The PHP Company

White Paper:

Team Development with Zend Studio for Eclipse

Technologies

January 2008

Table of Contents

Introduction

PHP Development efforts generally start with a single developer or a small team with a similarly small task. Projects tend to grow in both complexity and size as one good idea gets expanded and enlarged into a web application.

As a project gains size and becomes more successful, development teams need to scale (or should scale) to a larger number of developers at the same time. Further complexity often arises as the application is deployed at many sites with different versions of the developed product. One install may want special enhancements thus creating its own application branch.

The Problem

In order to grow business, a development company needs to increase both the number of customer site installations and proportionately, the number of developers. Typical PHP development efforts do not generally scale in both of these areas at the same time. However, as your development team increases there are many aspects of development that should be looked at more closely. Things like version control, team development environments, shared code libraries, and common coding styles, just to name a few.

Many developers working on the same project can cause problems on a shared development infrastructure since, as one example, an error introduced by one programmer has the potential to ripple through the entire project. When something like this occurs all other developers may need to suspend their work until the error is located and corrected.

In this paper we will be looking at the answers to the aforementioned issues within the context of Zend Studio for Eclipse. We will be looking at many team development issues that can be resolved by using this new Integrated Development Environment (IDE). We will be doing this by posing common development scenarios and then discovering how Zend Studio for Eclipse helps to alleviate the issues.

Source Code Management

On a basic level of team development the use of Source Code Management (SCM) tools are invaluable tools that really must be used for a development team to be productive in any way. The two main SCM products on the market today are CVS and SVN (aka: SubVersion) Zend Studio for Eclipse has integration architecture that accommodates both of these tools. The ability to control who is doing what to a web application is paramount in a team environment. There is nothing worse in this context than having a few days worth of development work being over written by a co-developer. Locking files, having local development copies, updating, and committing files in a controlled and managed process is the ideal approach and both of these SCM software tools are very capable of providing this need. Also, to have this SCM process integrated within each developer's IDE is all that much the better. Now let's look at some situations that a software development team is likely to find themselves in.

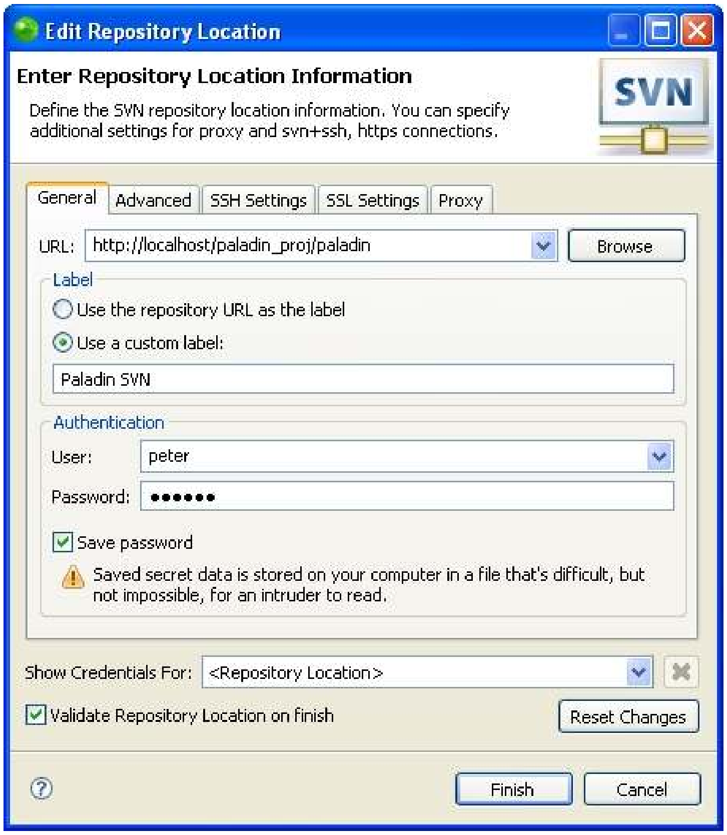
ℹ Zend Studio for Eclipse also integrates with the source control product known as CVS, however the SubVersion product is the focus of the SCM discussions in this paper.

Scenario 1

Your development efforts are ramping up for you next version of your web application. There is a development server that has just been setup with Subversion (SVN) source control software and your team of 8 programmers is about to start their coding efforts.

Solution Discussion

Before you start to write any code you should take the time to sit down with the entire development team and plan your approach. Your main issue here is that of coordinating the development of code among your entire team. You would certainly not want to have days of work potentially over-written by another programmer who is not aware of the work recently completed by a programmer in a different office. Luckily in this situation you have some source code control software installed to assist with this process. In this example, the tool is Subversion so we will see how Zend Studio for Eclipse makes use of this connection.



Enter Repository Location Information

Define the SVN repository location information. You can specify additional settings for proxy and svn+ssh, https connections.

URL: http://localhost/paladin_proj/paladin

Label

○ Use the repository URL as the label

● Use a custom label:

Paladin SVN

Authentication

User: peter

Password: ●●●●●●

☑ Save password

Saved secret data is stored on your computer in a file that's difficult, but not impossible, for an intruder to read.

Show Credentials For: <Repository Location>

☑ Validate Repository Location on finish

Figure 1 - Establishing a Connection to an SVN Repository Server.

Now lets take a look at how Zend Studio for Eclipse integrates with SVN and allows your programmer team to not get into each other's way while performing their respective programming tasks.

Zend Studio for Eclipse has a number of views that are specifically designed for working within the SVN parameters. In fact they are collected into a designated perspective called the SVN Repository Exploring perspective. The views in this perspective are primarily used to interact with the repository itself and not the checked out versions that your programmers will be working with on a day-to-day basis. The repository is the location of the protected source code that you would normally interact with via the SVN commands of commit, update, and so on.

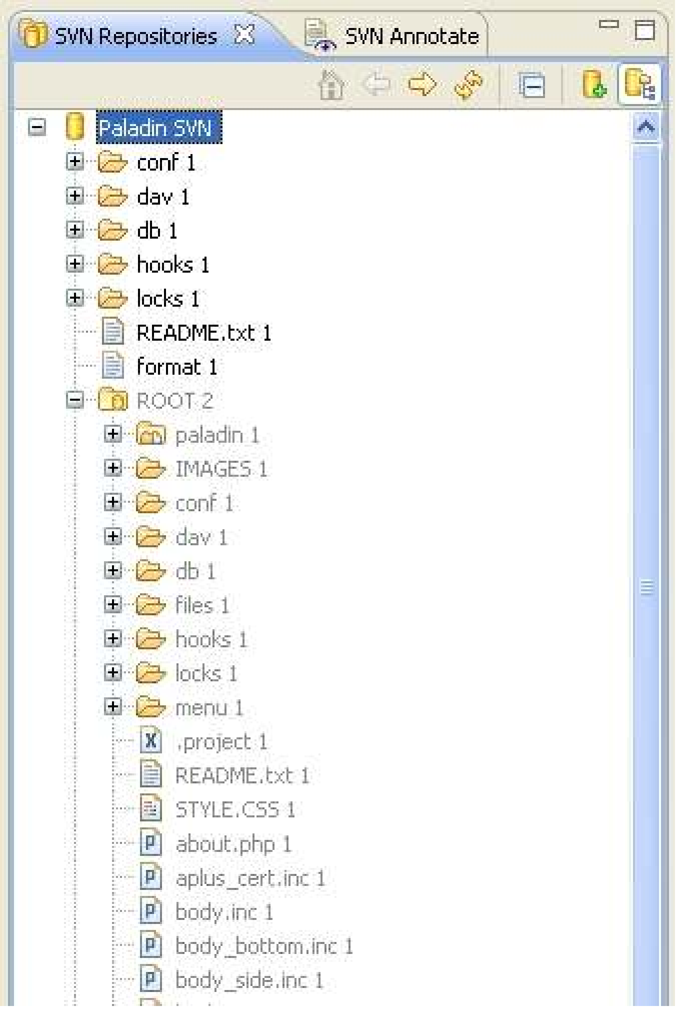
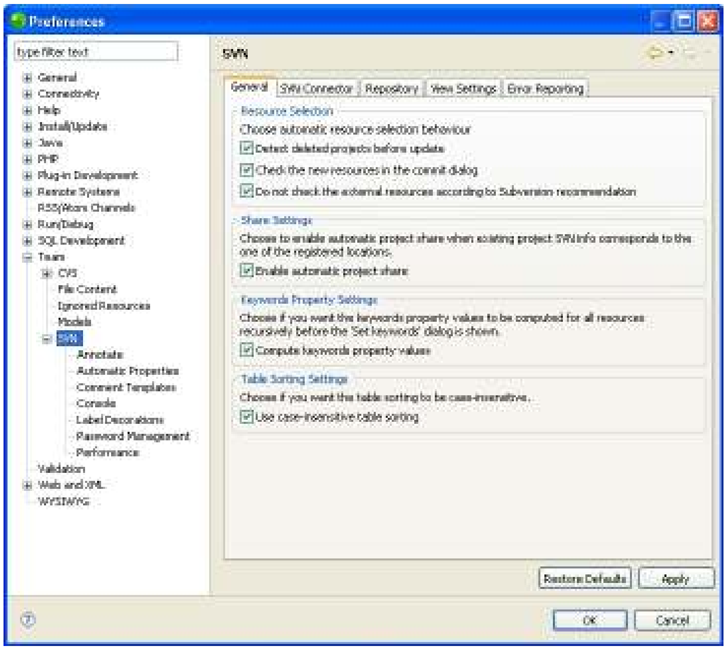


Figure 2 - Browsing a Repository of Files

As you are establishing your connection to the repository of code files however, it is a good idea to access this perspective and make your initial connections between the repository server and your local copy of the development environment. To do this you would right-click on the SVN Repositories view and select new from the pop-up menu. You will see a screen similar to that shown in figure 1 where you will provide all your connection coordinates for the server that the repository is being housed on. Once this connection is established you will see the repository information displayed in the SVN Repositories view. Again this is just a display of the protected source code in the repository itself, you can look at the files and even open them for detailed review, but you cannot edit them directly here. The repository display looks like that shown in Figure 2.

Once the connection to the repository is established you can then create an SVN Project and checkout your own files from the repository to make your own working copy. Then you can begin to work in your own environment and as code segments are completed you can commit them to the repository.

We will take a look at how to do this a little later, but first lets see how the over all SVN environment can be managed with the help of the Zend Studio for Eclipse preferences. If you click on Window then Preferences in the menu system you will be taken to the interface where you can control all the settings of Zend Studio for Eclipse. Specifically in this case, there is a tree segment called Team and under that option is one called SVN. Here you can control many aspects of how Zend Studio for Eclipse will interact with the SVN environment. This is shown in figure 3.



Resource Selection

Choose automatic resource selection behaviour

☑ Detect deleted projects before update

☑ Check the new resources in the commit dialog

☑ Do not check the external resources according to Subversion recommendation

Share Settings

Choose to enable automatic project share when existing project SVN info corresponds to the one of the registered locations.

☑ Enable automatic project share

Keywords Property Settings

Choose if you want the keywords property values to be computed for all resources recursively before the 'Set keywords' dialog is shown.

☑ Compute keywords property values

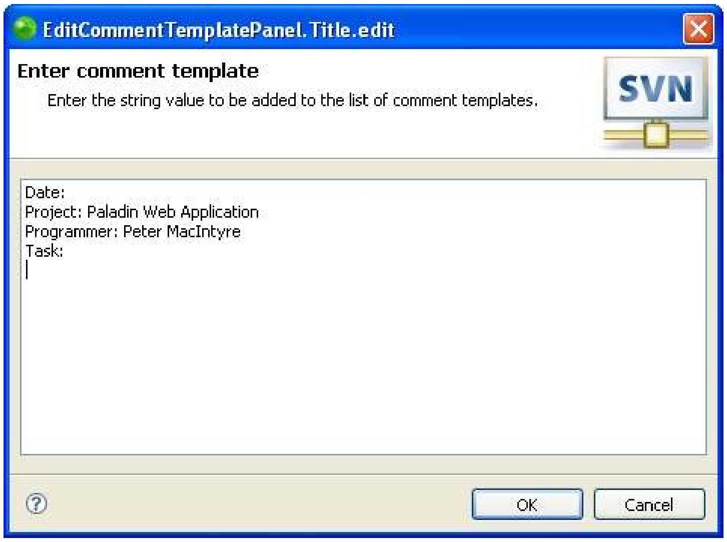
Table Sorting Settings

Choose if you want the table sorting to be case-insensitive.

☑ Use case-insensitive table sorting

Figure 3 - SVN Preferences Window

There are a lot of options here to look at but we want to focus on just one of them here in this paper. The item of note here is that of the Comment Templates. This feature allows the developers to establish and use a pre-defined block of text that can be used each time code is committed to the repository. This makes for standard and uniform comments that everyone on the development team can read and understand. A sample Comment Template is shown in figure 4 and its subsequent use in the commit process is shown in figure 5.

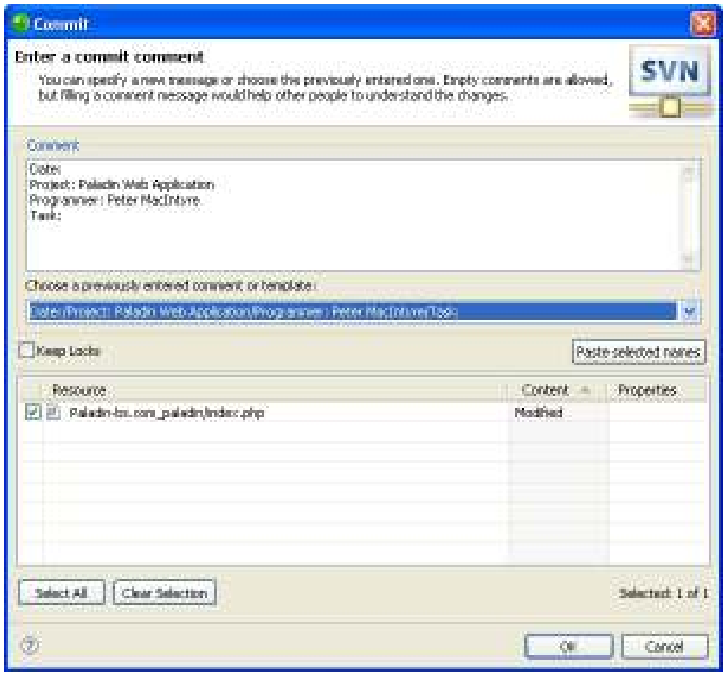


Enter comment template

Enter the string value to be added to the list of comment templates.

```
Date:
Project: Paladin Web Application
Programmer: Peter MacIntyre
Task:
```

Figure 4 - Creation of a Comment Template



Enter a commit comment

You can specify a new message or choose the previously entered one. Empty comments are allowed, but filling a comment message would help other people to understand the changes.

Comment

Date:
Project: Paladin Web Application
Programmer: Peter MacIntyre
Task:

Choose a previously entered comment or template:

Date:/Project: Paladin Web Application/Programmer: Peter MacIntyre/Task:

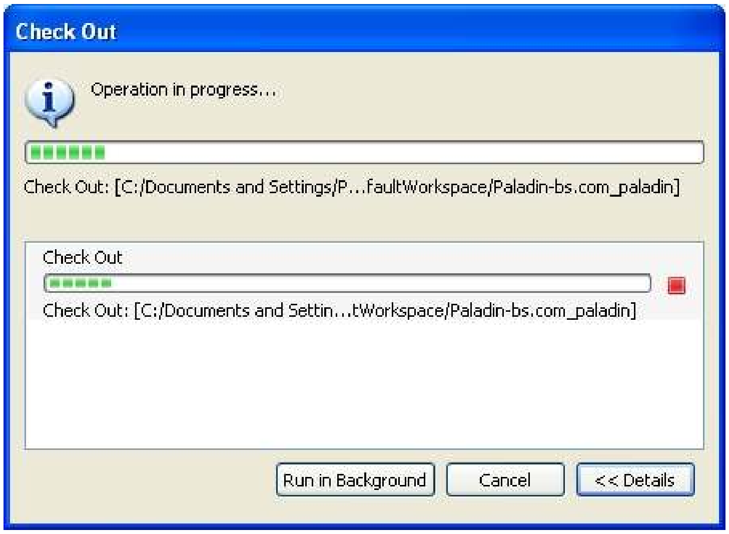
□ Keep Locks   Paste selected names

Resource	Content	Properties
☑ Paladin-bs.com_paladin/index.php	Modified	

Selected: 1 of 1

Figure 5 - Using the Comment Template when Committing a File to the Repository

Now to get back to the checking out process of the files from the repository and creating your own working copy we will set up an SVN project at the same time. This process is best performed in the PHP perspective, so if you are following along with this paper, switch back to the PHP Perspective and select new from the file menu and then select Other from the resulting menu. This will open the New project wizard and from here you simply open the SVN tree item and select "Projects from SVN" to begin the process. After selecting this option the wizard takes you to the check out screen where it will display all the SVN repositories that it knows about and has valid connections to so that you can choose which of them you want to make your working copy out of. Once you select the source of your files to check out and name your project, the actual checking out of the files begin. Figure 6 shows the checkout process under way.



Operation in progress...

Check Out: [C:/Documents and Settings/P...faultWorkspace/Paladin-bs.com_paladin]

Check Out

Check Out: [C:/Documents and Settin...tWorkspace/Paladin-bs.com_paladin]

Figure 6 - Checking out Files from a Repository into a Working Copy

Now that you have the files into your working copy your programmers will be able to do their local work on the project. You should be aware of some other views when you are working on an SVN sourced project within the PHP Perspective. First, Zend Studio for Eclipse knows that you are using the SVN repository so it opens up the console view the first time you perform an action that interfaces with the repository (like a commit).

Console View - This view is merely the display area where all the SVN activity output is shown. When you do a commit for example you should see something like the following in the console view:

```
  *** Commit

  svn commit "C:\Documents and
Settings\Pbmacintyre\Zend\workspaces\DefaultWorkspace\Paladin-
bs.com_paladin\index.php" -m "Date:

  Project: Paladin Web Application

  Programmer: Peter MacIntyre

  Task:" --username "peter"

                M C:/Documents and
Settings/Pbmacintyre/Zend/workspaces/DefaultWorkspace/Paladin-
bs.com_paladin/index.php

                  Transmitting file data: C:/Documents and
Settings/Pbmacintyre/Zend/workspaces/DefaultWorkspace/Paladin-
bs.com_paladin/index.php

  Committed revision 3

  *** Ok (took 00:01.266)
```

As you can see the Comment Template that was defined earlier is used here as well. Other views that you should be aware of are: History, Synchronize, and SVN Annotate. Lets take a quick look at these views and see what they provide to the developer.

NOTE: to open a view simply click on the window, show view, other menu sequence and the list of all available view will be displayed. Some of the above mentioned views are under the SVN tree and some are under the Team tree.

History View – If you click on the right-mouse menu while focused on a file in the editor and select Compare With, then Latest from Repository, you will be switched over to the history view with a new tab also opening in the editor view. What this process does is connects to the repository and performs a line by line comparison with the file that is in your working copy and the same file that is in the repository. Then Zend Studio for Eclipse shows you the results of this comparison in the editor view area and at the same time displays some historical statistics in the history view. A typical screen of this process is shown in figure 7.

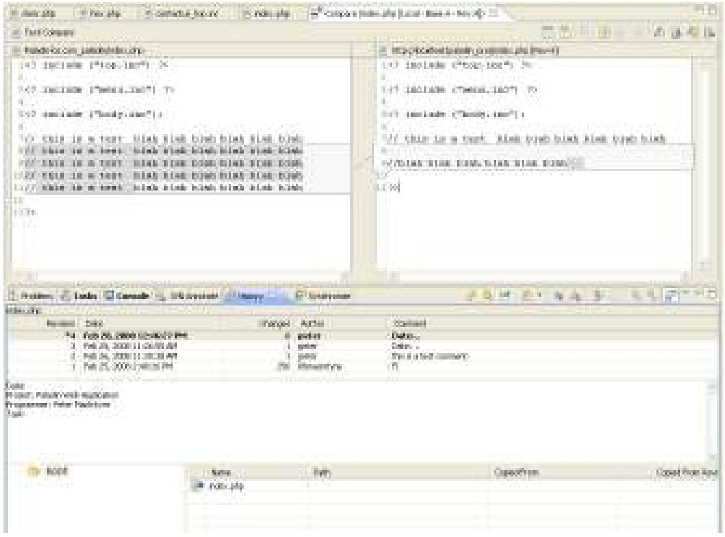


Figure 7 - Results of the Compare with Repository Request.

Synchronize View - This view is used to help maintain the up-to-date status of the files that are in your working copy. You can use it to verify that your working copy has the most recent changes that are committed to the repository. If one programmer has been working on a file and now its your turn to do some work, it is this process that you should run before you start doing your own changes. To start this process simply right-mouse on the file in question and select Synchronize with Repository from the Team menu.

SVN Annotate View – This view works hand-in-hand with the History view. Here you can select an item of a files lineage from the History view and with the right-menu option of Show Annotation you will be shown a listing of all the revisions that the file in focus has undergone. You can then jump between “versions” of the file and view that code as it looked during older revisions.

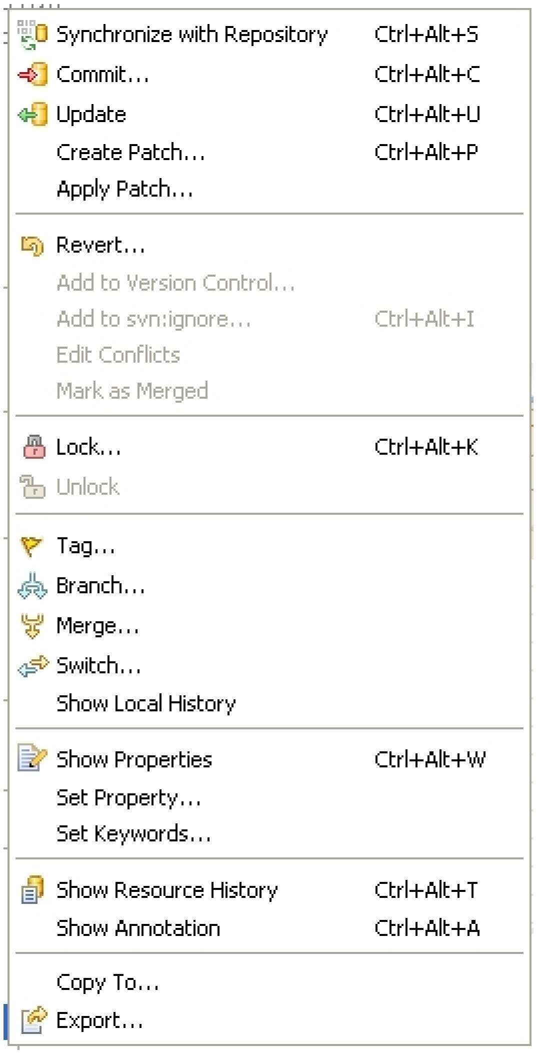


Figure 8 - Team menu with SVN Command Options

You have probably already found all the commands that are typical with the SVN software like Commit, Update, Lock, and so on. These are all accessible from the Team menu after a right-click on any file in either the code editor or the PHP Project Explorer view. The pop-up menu with all its options is shown in figure 8.

Scenario 2

Many developers are now working on your third version of a successful web application. There are 2 teams of developers, one team of 8 is in London, England and the other team of 12 is in Toronto, Canada. The two teams are 5 hours apart and have to work on the same code base that is housed on a development server at your head office in Phoenix, Arizona, USA that is another 2 time zones west of Toronto.

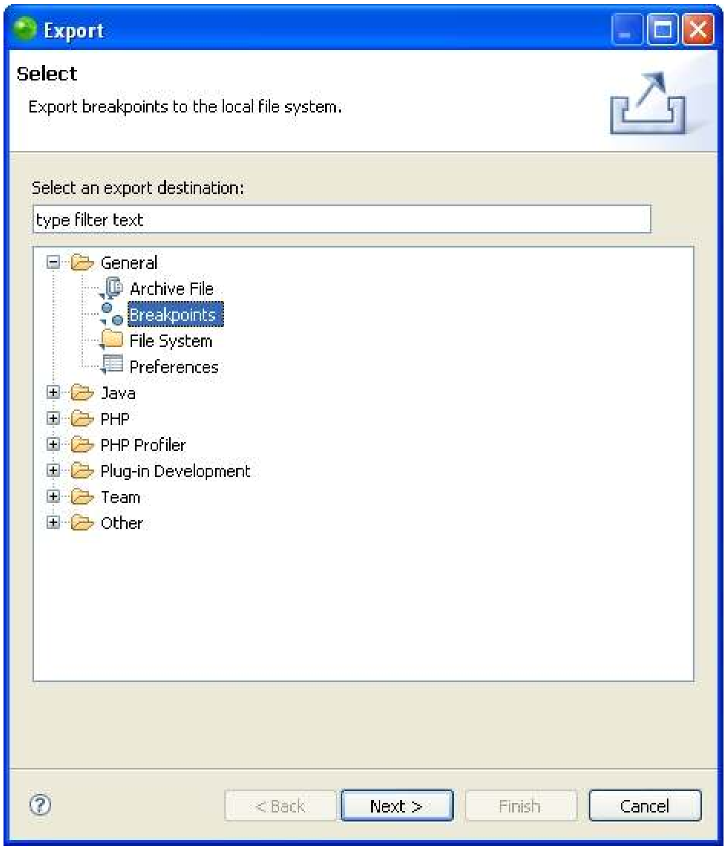
Solution Discussion

Apart from all that you have learned about SVN in the previous scenario discussion, there are still more team development features available in Zend Studio for Eclipse. Once you have established a connection to the Source Code Management server (remember both CVS and SVN are supported) from within Zend Studio for Eclipse your developers can begin the process of version control and checking out and committing their work to the development server. This ability alone can add so much stability and control to the development effort.

There are a lot of other team development options available to you within Zend Studio for Eclipse. Let's take a look at some of them now to see how they can be employed as part of the local network development effort.

In the establishment of the Zend Studio for Eclipse development environment on each developer's machine you can use the export/import wizard to great advantage. As one example, you can set up an entire development environment and have all the system wide preferences exported so that all the developers can have the same development environment (at least on the outset). You can even add that file to the SVN repository of the project so that any changes that are made over time can be committed and used by the whole team.

As another example if you are in the process of debugging a segment of code you can export your breakpoints and send them to the development team, then have the same debug processes run through by the entire team just by having them import the contents of this well-formed XML file into their individual IDEs. Figure 9 shows the first window of the export process.



Select

Export breakpoints to the local file system.

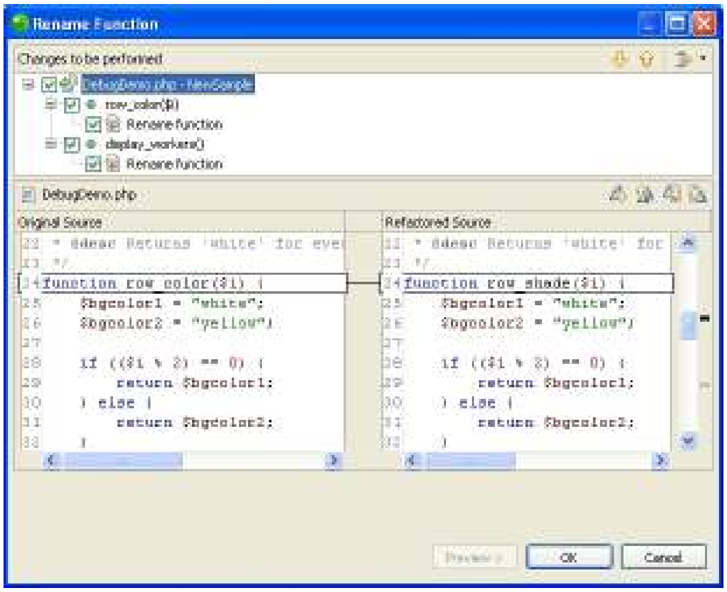
Select an export destination:

type filter text

Figure 9 - Exporting Breakpoint for Team Consumption

Refactoring

Refactoring is another great way to synchronize code throughout a development process. Refactoring is a great tool in Zend Studio for Eclipse and is really a glorified search and replace tool that can be used within a single file or throughout an entire project. For example, if one version of code has a function called row_color that is globally called row_shade, you can use the refactor tool to accomplish this. Figure 10 shows the offered results of the refactoring process.



Original Source:

```
 * @desc Returns 'white' for eve
 */
function row_color($i) {
    $bgcolor1 = "white";
    $bgcolor2 = "yellow";

    if (($i % 2) == 0) {
        return $bgcolor1;
    } else {
        return $bgcolor2;
    }
```

Refactored Source:

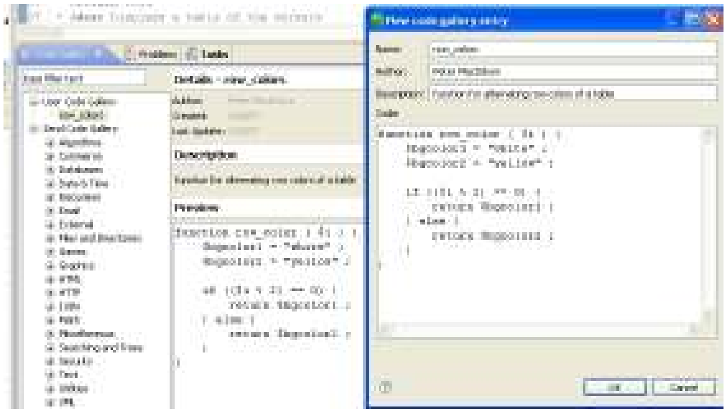
```
 * @desc Returns 'white' for
 */
function row_shade($i) {
    $bgcolor1 = "white";
    $bgcolor2 = "yellow";

    if (($i % 2) == 0) {
        return $bgcolor1;
    } else {
        return $bgcolor2;
    }
```

Figure 10 - Refactoring Process

Code Gallery

The last local development tool that we can look at is called the Code Gallery. This tool is not directly involved in team development processes, but it can certainly help with the creation of a repository of code libraries that your development team can use. Once a segment of code (a function or a class for example) has been written and tested it can be included into the local code gallery for team use and access. As well the code gallery can be expanded to a server level location for access from anyone who has the right credentials. The advantage here is that good tested code can be used across projects as well as across developers in the same project. Another great feature of the code gallery idea is that Zend has a code gallery of its own that you can access at any time and you can also submit your own code for possible inclusion into the global gallery. Figure 11 shows the code gallery creation screen and its controlling view in the background.



Description: Function for alternating row colors of a table

```
function row_color ( $i ) {
    $bgcolor1 = "white" ;
    $bgcolor2 = "yellow" ;

    if (($i % 2) == 0) {
        return $bgcolor1 ;
    } else {
        return $bgcolor2 ;
    }
}
```

Name: row_colors

Description: Function for alternating row colors of a table

Code:

```
function row_color ( $i ) {
    $bgcolor1 = "white" ;
    $bgcolor2 = "yellow" ;

    if (($i % 2) == 0) {
        return $bgcolor1 ;
    } else {
        return $bgcolor2 ;
    }
}
```

Figure 11 - Code Gallery Entry Creation & View

Zend Platform Integration

Zend Studio for Eclipse also has some valuable integrations with other Zend products. The one to look at in the context of team development is Zend Platform. This is a Zend product that can be installed on a cluster of servers for either development or production purposes. This product sits on top of the web server (Apache) and monitors all the server's activity that is generated. With this monitoring tool in place you can have it send emails to anyone when an error occurs. However there is a summary of all the server activities kept within Zend Platform that the developers can navigate to and review at their leisure. Zend Studio for Eclipse has also been integrated to allow for an interface connection to any Platform server. Figure 12 shows this integration tool opened up displaying the web interface to Platform and the list of the most recent events that occurred on that server.

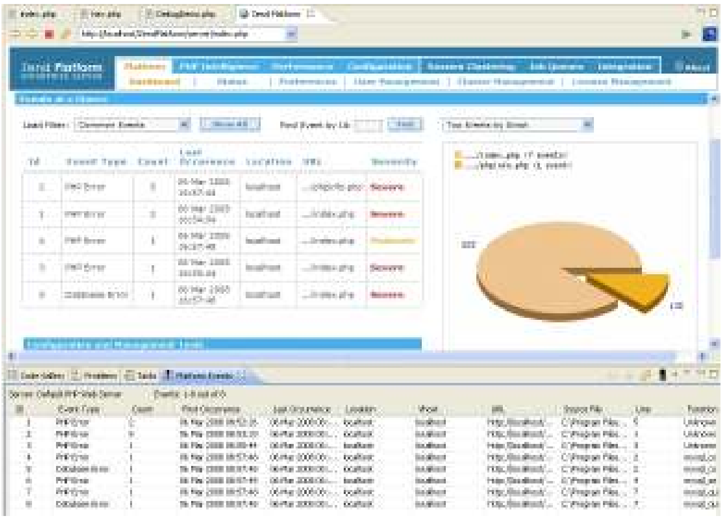


Figure 12 - Zend Platform Integration Display in Zend Studio for Eclipse

The usefulness here in the context of team development is that anyone on the support or development teams can access this interface and proceed to resolve any of the issues within the code base. As well if you right-click on one of the line items within the Events view you will have the options to look at the code in question, run a profile against the code, or to begin a debug process.

Task Lists

The last Zend Studio for Eclipse view that should be mentioned in the context of team development is that of the task list manager. Although this is not necessarily directly employable by a team it can be used very effectively for individual programmers within a team to manage their own list of things to do. The tasks list view is part of the general PHP development perspective and should be noticed on almost any basic perspective. It can be employed to keep track of any reminder items that the programmer wants to track. The tasks can be tied into a code file or just kept as a general item to be done. If it is tied into a code file then all the developer has to do to bring it into focus is to double-click on the task and the code editor will automatically open the connected file.

FTP / SFTP Connections

Finally there is one other major tool that can be used in the development of systems by remotely situated programmers. The feature of coding through an FTP connection can certainly be used to answer the key issue that was posed at the beginning of this second scenario. You can start an FTP type of project from within Zend Studio for Eclipse and have the IDE connect directly and continuously to that server while your programmers are coding against the project files. This is another method that can be used to keep your projects in one location no matter what the time zone and it also keeps them in one presumably safe and secure location.

When all the above aspects of Zend Studio for Eclipse are used in concert, they will certainly help the coordination of the development process to run smoothly. It is assumed however, that your developers will all be using these tools and helps, and will be consistent in their

application. Therefore it is certainly important to have your entire development team all in agreement with the use of not only the Zend Studio for Eclipse IDE, but also the consistent use of the tools and features that are available.

Summary

Naturally the full implementation of SCM software will be the biggest piece of any multi-developer team and it will be up to each development shop to determine which tool to use and how it should be implemented. There is a lot of research available on the web about both of the SCM tools that Zend Studio for Eclipse is compatible with, so be sure to select the one that best suits your shop.

Keep in mind that the larger the project that your development team deals with the more integration with tools is desired. With Zend Studio for Eclipse having so many of these team development features all in one IDE solution, it makes sense to take a good hard look at implementing this product throughout your enterprise.

Corporate Headquarters: Zend Technologies, Inc. 19200 Stevens Creek Blvd. Cupertino, CA 95014, USA • **Tel** 1-888-PHP-ZEND, 1-888-747-9363 • **Fax** 1-408-253-8801
Central Europe: (Germany, Austria, Switzerland) Zend Technologies GmbH Bayerstrasse 83, 80335 Munich, Germany • **Tel** +49-89-516199-0 • **Fax** +49-89-516199-20
International: Zend Technologies Ltd. 12 Abba Hillel Street, Ramat Gan, Israel 52506 • **Tel** 972-3-753-9500 • **Fax** 972-3-753-9501
France: Zend Technologies SARL, 5 Rue de Rome, ZAC de Nanteuil, 93110 Rosny-sous-Bois, France • **Tel** +33 1 4855 0200 • **Fax** +33 1 4812 3132
UK: Zend Technologies, 9 Burroughs Gardens, Hendon, London NW4 4AU, United Kingdom • **Tel** +44 20 8359 1215 • **Fax** +44 20 8203 6456
Italy: Zend Technologies, Largo Richini 6, 20122 Milano, Italy • **Tel** +39 02 5821 5832 • **Fax** +39 02 5821 5400