality. After these galleries are activated within your copy of Zend Studio for Eclipse, you can choose code from within them for use in your own projects.

**Debug**
Because a large part of Zend's fame is based on its rock-solid PHP debugger technology, you would expect a lot of options in the Debug portion for you to manage. This is indeed the case with the Debug options under the PHP branch within Zend Studio for Eclipse. Figure 3.9 shows the main level of the debugger preferences. Here, you can select the default server that you want to run the debugger through and what version of PHP you will be debugging within.

Within the PHP Debug branch are two sub-branches labeled Installed Debuggers and Workbench Options. Within the Installed Debuggers branch you can manage the finer details of the debugger that is in use by selecting it and clicking on Configure. There you can change the Debug Port, the client Host/IP number and a few other options. If you had a different debugger definition installed here, this is the place that you would adjust its settings as well.

Under the branch called Workbench Options you can make some choices about how the debugger will react when it is invoked. You can tell Zend Studio for Eclipse to revert to the PHP perspective when the debugger is finished and you can allow multiple debug sessions to run concurrently just to name the first two.

**Editor**
Moving on from the Debug options, we next go to the Editor options branch. These options are specific to the editing of PHP code. Other, more general editor options are controlled under the General Preferences branch discussed earlier in this chapter.
Code Assist

In the Code Assist section of the Editor preferences, you can control how and when the code promptings of PHP are activated. As you can see in Figure 3.10, there are a number of items that you can control. Interesting to note here is the fact that the code assistant is context sensitive. When you are in a code file and there are variables defined in other classes or functions within the same file, they are not, by default, offered within the Code Assist pop-up.

Other options here control other aspects of Code Assist. The biggest option is the one that directs Zend Studio for Eclipse to either overwrite on the insertion of code or insert the completed code. Insertion is the default choice.

Last on this screen is the enabling or disabling of the auto-activation of the code assist itself, and if the auto activation is enabled, how long in milliseconds to delay before the code assistant pops up.

You can also do some filtering within the list of code that is offered during the assist listing of possible hits. You can screen out constants, make the constants’ inclusion be case sensitive, and so on. At the bottom of this window is the control of how the code assistant is launched (automatically and with or without a timer delay). Figure 3.11 shows the code assistant activated when one of the session functions is being coded.

Notice that this code assistant is also available for the PHPDoc portion of your coding. When the PHPDoc borders are defined properly (/**), you can start typing in any of the predefined attributes (@author, @return, @Desc, and so on) of the block, and the code assistant begins to help you right away.
Folding

Code folding is the next option available to you as you travel down the Editor preferences branch. Code folding is the ability to suppress the code that is grouped together in a function or class methods. The benefit is to collapse any code that you are working on that you know is functioning properly because you simply want it out of the way for now. The preferences here are few, in that you can turn the folding feature on or off and that you can initially fold classes, functions, or PHPDoc portions of code. HTML code can also be folded in the code editor, but there are currently no preference features with which to control it.
Hovers
Next on the list of preferences within your domain of control in Zend Studio for Eclipse is the ability to manipulate the hovering effect within the PHP code editor. If you pause (hover) your mouse pointer over a function definition, for example, you see a pop-up box that describes the full definition of that function. And if you press the Control (Ctrl) button at the same time as you hover, you see the full code definition of that function.

Syntax Coloring
Syntax coloring is a great feature for PHP developers. The features within your control are those of the coloring of the code within its contexts. This is more simply known as the content types, and they are all listed in the control box at the top of this window. The PHP tags themselves (Boundary Maker), the HEREDOC designation, the variables, and so on are all colored differently within the code editor. As you can see in Figure 3.12, the comments are bold and italicized.

![Figure 3.12 The Syntax Coloring page of the Preferences window.](image)

If you ever get into trouble with the code colorizations, you can either restore the default settings for the content type in question or restore the default settings for all the content types by choosing the appropriate button at the bottom of this window.

Task Tags
The task tags are the next items within the Editor preferences that are at your disposal to manage. They are the different designations that you can set up within the Task Manager
view. They are the different tags that you assign to the different tasks that may need attention within your projects. You can make them High, Normal, or Low in priority. As shown in Figure 3.13, three task tags are defined in the Preferences window, and all three of them are in use in the code editor and listed in the Tasks view just below that. Additionally, you can see in the code editor that these task tags are activated only within the context of code comments.

![Figure 3.13 The Task Tags page of the Preferences window and tags in use in PHP code editor.](image)

**Typing**

In the next section, called Typing, you are able to control how certain parts of your code can be automatically completed for you. For example, when you begin to code a function and are beginning to open the curly braces, you can have Zend Studio for Eclipse automatically close the braces for you. This feature can be a great convenience, or it can be an annoyance. If you want to enter some other text and are not yet ready to close the brace, you spend just as much time removing the offered closing brace as you would save by having it done for you. On the other hand, how often have you been looking at your code wondering where the syntax error is, just to discover that you did not close an open brace?

Your decision whether to turn on this feature comes down to what you are used to and what you want Zend Studio for Eclipse to do for you. The other items that can be automatically completed for you are strings, round braces, square brackets, and PHPDoc
comment regions. You can also control how the Tab key works within the coding context from this dialog.

**Formatter**

Formatter is next on the list. Specific to the PHP portion of Zend Studio for Eclipse is how the code is indented and formatted when you run the code formatter within the code editor. You can write your code as ugly as you like and then run the code formatter on the open file, and it will perform its cleanup formatting on the code based on your settings here (see Figure 3.14).

![Figure 3.14 Display of selected formatting profile.](image)

What you initially see in this dialog is merely a sample of what your code will look like after it is run through the formatter. To alter the appearance, you need to select the format profile from the drop-down list at the top and then click on Show. In the resulting tabbed dialog, shown in Figure 3.15, you have total control of how your braces are indented, if your braces should be on the same line as the defining entity, where you want whitespace, and quite a lot more. When your changes are ready, just save them, and you return to the displaying dialog to see what it will all look like.

You can even define your own code formatting profile by clicking on the New button and using an already-defined profile as a starting point. You also have to name your new profile before you can begin altering your settings. And, as is true for many of these preference settings, you can also make a formatting profile on a project-level basis by using the link at the top right of this dialog.
All the above subsections in the Editor preferences area are merely directives that tell Zend Studio for Eclipse where these features are located. With a few exceptions, these items are simply told where (what path) their information or functionality lies. For example, the PHP Executables option just needs a name for the executable in question and its path location.

Skipping a few of the other minor branch options entirely, we will next touch on a few of the remaining less significant ones to make you aware of them and to briefly comment on their functionality.

**Profiler**

The Profiler option merely wants to know if it should automatically switch to the profile perspective when you start a profiling session. You can have it start, not start, or ask you each time.

**Templates**

The Templates option provides another great potential time-saver for PHP developers. The code templates section allows you to set up code frameworks or skeletons and incorporate them into your code as you write by using a simple keyboard combination. When these templates are built, they are invoked in concert with the code assist process.

If you are writing a lot of code, for example, that often uses an `if` then construct, you can set up a template called `ifthen` and define the skeleton of the code there, as shown in Figure 3.16. You can even insert standard values like a reference to today’s date and where you want the cursor to land after the block of code is inserted into the PHP editor.
To call these templates after they are defined, begin to type their name, and you see that name offered to you in the code assistant. When you have it in focus, press Enter to select that template, and it is inserted into your editor at the current location of the cursor.

These templates can also be imported and exported so that other developers or team members who are interested in your templates can use them.

**Summary**

This chapter really only scratched the proverbial surface of how you can manipulate Zend Studio for Eclipse to be a totally customizable Integrated Development Environment. We hope we have shown you enough information in this area of Zend Studio for Eclipse that you now feel comfortable with changing some of these features and controls yourself. Don’t be afraid to experiment with these features either, because there is quite often a way provided that will bring you back to the starting point.
Index

A

add function, project creation example, 163-165
advanced local variables, 73
Advanced tab (Properties view), 126
Always Run in Back-ground checkbox (General Preferences window), 32
Annotations area (Text Editors tab), 36
Appearance tree item (General Preferences window), 34
application creation example, 145
  Company Information Table, 146
  Country Table, 147
  Event Details Table, 147
  Event Information Table, 146
functionality, adding
  add function, 163-165
  database access, 158-160
  delete function, 160-162
  edit function, 165-167
  model creation, 157-158
Hello World scripts, 152
People Information Table, 146
run configuration setup, 152
server setup, 152
System Users Information Table, 147
table creation SQL, 148-149
//TODO commenting system, 157
writing code
  bootstrap files, 152-154
  configuration files, 154
  dashboard, 156
  Login view, 155-156
Zend Framework project creation, 151

B - C

bootstrap files, project creation example, 152-154
breakpoints, 106
  conditional breakpoints, 107
  importing/exporting, 107
  local debugging, 102
Breakpoints view (PHP Debug perspective), 106
Browser Output view (Console), 4, 91-93

classes, refactoring, 78-79
code
  folding, 12, 41, 48
  writing (project creation example)
    bootstrap files, 152-154
    configuration files, 154
    dashboard, 156
    Login view, 155-156
code analyzer, 38
  configuring, 67-69
  error color designations, 70
  functions of, 67
  security risk warnings, 70
tool tips, 70
  visual elements of, 70
Code Assist section (Editor section), 40
code assistant, 49-50, 125-126
code completion feature (code editor), 11
Code Coverage section (PHP Preferences window), 39
code editor, 47. See also views
code assistant, 49-50
code completion feature, 11
code error icons, 48
code formatter, 50
code range in focus, 48
cold folding, 12, 48
debugger breakpoint icons, 48
HTML code, 48
line number column, 48
marker bar, 48
navigating, 12
PHP code, 48
Properties view, 51
suspect code range indicators, 48
underlined code, 48
code error icons (code editor), 48
Code Folding section (Editor section), 41
code formatter (code editor), 50
Code Gallery section (PHP Preferences window), 39
Code Gallery view, 63-64
code range in focus (code editor), 48
Colors and Fonts tab (Appearance tree item), 34
Company Information Table (project creation example), 146
Compare/Patch tree item (General Preferences window), 34
conditional breakpoints, 107
configuration files, project creation example, 154
configuring
code analyzer, 67-69
views as perspectives, 29
working sets, 21-22
Confirm Exit When Closing Last Window option (Startup and Shutdown section), 37
Console
Browser Output view, 91-93
Debug Output view, 91-93
Content Types tree item (General Preferences window), 35
Country Table (project creation example), 147
CRM (Customer Relationship Management) application creation example, 145
  Company Information Table, 146
  Country Table, 147
Event Details Table, 147
Event Information Table, 146
functionality, adding
  add function, 163-165
database access, 158-160
delete function, 160-162
edit function, 165-167
model creation, 157-158
Hello World scripts, 152
People Information Table, 146
run configuration setup, 152
server setup, 152
System Users Information Table, 147
table creation SQL, 148-149
//TODO commenting system, 157
writing code
  bootstrap files, 152-154
customizing
configuration files, 154
dashboard, 156
Login view, 155-156
Zend Framework project creation, 151
CSS selector (HTML Editor view), 123
database queries, writing SQL queries, 86
database connections, 83-85
Hello World scripts, 152
dashboard, project creation example, 156
data members, refactoring, 78
database queries, writing SQL queries, 86
Database Development perspective
database connections, 83-85
New JDBC Connection Profile, 83
CVS, 113-118
cycling through
dashboard, project creation example, 156
data members, refactoring, 78
Database Development perspective
database connections, 83-85
New JDBC Connection Profile, 83
data queries, writing SQL queries, 86
CVS, 113-118
cycling through
files in editor, 33
perspectives in editor, 33
views in editor, 33
Debug Output view

Console, 91-93
PHP Debug perspective
Resume command, 110
Step Into command, 109
Step Over command, 110
Step Return command, 110
Debug section (PHP Preferences window), 39
debugger breakpoint icons (code editor), 48
debugging
breakpoints, 106-107
customizing preferences, 101
local debugging, 102
PHP Debug perspective, 105
Breakpoints view, 106
Debug Output view, 109-110
Expressions view, 111
main activity view, 109
Parameter Stack view, 112
starting sessions, 107
Step Into command, 109
stopping sessions, 109
Variables view, 110
PHPUnit, 91
creating unit tests, 95
project planning, 94
running unit tests, 95-99
selecting project elements to test, 95
test suites, 99
printing output, 91-93
remote debugging, 103
delete function, project creation example, 160-162
Design tab (HTML Editor view), 123-124
dummy.php files, remote debugging, 103

delete function, project creation example, 160-162
Design tab (HTML Editor view), 123-124
dummy.php files, remote debugging, 103

dummy.php files, remote debugging, 103

event Details Table (project creation example), 147
Event Information Table (project creation example), 146
example of project creation, 145
Company Information Table, 146
Country Table, 147
Event Details Table, 147
Event Information Table, 146
functionality, adding
add function, 163-165
database access, 158-160
delete function, 160-162edit function, 165-167
model creation, 157-158
Hello World scripts, 152
People Information Table, 146
run configuration setup, 152
server setup, 152
System Users Information Table, 147
table creation SQL, 148-149
//TODO commenting system, 157
writing code
  bootstrap files, 152-154
  configuration files, 154
dashboard, 156
Login view, 155-156
Zend Framework project creation, 151
exporting/importing
  breakpoints, 107
  projects, 23
Expressions view (PHP Debug perspective), 111

filters
  Problems View, 58
  Tasks View, 57
folding code, 12, 41, 48
Formatter section (PHP Preferences window), 44-45
formatting code, code formatter (code editor), 50
functionality, adding to project creation example
  add function, 163-165
database access, 158-160
delete function, 160-162edit function, 165-167
model creation, 157-158
functions, refactoring, 78-79

G

General Preferences window
  Always Run in Back-ground checkbox, 32
  Appearance tree item, 34
  Compare/Patch tree item, 34
  Content Types tree item, 35
  Editors section, 35-36
  Keep Next/Previous Part Dialog Open checkbox, 32
  Keyboard Mapping section, 36
  Perspectives section, 37
  Search section, 37
  Show Heap Status checkbox, 32
  Startup and Shutdown section, 37
  View Cycling dialog, 34
  Web Browser section, 37-38

F

Fast View, 5
File Associations tab (Editors section), 36
File Content configuration, 113
files
  creating, 24
cycling through, 33
Link with Editor feature (PHP Explorer View), 25
moving between, 25
refactoring, 79-80
remote files, accessing, 26-27
storing, PHP Project Wizard, 16
filtering projects in PHP Explorer View, 19
Welcome section, 38
Workspace section, 38

General tab
  Compare/Patch tree item, 34
  Properties view, 126

global variables, refactoring, 77-78

Guard (Zend)
  functions, 130
  installing, 129
  launching, 130
  operational requirements, 129
  project creation, 130
  troubleshooting, 132

Hello World scripts, project creation example, 152

Hiding/showing projects in PHP Explorer View, 19

Hovers section (Editor section), 42

HTML code, code editor, 48

HTML Editor view (PHP/HTML WYSIWYG Editor), 122
  Design tab, 123-124
  importing images into, 123
  Preview tab, 124
  Source tab, 123-126

HTML files, opening in PHP/HTML WYSIWYG Editor, 121

Ignored Resources configuration, 114

Image Insertion Wizard, launching, 123

images, importing into HTML Editor view (PHP/HTML WYSIWYG Editor), 123

importing/exporting
  breakpoints, 107
  images into HTML Editor view (PHP/HTML WYSIWYG Editor), 123
  projects into PHP Explorer View, 19, 22

installing
  plug-ins in Zend Studio for Eclipse, 170-171
  Zend Framework, 135
  Zend Guard, 129
  Zend Optimizer, 129

Keep Next/Previous Part Dialog Open checkbox (General Preferences window), 32

Keyboard Mapping section (General Preferences window), 36

Label Decorations tab (Appearance tree item), 34

launching
  Image Insertion Wizard, 123
  WYSIWYG designer, 121
  Zend Guard, 130

line number column (code editor), 48

Link with Editor feature (PHP Explorer View), 25

local debugging, 102

local variables
  advanced local variables, 73
  naive local variables, 73
  refactoring, 73-76

Login view, project creation example, 155-156
M - N

main activity view (PHP Debug perspective), 109
marker bar (code editor), 48
master perspectives. See presentations, 34
methods, refactoring, 78-79
Models configuration, 114
moving views, 57
MVC (Model-View-Controller) design, 136, 154

naive local variables, 73
naming projects, 16
navigating code editor, 12
New JDBC Connection Profile (Database Development perspective), 83
New SQL File dialog, writing SQL database queries, 86
New Zend Controller Wizard (Zend Framework), 137

O - P

Optimizer (Zend)
installing, 129
troubleshooting, 132
Outline View, 3, 8, 53, 59-61

Parameter Stack view (PHP Debug perspective), 112
People Information Table (project creation example), 146

perspectives
configuring views as, 29
customizing, 7, 30
cycling through, 33
Database Development perspective, 83-85
defining, 6
master perspectives. See presentations, 34
PHP Debug perspective, 105
  Breakpoints view, 106
  Debug Output view, 109-110
  Expressions view, 111
  main activity view, 109
  Parameter Stack view, 112
  starting sessions, 107
  Step Into command, 109
  stopping sessions, 109
  Variables view, 110
PHP perspective, 6-7
presentations, 34
saving, 29
Perspectives section (General Preferences window), 37

PHP code, code editor, 48

PHP Debug perspective, 105
  Breakpoints view, 106
  Debug Output view
    Resume command, 110
    Step Into command, 109
    Step Over command, 110
    Step Return command, 110
  Expressions view, 111
  main activity view, 109
  Parameter Stack view, 112
starting sessions, 107
Step Into command, 109
stopping sessions, 109
Variables view, 110

**PHP Explorer View, 3**
defining working sets in, 22
file creation, 24
filtering projects in, 19
hiding/showing projects in, 19
importing projects into, 19, 22
Link with Editor feature, 25
removing projects from, 19

**PHP Functions View, 54**

**PHP perspective, 6-7**

**PHP Preferences window**
  Code Analyzer section, 38
  Code Coverage section, 39
  Code Gallery section, 39
  Debug section, 39
  Editor section, 39
    Code Assist section, 40
    Code Folding section, 41
    Hovers section, 42
    Syntax Coloring section, 42
    Task Tags section, 42
    Typing section, 43-44
  Formatter section, 44-45
  Profiler section, 45
  Templates section, 45

**PHP Project Outline View, 4, 54, 59-60**

**PHP Project Wizard**
  changing default settings, 17
  Enable Project Specific Settings checkbox, 17
  file storage, 16
  launching, 15
  multiproject support, 19
  naming projects, 16
  selecting supporting libraries/projects, 17
  Workspace, overriding settings, 17
  Zend Framework Library projects, 18

**PHP/HTML WYSIWYG Editor, 121**
  HTML Editor view, 122
    Design tab, 123
    Design/Source tab, 124
    importing images into, 123
    Preview tab, 124
    Source tab, 123-126
  HTML files, opening in, 121
  Properties view, 126

**PHPUnit, 91**
  project planning, 94
test suites, 99
unit tests
  creating, 95
  running, 95-99
  selecting project elements to test, 95

**Platform (Zend), 132**

**plug-ins**
  adding to Zend Studio for Eclipse, 170-173
  web resources, 170

**preferences**
  General Preferences window
    Always Run in Background checkbox, 32
    Appearance tree item, 34
Compare/Patch tree item, 34
Content Types tree item, 35
Editors section, 35-36
Keep Next/Previous Part Dialog Open checkbox, 32
Keyboard Mapping section, 36
Perspectives section, 37
Search section, 37
Show Heap Status checkbox, 32
Startup and Shutdown section, 37
View Cycling dialog, 34
Web Browser section, 37-38
Welcome section, 38
Workspace section, 38
PHP Preferences window
   Code Analyzer section, 38
   Code Coverage section, 39
   Code Gallery section, 39
   Debug section, 39
   Editor section, 39-44
   Formatter section, 44-45
   Profiler section, 45
   Templates section, 45

Preferences management window, 12-13.
   See also customizing, Zend presentations, 34
   Preview tab (HTML Editor view), 124
   printing output as debugging/testing method, 91-93
Problems View, 4, 55-58
Profiler section (PHP Preferences window), 45
project creation example, 145
   Company Information Table, 146
   Country Table, 147
   Event Details Table, 147
   Event Information Table, 146
functionality, adding
   add function, 163-165
   database access, 158-160
   delete function, 160-162
   edit function, 165-167
   model creation, 157-158
Hello World scripts, 152
People Information Table, 146
directory configuration setup, 152
server setup, 152
System Users Information Table, 147
table creation SQL, 148-149
   //TODO commenting system, 157
writing code
   bootstrap files, 152-154
   configuration files, 154
   dashboard, 156
   Login view, 155-156
   Zend Framework project creation, 151

projects
   customizing settings, 17
   exporting, 23
   file creation, 24
   file storage, 16
   filtering in PHP Explorer View, 19
   hiding/showing in PHP Explorer View, 19
   importing into PHP Explorer View, 19, 22
   multiproject support, 19
   naming, 16
removing from PHP Explorer View, 19
selecting supporting libraries/projects, 17
Zend Framework Library projects, 18
projectwide renaming. See refactoring
Properties view
code editor, 51
PHP/HTML WYSIWYG Editor, 126

Q - R

reducing code. See folding code
refactoring
classes, 78-79
data members, 78
files, 79-80
folders, 79-80
functions, 78-79
methods, 78-79
requirements for, 73
variables
global variables, 77-78
local variables, 73-76
remote debugging, 103
Remote Systems View
remote files, accessing, 26-27
remote sites, defining, 26
renaming. See refactoring
Resume command (Debug Output view), 110

S

sample project creation, 145
Company Information Table, 146
Country Table, 147
Event Details Table, 147
Event Information Table, 146
functionality, adding
add function, 163-165
database access, 158-160
delete function, 160-162edit function, 165-167
model function, 157-158
Hello World scripts, 152
People Information Table, 146
run configuration setup, 152
server setup, 152
System Users Information Table, 147
table creation SQL, 148-149
//TODO commenting system, 157
writing code
bootstrap files, 152-154
configuration files, 154
dashboard, 156
Login view, 155-156
Zend Framework project creation, 151

saving perspectives, 29
scripts, running versus unit testing, 91
SDK (Software Development Kits), Zend
Studio for Eclipse plug-in installation, 172
Search section (General Preferences window), 37
Show Heap Status checkbox (General Preferences window), 32
Show Multiple Editor Tabs option (Editors tree item), 35
Show View command, 57
Source tab (HTML Editor view), 123-126
SQL integration
data manipulation, 88
database connections
establishing via Database Development perspective, 83
naming in Database Development perspective, 83
specifying in Database Development perspective, 85
viewing summaries of in Database Development perspective, 83-85
New SQL File dialog, writing SQL database queries, 86
table manipulation, 88
testing SQL via SQL Scrapbook (SQL Perspectives toolbar), 88
viewing data, SQL Results view, 86
SQL Perspectives toolbar, 88
SQL Results view, 86
SQL Scrapbook (SQL Perspectives toolbar), testing SQL, 88
starting
debugging sessions, 107
Image insertion Wizard, 123
WYSIWYG designer, 121
Zend Guard, 130
Startup and Shutdown section (General Preferences window), 37
Step Into command
Debug Output view, 109
PHP Debug perspective, 109
Step Over command (Debug Output view), 110
Step Return command (Debug Output view), 110
storage (files), PHP Project Wizard, 16
Structured Text Editors tab (Editors section), 36
suspect code range indicators (code editor), 48
SVN, 113-114
Syntax Coloring section (Editor section), 42
System Users Information Table (project creation example), 147

Tables
adding to HTML Editor view design area, 123
Company Information Table (project creation example), 146
Country Table (project creation example), 147
creation SQL (project creation example), 148-149
Event Details Table (project creation example), 147
Event Information Table (project creation example), 147
People Information Table (project creation example), 146
System Users Information Table (project creation example), 147
Task Tags section (Editor section), 42
Tasks View, 4, 56-57
Templates section (PHP Preferences window), 45
testing

printing output, 91-93
SQL, SQL Scrapbook (SQL Perspectives toolbar), 88
unit testing
   PHPUnit, 91, 94-99
      running scripts versus, 91
Text Compare tab (Compare/Patch tree item), 34
Text Editors tab (Editors section), 36
third-party plug-ins
   adding to Zend Studio for Eclipse, 170-173
      web resources, 170
//TODO commenting system, project creation example, 157
troubleshooting
   Zend Guard, 132
      Zend Optimizer, 132
Typing section (Editor section), 43-44

variables
   global variables, refactoring, 77-78
   local variables
      advanced local variables, 73
      naive local variables, 73
      refactoring, 73-76
Variables view (PHP Debug perspective), 110
version control integration
   CVS, 113-118
      File Content configuration, 113
      Ignored Resources configuration, 114
      Models configuration, 114
      open source server downloads, 113
   SVN, 113-114
View Cycling dialog (General Preferences window), 34
views. See also code editor
   available views list, 5
      Breakpoints view (PHP Debug perspective), 106
      Browser Output View, 4, 91-93
   code element representation example, 59-60
   Code Gallery view, 63-64
      configuring as perspectives, 29
      cycling through, 33
      Debug Output view
         Console, 91-93
            PHP Debug perspective, 109-110
      defining, 3
      Expressions view (PHP Debug perspective), 111
      Fast View, 5

underlined code in code editor, 48

unit testing
   PHPUnit, 91
      creating tests, 95
      project planning, 94
      running tests, 95-99
      selecting project elements to test, 95
      test suites, 99
      running scripts versus, 91
updates, 169
HTML Editor view (PHP/HTML WYSIWYG Editor), 122
  Design tab, 123
  Design/Source tab, 124
  importing images into, 123
  Preview tab, 124
  Source tab, 123-126
Login view, project creation example, 155-156
main activity view (PHP Debug perspective), 109
moving, 57
Outline View, 3, 8, 53, 59-61
Parameter Stack view (PHP Debug perspective), 112
perspectives, 6
PHP Explorer View, 3
  defining working sets in, 22
  file creation, 24
  filtering projects in, 19
  hiding/showing projects in, 19
  importing projects into, 19, 22
  Link with Editor feature, 25
  removing projects from, 19
PHP Functions View, 54
PHP Project Outline View, 4, 54, 59-60
Problems View, 4, 55-58
Properties view
code editor, 51
  PHP/HTML WYSIWYG Editor, 126
Remote Systems View, 26-27
Show View command, 57
SQL Results view, 86
Tasks View, 4, 56-57
Variables view (PHP Debug perspective), 110
Zend Framework, creating via, 139

W
Web Browser section (General Preferences window), 37-38
web resources, Zend Studio for Eclipse, 173-174
Welcome section (General Preferences window), 38
wizards
  Image Insertion Wizard, launching, 123
  PHP Project Wizard
    changing default settings, 17
    Enable Project Specific Settings checkbox, 17
    file storage, 16
    launching, 15
    multiproject support, 19
    naming projects, 16
    overriding Workspace settings, 17
    selecting supporting libraries/projects, 17
    Zend Framework Library projects, 18
  Zend Controller Wizard (Zend Framework), 137
working sets
  configuring, 21-22
  defining, 9-10
  selecting, 10
Workspace section (General Preferences window), 17, 38
writing
code (project creation example)
  bootstrap files, 152-154
  configuration files, 154
dashboard, 156
  Login view, 155-156
SQL database queries, New SQL
  File dialog, 86
WYSIWYG (What You See Is What You
  Get) designer, 121

Z
Zend code gallery, 65
Zend Framework
  content separation, 139-140
  controller creation, 137
directory creation, 139
directory structure of, 136-137
downloading, 135
  installing, 135
MVC (Model-View-Controller)
  design, 136, 154
  project creation, 136-137
  project creation example, 151
  view creation, 139
Zend Controller Wizard, 137
Zend_Acl library, 141
Zend_Controller library, 142
Zend_Db library, 142-143
Zend_Gdata library, 143
Zend_Mail library, 143-144
Zend_Pdf library, 144
Zend_Service library, 144

Zend Guard
  functions of, 130
  installing, 129
  launching, 130
  operational requirements, 129
  project creation, 130
  troubleshooting, 132
Zend Optimizer
  installing, 129
  troubleshooting, 132
Zend Platform, 132
Zend Studio for Eclipse
  adding plug-ins, 170-173
  updates, 169
  web resources, 173-174

Zend Framework Library, project creation,