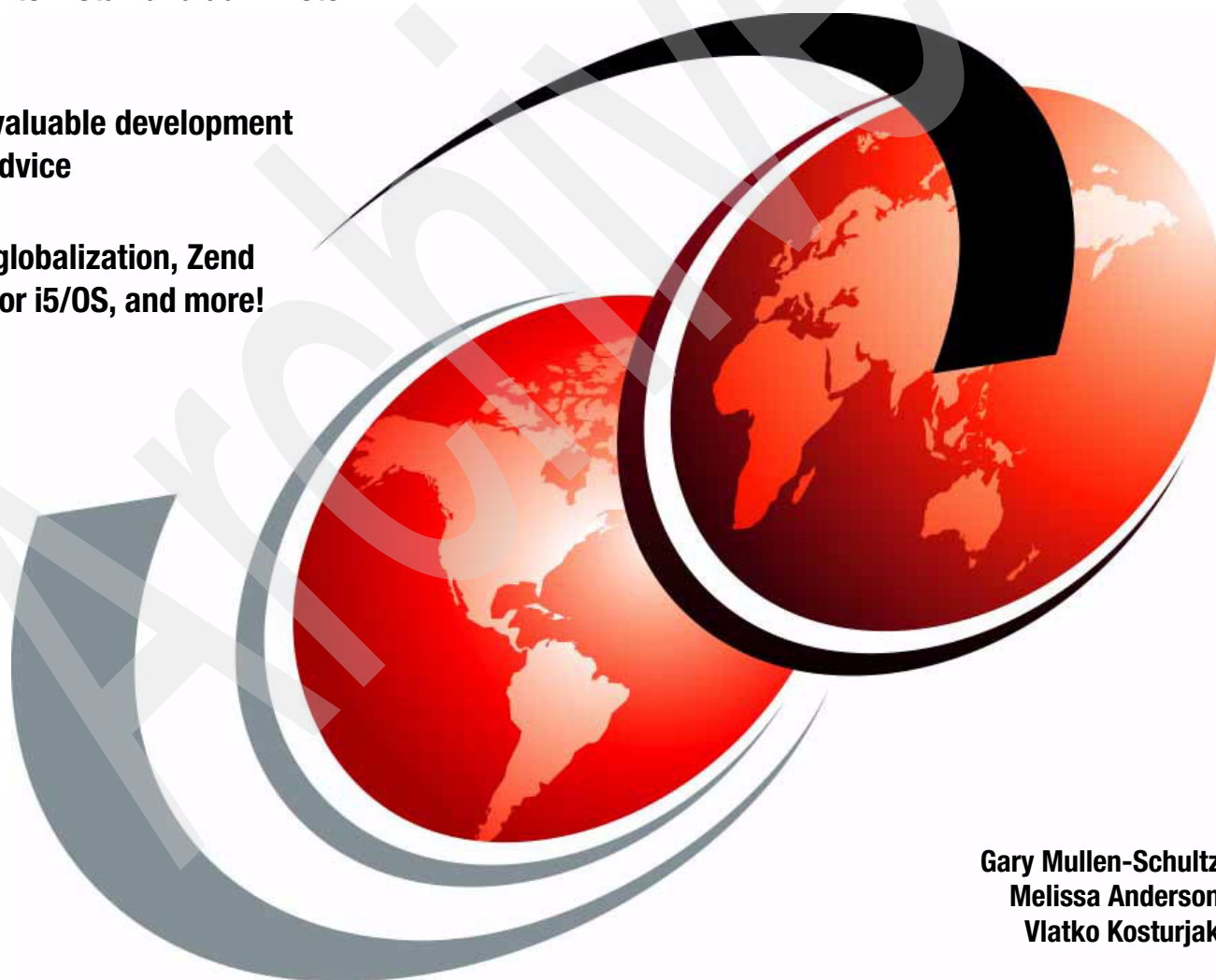


PHP: Zend for i5/OS

Learn how to install and administer

Discover valuable development
tips and advice

Security, globalization, Zend
Platform for i5/OS, and more!



Gary Mullen-Schultz
Melissa Anderson
Vlatko Kosturjak

Redbooks



International Technical Support Organization

PHP: Zend for i5/OS

January 2007

Archived

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

Archived

First Edition (January 2007)

This edition applies to Version 1.0, Release 5.0, Modification 0.0 of Zend Core for i5/OS, Version 2.0, Release 1.0, Modification 2.0 of Zend Platform for i5/OS, and Version 5.0, Release 2.0, Modification 0.0 of Zend Studio for i5/OS.

© Copyright International Business Machines Corporation 2007. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

| | |
|---|------|
| Notices | vii |
| Trademarks | viii |
| Preface | ix |
| The team that wrote this book | ix |
| Become a published author | x |
| Comments welcome | x |
| Chapter 1. Welcome to PHP on i5/OS! | 1 |
| 1.1 Welcome! | 2 |
| 1.1.1 IBM and Zend Core | 2 |
| 1.1.2 Zend Core for IBM | 2 |
| 1.2 Previous support of PHP on i5/OS | 3 |
| 1.3 Current support of PHP on i5/OS | 3 |
| 1.3.1 Zend Core for i5/OS | 3 |
| 1.3.2 Zend Studio for i5/OS | 4 |
| 1.3.3 Zend Platform for i5/OS | 4 |
| 1.4 How i5 implementation differs from Zend Core | 5 |
| 1.5 Benefit to developers and customers | 5 |
| Chapter 2. Overview of PHP | 7 |
| 2.1 History of PHP | 8 |
| 2.2 Why PHP? | 8 |
| 2.3 Who is using PHP? | 9 |
| 2.4 Quick comparison to Java | 9 |
| Chapter 3. Administration | 11 |
| 3.1 Packaging | 12 |
| 3.2 Installing Zend Core for i5/OS | 12 |
| 3.2.1 Before you begin | 12 |
| 3.2.2 Installing the Zend Core for i5/OS product | 15 |
| 3.2.3 Starting the Zend Core for i5/OS environment | 18 |
| 3.2.4 Verifying the installation | 19 |
| 3.2.5 Common installation errors | 21 |
| 3.2.6 Reinstalling Zend Core for i5/OS | 21 |
| 3.2.7 Uninstalling Zend Core for i5/OS | 22 |
| 3.2.8 Product structure | 22 |
| 3.2.9 Runtime environment | 24 |
| 3.3 Configuring Zend Core for i5/OS | 26 |
| 3.3.1 Administration tools | 26 |
| 3.4 Configuring multiple instances | 30 |
| 3.4.1 Multiple i5/OS Apache instances, one PHP server | 30 |
| 3.4.2 Multiple i5/OS Apache instances, multiple PHP servers | 32 |
| 3.4.3 One i5/OS Apache instance, multiple PHP servers | 34 |
| 3.5 Backing up Zend Core for i5/OS | 36 |
| Chapter 4. Application development | 39 |
| 4.1 PHP and Web development | 40 |
| 4.1.1 Quick Web example | 40 |

| | |
|---|----|
| 4.2 PHP as a command-line scripting language | 41 |
| 4.2.1 PHP interactive mode | 41 |
| 4.2.2 PHP one-line scripts (popularly called oneliners) | 43 |
| 4.2.3 PHP and scripting | 43 |
| 4.2.4 Calling PHP from CL | 45 |
| 4.3 Development tools | 45 |
| 4.3.1 Zend Studio for i5/OS | 46 |
| 4.3.2 Eclipse | 47 |
| 4.3.3 NuSphere PhpED | 48 |
| 4.3.4 Maguma | 49 |
| 4.4 i5 PHP API Toolkit functions | 49 |
| 4.4.1 Main functions | 50 |
| 4.5 XML support | 54 |
| 4.6 XML-RPC support | 55 |
| 4.7 SOAP support | 56 |
| Chapter 5. Database access. | 59 |
| 5.1 DB2 | 60 |
| 5.1.1 DB2 access options | 60 |
| 5.1.2 Performance considerations | 69 |
| 5.1.3 Troubleshooting | 71 |
| 5.2 MySQL | 71 |
| 5.2.1 MySQL architecture | 71 |
| 5.2.2 MySQL directory structure | 71 |
| 5.2.3 Installation and configuration | 72 |
| 5.2.4 Access options | 75 |
| 5.3 Migrating from MySQL to DB2 | 77 |
| 5.4 When to choose MySQL or DB2 | 78 |
| Chapter 6. Java Bridge support. | 79 |
| 6.1 Overview | 80 |
| 6.2 Java Bridge configuration | 80 |
| 6.2.1 Main configuration file: javamw.rc | 80 |
| 6.2.2 Java Bridge status | 80 |
| 6.3 Sample programs | 81 |
| 6.3.1 Simple test program | 81 |
| 6.3.2 IBM Toolbox for Java and PHP | 83 |
| 6.4 Troubleshooting tips | 85 |
| 6.4.1 php_error_log file | 85 |
| 6.4.2 var_dump() function | 86 |
| Chapter 7. Security | 87 |
| 7.1 Security considerations | 88 |
| 7.1.1 SSL configuration | 88 |
| 7.1.2 Access to directory structure | 88 |
| 7.1.3 Reverse proxy | 89 |
| 7.1.4 User profiles | 89 |
| 7.1.5 More information | 90 |
| Chapter 8. Performance | 91 |
| 8.1 Performance tuning | 92 |
| 8.1.1 Hardware | 92 |
| 8.1.2 i5/OS | 93 |
| 8.1.3 i5/OS PASE | 93 |

| | |
|--|------------|
| 8.1.4 DB2 for i5/OS | 93 |
| 8.1.5 Apache | 93 |
| 8.1.6 Zend Core for i5/OS | 93 |
| Chapter 9. Troubleshooting | 95 |
| 9.1 Before you start. | 97 |
| 9.2 Troubleshooting the browser. | 97 |
| 9.3 Troubleshooting the Web servers | 97 |
| 9.4 Troubleshooting Zend Core | 99 |
| 9.5 Troubleshooting application resources | 100 |
| 9.5.1 Working with object authorities | 100 |
| 9.5.2 Working with i5/OS job logs | 101 |
| 9.5.3 Working with spooled files. | 101 |
| 9.5.4 Working with the history log | 102 |
| 9.5.5 Working with message queues | 103 |
| 9.5.6 Using a debugger | 103 |
| 9.5.7 Logging Toolkit activity | 103 |
| Chapter 10. Globalization | 105 |
| 10.1 Overview | 106 |
| 10.1.1 PHP internationalization support. | 106 |
| 10.1.2 Layers on i5/OS | 106 |
| 10.2 Globalization configuration (single-byte) | 109 |
| 10.2.1 i5/OS | 110 |
| 10.2.2 i5/OS PASE | 110 |
| 10.2.3 PHP | 110 |
| 10.2.4 Apache | 110 |
| 10.2.5 DB2 for i5/OS | 111 |
| Chapter 11. Advanced development topics | 113 |
| 11.1 Implementation of templates | 114 |
| 11.2 Model View Controller (MVC) design and frameworks | 116 |
| 11.3 Tips and tricks | 118 |
| 11.3.1 Accessing system environment variables using predefined arrays. | 118 |
| 11.3.2 Reverse proxy issues | 118 |
| Related publications | 119 |
| IBM Redbooks | 119 |
| Other publications | 119 |
| Online resources | 119 |
| How to get IBM Redbooks | 120 |
| Help from IBM | 120 |
| Abbreviations and acronyms | 121 |
| Index | 123 |

Archived

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

Redbooks (logo) ™
developerWorks®
eServer™
ibm.com®
iSeries™
i5/OS®
z/OS®
AIX®

Cloudscape™
DB2 Connect™
DB2 Universal Database™
DB2®
DRDA®
Informix®
IBM®
POWER™

Rational®
Redbooks™
System i™
System i5™
System z™
WebSphere®

The following terms are trademarks of other companies:

Java, JavaScript, JDBC, J2EE, Sun, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Internet Explorer, Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbook will help you install, configure, and become productive with PHP on System i™ using Zend Core for i5/OS® and Zend Platform for i5/OS. If you are evaluating PHP, this book will also help by providing background information and comparisons with other tools.

Some of the topics addressed include:

- ▶ Installation, configuration and administration
- ▶ Application development
- ▶ Strategies to access DB2® for i5/OS and MySQL data from your PHP applications
- ▶ Security, performance, troubleshooting, and more

Emphasis has been placed on highlighting i5/OS-specific functions and features in this book rather than those generic to PHP on all platforms.

The team that wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Rochester Center.

Gary L. Mullen-Schultz is a Consulting IT Specialist at the ITSO, Rochester Center. He leads the team responsible for producing Blue Gene/L documentation, and focuses on i5/OS application development topics such as PHP, Java™, and WebSphere®. He is a Sun™ Certified Java Programmer, Developer, and Architect, and has three issued patents.

Melissa Anderson, a former member of the IBM Rochester System i Technology Center (ITC) e-business team, has taught and supported Web technologies on System i. Prior to joining ITC, she was a Web site developer and a technical writer for various System i Java and Web products. She is a medal-winning speaker at the COMMON education conferences. She is currently completing a degree in Computer Science at Winona State University.

Vlatko Kosturjak is an IT Security Specialist in IBM Croatia. In his practice, Vlatko specializes in ethical hacking, IT auditing, and implementing ISMS based on international security standards. Vlatko has given numerous talks about PHP, mod_perl, and critical Web development with an emphasis on security. He is an LPI-certified Linux® specialist with experience in deploying a wide range of secure Linux solutions. Vlatko has extensive experience with Linux on almost every platform (from PDAs to System z™), including developing, porting, and contributing to open source software.

Thanks to the following people for their contributions to this project:

Jim Bainbridge
Tony Cairns
Craig Johnson
Kent Milligan
David Peraza
Michael Sandberg
Steve Will
IBM Rochester

Whei-Jen Chen
ITSO San Jose

A special thanks to Alan for his thorough and thoughtful comments and contributions to the book:

Alan Seiden
Strategic Business Systems, Inc.

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

ibm.com/redbooks

- Send your comments in an e-mail to:

redbooks@us.ibm.com

- Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400



Welcome to PHP on i5/OS!

This chapter provides an introduction to the new support of PHP on i5/OS.

1.1 Welcome!

Zend and IBM have partnered to deliver Zend for i5/OS, a complete, all-in-one PHP development and production environment that is fully integrated with the innovative i5/OS platform. Zend for i5/OS includes Zend Core, Zend Studio, and Zend Platform to provide System i businesses with a complete solution for the development and management of modern Web applications. For System i businesses, Zend for i5/OS is one of the most productive and cost-effective routes to rapid Web enablement and the integration of Web Services supporting SOA.

1.1.1 IBM and Zend Core

IBM and Zend announced the availability of Zend Core for IBM, the industry's first integrated solution designed specifically to help developers deploy database applications and services based on the popular PHP Web language. IBM and Zend also announced that they are jointly working on furthering PHP technology to include improved high-level database integration frameworks and enhanced PHP Web services standards. At the same time, Zend announced Zend Network, a comprehensive service and support system delivering enterprise-quality updates and support to PHP developers. Zend Core for IBM integrates IBM DB2 Universal Database™ and IBM Cloudscape™, based on the Apache Software Foundation Derby project, with Zend's open source PHP environment. The combination of Zend Core for IBM and the Zend Network provides a seamless, out of the box, enterprise-grade PHP environment, easing PHP application development and deployment.

Zend Core for IBM is based on PHP 5 technology and includes tight integration with Cloudscape database server and DB2, and native support for XML and Web Services, which promotes the increased adoption of service-oriented architectures (SOA). Zend Core enables PHP application use of all DB2 server environments (including System i and System z via DB2 Connect™). It helps automate and reduce the deployment costs through a seamless, automated installation and configuration process, while providing high performance and reliable integration. In addition, it offers an upgrade path from the easy-to-use, lightweight Cloudscape database to the easy-to-use, robust DB2 Universal Database platform by providing a consistent API to developers.

The plans to develop Zend Core for IBM were announced in February of 2006. Under the terms of the agreement, IBM worked with Zend to create the product, which is the industry's first seamless, user-ready PHP database application development and production environment. As part of this effort, any bug and security fixes made to Zend Core for IBM will also be shared with the PHP community, further enhancing the overall PHP code base.

1.1.2 Zend Core for IBM

Zend Core is an enhanced version of the open source PHP engine. It delivers all of the necessary drivers and third-party libraries to work with databases, such as DB2 and MySQL. Zend Core makes software installation easier and faster with its install utility. It also provides native support for XML and Web Services, and delivers a rapid development and deployment foundation for database-driven applications.

Product highlights include:

- ▶ PHP 5
- ▶ Easy to install
- ▶ Graphical Web-based Administration Console for Cloudscape servers and PHP environment

- ▶ Bundled IBM Cloudscape server and DB2 drivers for enterprise database Web application deployments
- ▶ Easy to access documentation
- ▶ Support options available from Zend

IBM and Zend are collaborating on development and support for the PHP environment. This collaboration reinforces the IBM commitment to the open source community. IBM has submitted newly introduced and optimized DB2 and Cloudscape extensions for PHP to the PHP community and integrated them into Zend Core for IBM.

This expands the IBM investment in open source by helping developers more effectively create and deploy applications. The availability of Zend Core for IBM significantly enhances developer support for the IBM Cloudscape open-source database, because PHP is one of the most popular Web programming languages in the world. In addition, the Zend Core PHP maintenance process ensures better stability and reliability of PHP as a result of the intensive testing and bug fixes that are posted back to the PHP community of users across all facets of the market.

1.2 Previous support of PHP on i5/OS

Before the release of Zend for i5/OS, some customers successfully ran PHP applications on i5/OS by porting the open source PHP engine. In fact, a Redpaper published in 2003 describes how to accomplish this: *Bringing PHP to Your IBM eSeries iSeries Server*, REDP-3639.

In addition, a Web site has been available for several years (<http://www.i5php.net/>) that contains compiled binaries of the open-source PHP code ready for specified releases of i5/OS. Thus, PHP support has in fact been available and in production for some time!

1.3 Current support of PHP on i5/OS

One of the primary benefits of having Zend provide a full suite is that product-level support is available. In the past, customers had to fix bugs on their own, with no real installation or update scripts to help. Informal forums were the only way to get help from others.

Now, support is available directly from Zend for problems encountered with their products. This is a much more positive situation for most customers who want to run their PHP applications in a more mission-critical fashion.

1.3.1 Zend Core for i5/OS

Zend Core for i5/OS is an enhanced version of the open source PHP with resources and capabilities specific to i5/OS. It is provided free of charge by Zend to System i customers.

Zend for i5/OS is a fully tested and enhanced version of the open source PHP engine. It provides the PHP runtime for i5/OS and is packaged to make software installation easier and faster with the provided setup application. Zend for i5/OS includes Zend Core for i5/OS, Zend Studio for i5/OS, and Zend Platform for i5/OS.

Zend Core for i5/OS is enhanced to take advantage of i5/OS-specific resources and capabilities, and provides a seamless PHP development and production environment product that is fully supported by Zend. In addition, it integrates with DB2 for i5/OS. The product

includes native support for XML and Web Services in support of the increasing adoption of SOA applications. It delivers a rapid development and deployment foundation for database-driven applications and offers an upgrade path from the Cloudscape database to the DB2 by providing a consistent API between the two.

Find more information about Zend for i5/OS at:

http://www.zend.com/products/zend_core/zend_for_i5_os

Besides the software downloads, that site contains additional information such as data sheets and white papers about Zend Core for IBM. Developers should take a look at the Developer Zone, which features tutorials, documentation, discussion forums, and much more. Developer Zone is located at:

<http://www.zend.com/developers.php>

Zend Core for i5/OS supports PHP 5.1 on i5/OS V5R3 (in beta at the time this document was written) and V5R4, and was released on July 31, 2006.

Some of the benefits of Zend Core for i5/OS include:

- ▶ IBM DB2 native support through `ibm_db2` and ODBC PHP extensions
- ▶ Zend Network automatic online updating services
- ▶ Web administration GUI
- ▶ Zend Core for i5/OS setup tool
- ▶ Access to existing native i5/OS resources—CL and RPG programs, data areas, and more—via the PHP Toolkit
- ▶ Support for PHP 5.1.6 and Apache on i5/OS
- ▶ Easy installation
- ▶ Certified PHP, fully supported by Zend
- ▶ Numerous popular PHP extensions, allowing support for thousands of PHP applications and scripts

1.3.2 Zend Studio for i5/OS

Zend Studio for i5/OS is an integrated development environment (IDE) that can greatly speed PHP development.

Benefits include:

- ▶ Instant online debugging and error fixing against Zend Core for i5/OS
- ▶ Supports PHP 5
- ▶ Troubleshooting with PHP Intelligence
- ▶ Syntax completion
- ▶ SQL tool for exploring your DB2 for i5/OS and MySQL databases
- ▶ Superb support for SOAP, with automatic creation of WSDL files
- ▶ Works exclusively with Zend Core for i5/OS

1.3.3 Zend Platform for i5/OS

Zend Platform for i5/OS is the all-in-one production environment that ensures that PHP applications are always available, fast, reliable and scalable on the i5/OS platform. Zend Technologies fully supports all products.

Important: Zend Platform for i5/OS is *not* a no-charge product like Zend Core and Studio for i5/OS.

Zend Platform for i5/OS brings additional value, including:

- ▶ Comprehensive PHP application insight
- ▶ Online debugging and immediate error fixing with Zend Studio for i5/OS
- ▶ Runtime code optimization
- ▶ PHP/Java integration bridge
- ▶ Dynamic content caching

1.4 How i5 implementation differs from Zend Core

A major advantage of Zend Core for i5/OS is that it introduces an extremely stable and robust platform for deploying and hosting PHP applications. By taking advantage of proven System i hardware and Zend Support for PHP, ISVs and customers can focus on developing needed applications rather than fixing quirks and bugs in the hardware or operating system.

There is no significant difference between writing PHP applications for Zend Core for i5/OS and the other platforms. The main difference is in calling i5/OS-specific operating system commands and in the use of advanced database features.

Be aware that, at the time this was written, clustering support was not implemented in Zend Core for i5/OS. Refer to Chapter 4, “Application development” on page 39 and Chapter 3, “Administration” on page 11 for more information about Zend Core for i5/OS specifics.

1.5 Benefit to developers and customers

As more companies come to rely on the Internet to support their core business processes, there is an increasing demand for Web sites that are better able to serve their customers and more efficiently take value out of existing data, services, and IT infrastructure.

Increasingly, businesses are choosing PHP and IBM data servers for the low cost, increased reliability, high performance, and speed of development that this platform provides.

PHP can be used as an alternative to CL scripting or RPG programming. This means that developers who are unfamiliar with CL or RPG, but familiar with PHP, can write scripts for i5/OS.

One huge benefit for Zend for i5/OS users is the fact that Zend Studio for i5/OS is available for no charge for users of Zend Core for i5/OS.

Archived



Overview of PHP

This chapter describes PHP and discusses its history and popularity. It also covers some advantages and disadvantages in comparison to Java and general recommendations for solving some of the disadvantages of PHP.

2.1 History of PHP

In the year 2005, the PHP community celebrated the 10th anniversary of PHP. PHP was written by Rasmus Lerdorf from Denmark and named Personal Home Page/Form Interpreter (PHP/FI), because it was created to automate the handling of certain parts of a Web site, such as forms.

Before PHP, developers used mainly Perl, another open-source scripting language. While Perl, which stands for Practical Extraction and Reporting Language, is very useful, it proved cumbersome for some to write and maintain code. Over the years, many discussions have taken place between PHP and Perl supporters, trying to find out which is the *better* language. We do not want to enter this discussion because we think there is no better language as such, but it is a fact that some tools and languages are better suited for the task at hand. A famous comparison cites that you can use a hammer to put a screw into a wall, but a screwdriver would be the better choice for that matter.

After a rewrite of the open source PHP/FI tools in the year 1997 by Lerdorf, a programmer working for an Internet Service Provider in Israel (Zeev Suraski) and one of his students (Andi Gutmans) decided to completely rewrite the scripting language and the parser of PHP/FI. Together with Rasmus Lerdorf they released what was called PHP 3 in June 1998.

In 1998, there were about 50,000 Internet domains using PHP but before long this number reached the barrier of one million domains. Zeev Suraski and Andi Gutmans were apparently not yet satisfied, however, and three and a half years later, in May 2002, PHP 4 was released. At the same time they created a new engine, which was called Zend. This is now the name of the company that stands behind PHP.

But it would be unfair to name only Rasmus Lerdorf, Zeev Suraski, and Andi Gutmans as the creators of PHP. In fact, as with most open source projects, PHP unifies the efforts of many people. They are probably too numerous to name here, but as users of PHP we are thankful for their efforts.

PHP 5, announced in July 2004, represented another rewrite of the Zend engine. The number of Web sites using PHP has reached 22 million and is still growing. Zend estimates that there are about 2.5 million PHP developers.

2.2 Why PHP?

The short answer to this question might be: because it is in use on millions of Web sites by hundreds of thousands or even millions of people, ranging from amateurs to Web professionals who are leveraging it for their family Web site as well as for creating full-blown commercial applications. There must be a reason for this success.

PHP was designed for Web development. It makes it possible to write dynamically generated Web pages quickly and easily, to read and write files, access and process form data, send e-mails, read and write cookies and maintain session variables, and much more. Not to forget one of its major strengths: integrated access to database records, which includes reading and writing records from all major databases. Because doing all of this with PHP is relatively easy, especially compared to some other development languages, it can safely be called a rapid programming language for Web developers.

Furthermore, the PHP community has created an impressive amount of additional functionality through freely available code samples (see <http://www.hotscripts.com>, with

more than 12,000 PHP scripts; or <http://www.phpclasses.org>, which provides many PHP classes), PHP extensions (see <http://pear.php.net>), and assorted documentation.

In addition, PHP does not use a lot of system resources while offering high speed of execution, stability, and extensibility, which are just some of the reasons why PHP is so successful and still growing in popularity. Bringing this success to i5/OS makes sense for both traditional PHP developers and long-standing i5/OS programmers.

2.3 Who is using PHP?

As mentioned previously, many Web sites use PHP. Major companies that have adopted PHP include Yahoo! (see <http://public.yahoo.com/~radwin/talks/yahoo-phpcon2002.htm> for “Making the Case for PHP at Yahoo!”), Lufthansa for its e-ticketing system, Electronic Arts for Sim City Online, Boeing for a payload measure system, and Orange for its WAP portal.

The 5/2004 issue of the German *PHP Magazin* (<http://www.php-magazin.de>) has a report about how a team migrated a Java-based Content Management System (CMS) to a PHP-based one. If more proof is needed, this article shows that companies have even successfully implemented large-scale transactional systems based on PHP instead of Java.

2.4 Quick comparison to Java

PHP was invented and designed for creating Web applications. Java was designed as a general programming language. The PHP syntax is similar to C, Java, and Perl.

PHP is easier to get started with, especially for small to medium installations. Java works well for a large team that needs built-in management tools to coordinate large-scale development and deployment, or when the application is largely not Web-based. Some shops use both: Java on the back end and PHP on the front end. See the IBM developerWorks® article “Pair J2EE with PHP to implement a common Web application infrastructure” by Dan Krook for an excellent discussion of this:

http://www.ibm.com/developerworks/websphere/techjournal/0505_krook/0505_krook.html

Although PHP and Java share some syntax and design, each has advantages and disadvantages. PHP advantages over Java include:

- Easy to use

PHP is a scripting language that can be embedded directly into HTML. This makes it easy to mix presentation with business operation and data access. However, unlike JavaScript™ where the script is executed in the browser on the client side, PHP runs on the server. This means that none of your PHP is visible to the end users in the final HTML source, which makes PHP more secure. Also, deployment is much easier because you do not have to compile PHP programs or spend time learning deployment tools to create PHP. You can simply insert changes into the Web page and get quick turnaround.

- Short learning curve

Getting started is easy. PHP also provides object oriented (OO) features that enable you to design modern Web-based applications that are robust and secure. With PHP, you can create Web pages that reflect current information from your favorite database quickly. You can get information from the user viewing your Web page to customize the page specifically for that user. Other functions provide access to e-mail, flat files, or other data such as Lightweight Directory Access Protocol (LDAP) data stores. PHP also includes a spell checker, XML functionality, image-generation, and other functions.

- Active and vibrant developer community

There are a number of Web sites, blogs, discussion forums, and other resources available to PHP developers. There are thousands of readily available PHP-based applications, libraries of functions, components, and frameworks that can be used to develop applications more quickly. Much of this software is free and can be used on any project. This provides a community an invaluable asset to assist developers with ideas, examples, techniques, and other advice on PHP.

Advantages of Java over PHP include:

- Full implementation of object-oriented language features

Java has a full implementation of object-oriented (OO) language features; PHP was not originally designed as an object-oriented language. Only later was OO programming introduced to PHP in Version 3, with limited functionality and based largely on C++ and Java. In PHP 5 many OOP components (such as interfaces, access control, and abstract class) were added, so PHP finally became “OO ready.” However, it still lacks a number of OO features that Java provides today.

- Better code maintenance

Because PHP is very easy to start with, it is easy to write code that is difficult to maintain. Java has developed methods and concepts that more firmly encourage clean and maintainable code. If a PHP developer does not take care to follow common “best practice” programming rules (separation of business logic from GUI, database abstraction, and so forth), PHP applications can be hard to port to other platforms or databases. They can also be difficult to enhance later with new functionality.

- Enterprise manageability with J2EE™

Most full J2EE implementations (for example, IBM WebSphere Application Server products) provide extensive support for clustering, failover, and overall management of an enterprise Web system. PHP implementations are generally not as advanced or tightly integrated in this respect.

This list of advantages and disadvantages of PHP and Java was current when this book was written. Both languages are changing rapidly, and many of the issues are improving. In addition, note that some of disadvantages of both languages can be eased by using extensions or other add-on products.



Administration

In this chapter we provide an introduction to administering Zend Core for i5/OS, including installation, configuration, administration tools, managing the runtime environment, and backing up the product.

3.1 Packaging

Zend provides PHP functionality on System i through Zend Core for i5/OS.

3.2 Installing Zend Core for i5/OS

To install Zend Core on i5/OS, you restore the licensed program from an i5/OS save file that you download from the Zend Web site. Most of the installation activity concerns ensuring that your system meets all requirements for the product.

Tip: Take the time to verify the prerequisites. This can eliminate most installation problems.

As of the time this was written, Zend Core for i5/OS could be downloaded from:

http://www.zend.com/products/zend_core/zend_for_i5_os

3.2.1 Before you begin

Taking time to check a few things on your system can help you avoid common installation problems. We recommend the following activities to ensure that your system is ready for installing Zend Core for i5/OS.

Checking hardware prerequisites

At this time, there are no formal hardware requirements for running Zend Core for i5/OS. For the Zend Core product itself, not including software prerequisites, we recommend at least 200 MB of hard disk space.

The PHP environment provided by Zend Core for i5/OS itself is not highly processing-intensive or heavily constrained by system resources. Hardware resource requirements depend on the answers to the following questions:

- ▶ How many PHP applications are you planning to run? How large and complex are they?
- ▶ How many users are you planning to support? How intensive do you anticipate their usage to be (light or heavy)?
- ▶ How processing-intensive are your PHP applications? Is there a high degree or low degree of dynamic content?
- ▶ How much database or system object access do your PHP applications perform?

The higher the number of applications or files, users, processing, and resource access, the more hardware resources you need.

Checking software prerequisites

Before you install Zend Core for i5/OS, make sure all prerequisite software and fixes have been installed.

Checking licensed programs

Perform the following steps to check that all prerequisite licensed programs are installed on your system:

1. Sign on to the i5/OS system and run the GO LICPGM command.

2. On the Work with Licensed Programs display, type option 10 (Display installed licensed programs).
3. Press F11 twice to display the product options.
4. Ensure that the software listed in Table 3-1 is installed.

Table 3-1 Software prerequisites

| Licensed program | Option | Description text |
|------------------|---------------|--|
| 5722SS1 | *BASE | i5/OS Version 5 Release 4 (V5R4) |
| 5722SS1 | 13 | System Openness Includes |
| 5722SS1 | 30 | Qshell |
| 5722SS1 | 33 | Portable App Solutions Environment |
| 5722SS1 | 34 | Digital Certificate Manager ¹ |
| 5722DG1 | *BASE | IBM HTTP Server for i5/OS |
| 5722JV1 | *BASE | IBM Developer Kit for Java ² |
| 5722JV1 | Option 5 or 6 | Java Development Kit 1.3 or 1.4 ² |
| 5722SC1 | *BASE | IBM Portable Utilities for i5/OS |
| 5722SC1 | Option 1 | OpenSSH, OpenSSL, zlib |

¹ Digital Certificate Manager is used for configuring secure sockets layer (SSL).

² IBM Developer Kit for Java (5722JV1) is not required to run Zend Core for i5/OS. It is required to support the IBM Web Administration for i5/OS tool that is shipped with IBM HTTP Server for i5/OS (5722DG1). The Web Administration tool is a graphical browser-based tool that you can use to administer HTTP servers. Java is also required if you plan to use the Java Bridge functionality included with Zend Platform for i5/OS (see Chapter 6, "Java Bridge support" on page 79 for more details).

5. Press F3 twice to return to the main menu.

Checking i5/OS fixes

Make sure you have the latest individual and group fixes for your system. (*Group fixes* are collections of fixes that pertain to a specific product.)

See the following resources to identify the latest fix levels:

- i5/OS fixes (including database)
http://www-912.ibm.com/s_dir/slkbasesf/recommendedfixes
- i5/OS PASE fixes
<http://www.ibm.com/servers/enable/site/porting/series/pase/misc.html>
- IBM HTTP Server for i5/OS fixes
<http://www.ibm.com/servers/eserver/series/software/http/services/service.html#PTF>
- IBM Developer Kit for Java
http://www-912.ibm.com/s_dir/sline003.NSF/GroupPTFs?OpenView&view=GroupPTFs
(Expand **R540** then click **SF99291: 540 Java**.)

Use the Display PTF (DSPPTF) command (for individual fixes) and the Work with PTF Groups (WRKPTFGRP) command (for group fixes) to check what fixes have been applied to your system.

You should order and install any missing fixes prior to installing Zend Core for i5/OS.

Checking user profile authorities

For the installation process (and other administrative activities), you need to use a user profile of *SECOFR user class (with all special authorities). Use the Work with User Profiles (WRKUSRPRF) command to check your user profile.

Removing an alpha or beta version of Zend Core for i5/OS

Tip: If you are installing Zend Core for i5/OS for the first time, skip to “Checking for existing product directories and library” on page 14.

You must remove any alpha-level or beta-level versions of Zend Core for i5/OS prior to installing the production-level version of the product. If remnants of earlier versions are detected on the system, the installation will fail.

The steps you take to remove a pre-release of the product depends on which version you have. Follow the steps in 3.2.7, “Uninstalling Zend Core for i5/OS” on page 22. If the Zend Core product appears in the licensed program list, continue with the uninstallation steps to remove the product.

If Zend Core does not appear in the licensed program list (meaning that you have an alpha or early beta version), perform the following steps:

1. Sign on a 5250 session to the i5/OS system.
2. Run the following command to call the uninstallation program:
`DLTLICPGM 1ZCORE5`
3. The uninstaller does not remove the ZENDCORE library, so run the following command to delete it:

`DLTLIB LIB(ZENDCORE)`

Checking for existing product directories and library

Ensure partial installations of the product does not exist. This can cause the installation to fail.

Perform the following steps to check for and clean up any partial installations:

1. Make sure the ZENDCORE library does not exist:
 - a. On the i5/OS command line, enter the following command:
`WRKLIB LIB(ZENDCORE)`
 - b. If the ZENDCORE library is listed, type 4 on the option line in front of it and press Enter.
 - c. Press Enter to confirm the deletion.
 - d. Press F3 to return to the main command menu.
2. Start Qshell. Enter the following command on the i5/OS command line:
`STRQSH`

Tip: We recommend using Qshell to check for existing resources because it enables you to recursively delete a directory and its contents.

3. Make sure the /www/zendcore directory does not exist:

Attention: Before deleting this directory make sure you back up any important files (such as php.ini and httpd.conf).

- a. Change to the /www directory:
`cd /www`
 - b. Display the contents of the directory:
`ls`
 - c. If a directory named zendcore appears, delete it with the following command:
`rm -r zendcore`
4. Make sure the /usr/local/Zend directory does not exist:
 - a. Change to the /usr/local directory:
`cd /usr/local`
 - b. Display the contents of the directory:
`ls`
 - c. If a directory name Zend appears, delete it with the following command:
`rm -r Zend`
 5. Press F3 to exit Qshell.

Checking TCP/IP configuration

Web technologies rely heavily on TCP/IP. Before you install the Zend Core product, ensure that TCP/IP is appropriately configured.

In particular, check the following configuration settings:

- ▶ Ensure that a host name is defined for the system. Run the Configure TCP/IP (CFGTCP) command and select option 12 (Change TCP/IP domain information) to display this setting. Make sure that a value is listed in the Host name field.
- ▶ Make sure that the loopback entry (which represents “localhost” or 127.0.0.1) is configured in the TCP/IP host table. Run the Configure TCP/IP (CFGTCP) command and select option 10 (Work with TCP/IP host table entries) to display the host table. Ensure that an entry for the IP address 127.0.0.1 exists and is mapped to host names LOOPBACK and LOCALHOST.

Also on this display, check that the IP address for the i5 system is mapped to its host name (such as SYSTEMA) and fully qualified host name (such as SYSTEMA.MYCOMPANY.COM). You should be able to successfully ping both the loopback address and the fully qualified host name of your system from i5/OS, and the fully qualified host name from any browser that will be used to access PHP scripts.

3.2.2 Installing the Zend Core for i5/OS product

After you have finished the prerequisite setup and checking, you are ready to install the Zend Core product:

1. Download the Zend Core for i5/OS product from Zend.com:
https://www.zend.com/core/oem_registration.php?access_code=IBMi50SZend

You must first register with Zend.com. During this process, you create a Zend Network user ID and password, which you can use to log in to the Zend download pages and to update the product after it has been installed. You can provide this user ID and password to the product during installation or from the setup menu after installation.

Download the Zip file to your workstation system.

2. Extract the Zip file to a temporary directory on your workstation.

The following files are extracted:

| | |
|--------------------------------------|--|
| zcoresavf.savf | A save file that contains the Zend Core for i5/OS product. |
| Release_notes | Contain information about submitting feedback, product features and extensions, system prerequisites, installation instructions, information about setting up a debug connection between Zend Studio and Zend Core, and links to component licenses. |
| Readme | Contains basic installation instructions and system prerequisites. |
| Zend_Core_User_Guide_i5OS.pdf | Includes installation instructions, getting-started information, and documentation about the Zend Core console. |

3. Log on to the i5/OS system with a user profile of *SECOFR user class with all special authorities.
4. Create a save file with the following command:

```
CRTSAVF FILE(QGPL/ZCORESAVF) TEXT('Zend Core product save file')
```
5. Verify that FTP is running on your i5/OS system by typing:

```
NETSTAT *CNN
```

Look for FTP or 21 in the Local Port column.
6. On your workstation, open a command prompt and transfer the Zend Core save file to the i5/OS system:
 - a. Change directory to the directory that contains the files you extracted from the ZIP file, for example:

```
cd temp
```
 - b. Run the FTP command, specifying the name of your i5/OS system, for example:

```
ftp systema
```
 - c. If requested, enter a valid user profile and password.
 - d. Enter the **bin** command to specify a binary transfer.
 - e. Transfer the save file to the i5/OS system:

```
put zcoresavf.savf
```
 - f. When the transfer has completed, enter the **quit** command.
7. Return to your 5250 session and run the Restore Licensed Program (RSTLICPGM) command to install the product:

```
RSTLICPGM LICPGM(1ZCORE5) DEV(*SAVF) SAVF(QGPL/ZCORESAVF)
```

Note: The value of the licensed program (LICPGM) parameter starts with the number 1, not the capital letter i.

Alternatively, you can type RSTLICPGM and press F4 to prompt the command parameters. Press F10 to view additional parameters. Specify values for Product, Device, Save file, and Library, and press Enter.

Restore Licensed Program (RSTLICPGM)

Type choices, press Enter.

| | | |
|----------------------------------|-------------------|-------------------------------|
| Product | <u>1ZCORE5</u> | Character value |
| Device | <u>*SAVE</u> | Name, *SAVF |
| + for more values | | |
| Optional part to be restored . . | <u>*BASE</u> | *BASE, 1, 2, 3, 4, 5, 6, 7... |
| Type of object to be restored . | <u>*ALL</u> | *ALL, *PGM, *LNG |
| Language for licensed program . | <u>*PRIMARY</u> | Character value, *PRIMARY... |
| Output | <u>*NONE</u> | *NONE, *PRINT |
| Release | <u>*FIRST</u> | Character value, *FIRST |
| Replace release | <u>*ONLY</u> | Character value, *ONLY, *NO |
| Volume identifier | <u>*MOUNTED</u> | |
| + for more values | | |
| Sequence number | <u>*SEARCH</u> | 1-16777215, *SEARCH |
| End of media option | <u>*REWIND</u> | *REWIND, *LEAVE, *UNLOAD |
| Save file | <u>ZCORESAVEZ</u> | Name |
| Library | <u>OGPL</u> | Name, *LIBL, *CURLIB |

More...

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys

Figure 3-1 Prompting the Restore Licensed Program (RSTLICPGM) command

8. After a few minutes, the installation wizard starts. During the wizard, you perform the following actions:
 - Read and accept the license agreement.
 - Provide a password that you want to use to access the Zend Core Web Administration Console. This password is arbitrary and not tied to any user profile or ID.
 - Enter your Zend Network user ID and password (which you used to download the product Zip file). Providing this information enables you to set up automatic updates for the Zend Core for i5/OS installation. You do not have to enter your user ID and password at this time; you can specify this information later.

Note: If you press F3 or make a choice that causes the installation wizard to exit before the installation process is complete, follow the steps in “Checking for existing product directories and library” on page 14 to clean up your system. Note that pressing F3 from the “Zend Network ID” prompt does not terminate the installation process—it continues without configuring this specific feature.

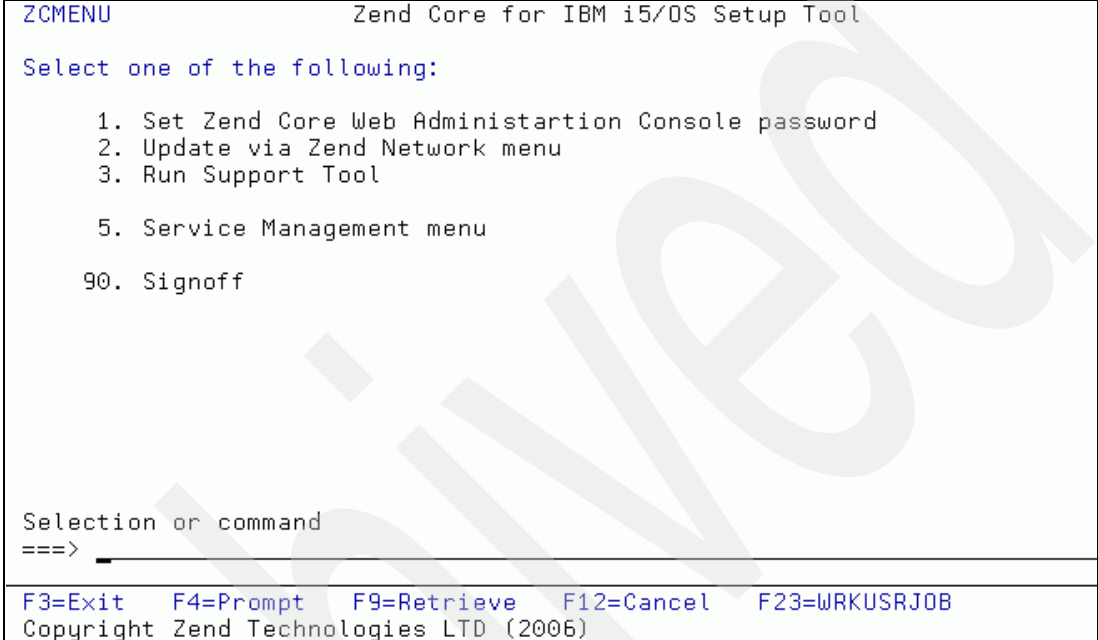
3.2.3 Starting the Zend Core for i5/OS environment

After the product is installed, start the environment. Use the Zend Core for IBM i5/OS Setup Tool.

1. On the i5/OS command line, enter the following command:

```
GO ZENDCORE/ZCMENU
```

The Zend Core setup tool appears, as shown in Figure 3-2:

A screenshot of a terminal window showing the 'Zend Core for IBM i5/OS Setup Tool' menu. The menu is titled 'ZCMENU' and lists several options: 1. Set Zend Core Web Administration Console password, 2. Update via Zend Network menu, 3. Run Support Tool, 5. Service Management menu, and 90. Signoff. Below the list is a prompt 'Selection or command' followed by '===>' and a cursor. At the bottom, there is a footer with function key shortcuts: F3=Exit, F4=Prompt, F9=Retrieve, F12=Cancel, F23=WRKUSRJOB, and a copyright notice for Zend Technologies LTD (2006).

```
ZCMENU                                Zend Core for IBM i5/OS Setup Tool

Select one of the following:

    1. Set Zend Core Web Administration Console password
    2. Update via Zend Network menu
    3. Run Support Tool

    5. Service Management menu

   90. Signoff

Selection or command
===>

F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F23=WRKUSRJOB
Copyright Zend Technologies LTD (2006)
```

Figure 3-2 Zend Core for IBM i5/OS Setup Tool

2. On the command line, type 5 to select the Service Management menu, and press Enter.

The Service Management menu appears, as shown in Figure 3-3 on page 19.

| ZCAMENU | Zend Core for IBM i5/OS Setup Tool |
|--|------------------------------------|
| Select one of the following: | |
| 1. Start Zend Core Subsystem 2. Stop Zend Core Subsystem 4. Start Apache server instances 5. Stop Apache server instances 6. ReStart Apache server instances 8. Start i5_COMD service 9. End i5_COMD service | |
| Selection or command ===> | |
| F3=Exit F4=Prompt F9=Retrieve F12=Cancel F23=WRKUSRJOB | |

Figure 3-3 The Service Management menu

3. On the command line, type 1 to select the option to start the Zend Core subsystem, and press Enter.
This action starts the jobs that run in the ZEND subsystem.
4. Starting the ZEND subsystem causes all necessary Core and Platform jobs to be started.
5. Verify that the appropriate jobs are running.
 - a. First, check that the IBM HTTP Server for i5/OS instance (ZENDCORE) is running.
On the command line, enter the following command:

```
WRKACTJOB SBS(QHTTPSVR)
```

 A list of active jobs running in the QHTTPSVR subsystem appears. Ensure that five jobs named ZENDCORE appear in the list. (You might need to press the PageDown key to scroll through the list.)
 - b. Press F3 to return to the Service Management menu.
 - c. Check that the Zend Core jobs are running. Enter the following command:

```
WRKACTJOB SBS(ZEND)
```

 A list of active jobs running in the ZEND subsystem appears. Ensure that one or more instances of the following jobs appear in the list:
 - I5_COMD
 - ZC_STR_PRN
 - ZENDCOREAP
 - d. Press F3 to return to the Service Management menu.
6. Press F3 twice to return to the main command menu.

3.2.4 Verifying the installation

After you have started the Zend Core environment, verify that the installation was successful.

To verify the installation, ensure that PHP files can be processed. Probably the easiest way to do this is to access the Zend Core for i5/OS administration console. The console is a browser-based tool that is used to administer the Zend Core product. The tool is written in PHP, so if the console is functional, the PHP environment is working.

To access the Zend Core for i5/OS administration console, perform the following steps:

1. Open a Web browser and enter the following URL:

`http://<server_name>:89/ZendCore`

where <server_name> is the host name or IP address of your i5 system.

2. A redirection page appears. In a few seconds, the Zend Core console logon page appears, as shown in Figure 3-4.

Log in to the console with the administrative password you entered in the installation wizard.



Figure 3-4 The Zend Core for i5/OS console logon page

3. The console appears, as shown in Figure 3-5 on page 21.

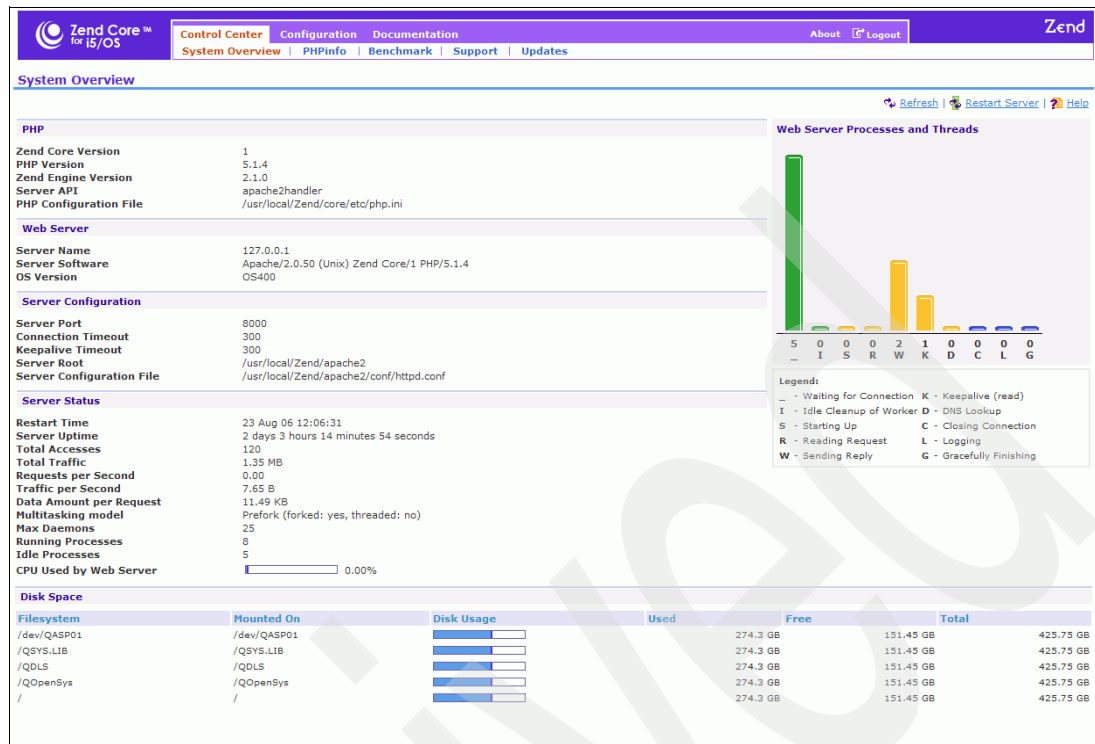


Figure 3-5 Zend Core for i5/OS console

After you have verified that the console is functioning, click **Logout** in the upper-right corner of the console and close the browser window.

3.2.5 Common installation errors

If an installation fails, you will likely see the following error message:

Objects for product 1ZCORE5 option *BASE release *FIRST not restored.

Check for messages in the job log. Installation failures are usually caused by one or more of the following conditions:

- ▶ Your user profile does not have sufficient authority.
- ▶ Structures from a previous installation are found by the installer.
- ▶ Your library list does not contain the QGPL or QTEMP libraries.
- ▶ Prerequisite software products or fixes are missing.
- ▶ /usr/local exists as a symbolic link. Delete it. Optionally re-create it as a true directory.

After you have corrected the problem, make sure you remove any product files that were created during the failed installation. See “Checking for existing product directories and library” on page 14 for more information.

3.2.6 Reinstalling Zend Core for i5/OS

If, for some reason, you need to reinstall Zend Core for i5/OS, make sure you first uninstall the existing installation as outlined in the next section.

3.2.7 Uninstalling Zend Core for i5/OS

Before you uninstall Zend Core for i5/OS, make sure you back up any files you want to save for future reference. The uninstaller deletes all product and configuration files. For more information, see 3.5, “Backing up Zend Core for i5/OS” on page 36.

1. Sign on a 5250 session to your i5/OS system, using a user profile of *SECOFR user class with all special authorities.
2. Run the Delete Licensed Program (DLTLICPGM) command to uninstall Zend Core for i5/OS:

```
DLTLICPGM LICPGM(1ZCORE5)
```

3. After the uninstallation completes, a message is displayed:

```
*PGM objects for product V5R4M0 option *BASE release V5R4M0 deleted.  
Objects for product 1ZCORE5 option *ALL release *ONLY deleted.
```

Common problems with uninstallation:

- ▶ The ZENDCORE library is set as the current library. Use the Change Current Library (CHGCURLIB) command to set another library as the current library.
- ▶ Product files are missing. The uninstaller uses certain files in the ZENDCORE library. In the unlikely case that some of these files are missing, the uninstaller might fail. To recover, reinstall the product so you have a complete installation, and then uninstall. (See “Reinstalling Zend Core for i5/OS” on page 21.)

3.2.8 Product structure

The Zend Core for i5/OS installation creates the following objects.

Library

The ZENDCORE library contains i5/OS-specific code for installing, configuring, and starting the product environment.

User profiles

The Zend Core installation creates the user profiles listed in Table 3-2.

Table 3-2 Zend Core user profiles

| User profile | User class | Special authorities | Group profile | Description |
|--------------|------------|---------------------|---------------|---|
| ZENDADMIN | *SECOFR | (ALL) | *NONE | Zend Core administrative user, used to start certain ZEND subsystem jobs and to run Zend Core pseudo random number generator job, which is used for encryption services |
| ZENDTECH | *USER | *NONE | *NONE | Used to update the PHP configuration |
| NOBODY | *USER | *NONE | NOGROUP | Used to run the Apache and Zend Core jobs (ZENDCOREAP) in i5/OS PASE |

| User profile | User class | Special authorities | Group profile | Description |
|--------------|------------|---------------------|---------------|---|
| NOGROUP | *USER | *NONE | *NONE | Group profile for the NOBODY user profile, specified in the i5/OS PASE Apache HTTP Server configuration |

Important: Passwords are not defined for these user profiles, so they cannot be used to sign on to the system.

Zend Core for i5/OS uses the following system-supplied user profiles:

- ▶ QSECOFR
The controlling job for the ZENDCOREAP group of jobs is run under QSECOFR.
- ▶ QTMHHTTP
The jobs supporting the IBM HTTP Server for i5/OS server (ZENDCORE) run under the default HTTP server user profile. These jobs run in the QHTTPSVR subsystem.
- ▶ QTCP
The job that supports the i5 Toolkit for Zend Core runs under the QTCP user profile.

Directories

Zend Core for i5/OS uses these main directory structures:

- ▶ /www/zendcore
This directory contains your IBM HTTP Server (powered by Apache) for i5/OS HTTP server instance. The subdirectories of this directory are as follows:
 - /www/zendcore/conf
This directory contains the httpd.conf file, which represents the configuration of the ZENDCORE server instance.
 - /www/zendcore/htdocs
This directory is the document root for both the IBM HTTP Server instance and the i5/OS PASE Apache server instance. Your PHP files and other application resources reside in this directory.
 - /www/zendcore/logs
This directory contains the log files (access_log and error_log, by default) for the ZENDCORE server instance
- ▶ /usr/local/Zend
This directory contains license and product information files. It also contains two important subdirectories:
 - /usr/local/Zend/apache2
This directory contains the Apache HTTP Server that runs in i5/OS PASE. Several subdirectories contain the product binaries. From an administrative standpoint, you only need to be familiar with the following subdirectories:
 - /usr/local/Zend/apache2/bin
This directory contains binaries for the Apache HTTP Server instance.

- /usr/local/Zend/apache2/conf
This directory contains the httpd.conf file, which represents your i5/OS PASE Apache server instance.
- /usr/local/Zend/apache2/logs
This directory contains the product log files (access_log and error_log, by default).
- /usr/local/Zend/Core
This directory contains the Zend Core product. Of the various subdirectories, the following directories are important:
 - /usr/local/Zend/Core/bin
This directory contains binaries for the Zend Core product.
 - /usr/local/Zend/Core/etc
This directory contains the configuration files for the PHP environment, most notably the php.ini file.
 - /usr/local/Zend/Core/logs
This directory contains the Zend Core product log files (php_error_log and ini_modifier_log).

3.2.9 Runtime environment

This section examines the Zend Core for i5/OS runtime environment as it pertains to administrative activities.

Components

To understand the technical details of Zend Core for i5/OS administration, it is helpful to take a high-level look at the runtime components (pictured in Figure 3-6) and how they interact with each other.

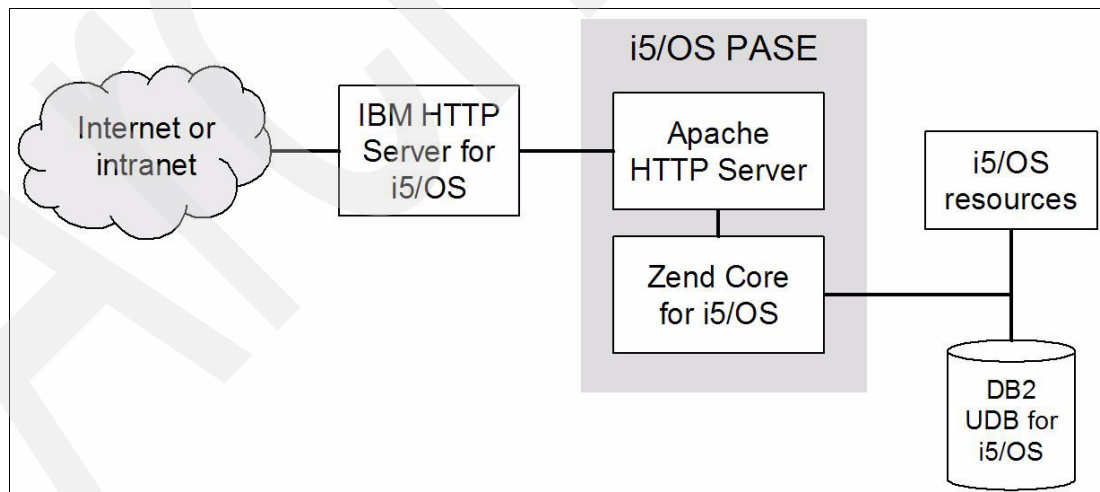


Figure 3-6 Zend Core for i5/OS runtime environment

These components make up the PHP environment runtime provided by Zend Core for i5/OS:

► IBM HTTP Server or i5/OS

Client requests are received by the IBM HTTP Server for i5/OS instance. The server instance is named ZENDCORE, and it listens on port 89 by default.

The server is set up to pass all requests to the Apache HTTP server running in i5/OS PASE. It also passes responses from the PASE-based Apache HTTP server back to the user's browser. In the first case, the i5/OS server acts as a *proxy*; in the second case, it acts as a *reverse proxy*.

Using the native i5/OS HTTP server for the front end has many natural advantages:

- The server instance can be managed with the IBM Web Administration for i5/OS tool. An i5-specific tool, Web Administration offers point-and-click and wizard-based configuration for your HTTP server.
- Using the proxy/reverse proxy scheme provides an additional layer of defense between the entry point into the server and where dynamic code is actually processed.
- This configuration enables us to easily add secure sockets (SSL or HTTPS) to our configuration. See Chapter 7, "Security" on page 87 for more details.

► **i5/OS PASE**

i5/OS PASE is the AIX® runtime in i5/OS. It is not an emulator: It takes advantage of the hardware common to both i5 and p5 systems and the ability to switch between the different runtimes. It allows AIX applications to be easily ported to System i5™, providing i5 ease-of-administration with the robustness of AIX applications.

► **Apache HTTP Server**

Because Zend Core for i5/OS runs in the i5/OS PASE environment, a second HTTP server instance is provided, also running in i5/OS PASE, to improve processing performance.

The Apache server provides the Web server environment for PHP applications. By default, it listens on port 8000.

As previously stated, requests are forwarded to the Apache server for processing by the IBM HTTP Server for i5/OS instance. The Apache server is configured to only accept requests from the local system (127.0.0.1).

► **Zend Core for i5/OS**

Zend Core provides the PHP runtime environment. Requests for PHP processing is passed from the Apache HTTP Server to Zend Core. The PHP engine in Zend Core interprets the PHP code and performs any necessary processing, such as querying a database or accessing system resources.

The PHP engine replaces the PHP code with the results of its processing and returns the response to the Apache HTTP Server to be sent back through the IBM HTTP Server to the user.

► **i5/OS data and resources**

PHP applications can access i5/OS data and resources such as program (*PGM) objects, files, data queues, commands, data areas, and so on. Various application programming interfaces (APIs) can be used to programmatically access these resources, which are discussed in Chapter 4, "Application development" on page 39.

Subsystems and jobs

The subsystems and job displayed in Figure 3-7 on page 26 are the defaults for Zend Core for i5/OS.

| Work with Active Jobs | | | | | | |
|-----------------------|---------------|--------------|------|-------|--------------|--------|
| Opt | Subsystem/Job | Current User | Type | CPU % | Function | Status |
| | QHTTPSVR | QSYS | SBS | .0 | | DEQW |
| | ZENDCORE | QTMHHTTP | BCH | .0 | PGM-QZHBMAIN | SIGW |
| | ZENDCORE | QTMHHTTP | BCI | .0 | PGM-QZSRLOG | SIGW |
| | ZENDCORE | QTMHHTTP | BCI | .0 | PGM-QZSRLOG | SIGW |
| | ZENDCORE | QTMHHTTP | BCI | .0 | PGM-QZSRHTTP | SIGW |
| | ZENDCORE | QTMHHTTP | BCI | .0 | PGM-QZSRHTTP | DEQW |
| | ZEND | QSYS | SBS | .0 | | DEQW |
| | I5_COMD | QTCP | ASJ | .0 | PGM-EASYCOMD | TIMW |
| | ZC_STR_PRN | ZENDADMIN | BCI | .0 | PGM-prngd | SELW |
| | ZENDCOREAP | QSECOFR | BCI | .0 | PGM-httpd | SELW |
| | ZENDCOREAP | QSECOFR | BCI | .0 | PGM-httpd | DEQW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | SELW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | TIMW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | TIMW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | TIMW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | TIMW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | TIMW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | TIMW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | TIMW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | TIMW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | TIMW |
| | ZENDCOREAP | NOBODY | BCI | .0 | PGM-httpd | TIMW |

Figure 3-7 Subsystems and jobs pertaining to Zend Core for i5/OS

Each subsystem and job has a particular task:

- ▶ QHTTPSVR subsystem: The ZENDCORE jobs support the IBM HTTP Server for i5/OS instance. The first job (with function PGM_QZHBMAIN) is the main job. The jobs with PGM-QZSRLOG function handle the logging function of the server instance. The remaining two jobs are for server processes.
- ▶ ZEND subsystem: Three job types run in the ZEND product subsystem:
 - I5_COMD: The service for the i5 PHP Toolkit
 - ZC_STR_PRN: The pseudo random number generator job, used for calculating encryption keys
 - ZENDCOREAP: The i5/OS PASE Apache HTTP Server jobs

3.3 Configuring Zend Core for i5/OS

This section addresses configuring the Zend Core for i5/OS environment.

3.3.1 Administration tools

Various tools are included with Zend Core and i5/OS that you can use to configure and administer your PHP environment.

Zend Core Setup Tool

Included in the ZENDCORE library, the Setup Tool is used to start, update, and troubleshoot the environment.

Access the Setup Tool by running the following command on the i5/OS command line:

```
GO ZENDCORE/ZCMENU
```

Figure 3-8 shows the Setup Tool.

```
ZCMENU                                Zend Core for IBM i5/OS Setup Tool

Select one of the following:

    1. Set Zend Core Web Administration Console password
    2. Update via Zend Network menu
    3. Run Support Tool

    5. Service Management menu

   90. Signoff

Selection or command
===> _

F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel   F23=WRKUSRJOB
Copyright Zend Technologies LTD (2006)
```

Figure 3-8 The Zend Core for IBM i5/OS Setup Tool

The main menu of the Setup Tool contains the following options:

- ▶ 1. Set Zend Core Web Administration Console password
Use this option to set or change the password you use to access the browser-based Zend Core console.
- ▶ 2. Update via Zend Network menu
Use this option to check for updates to the product. This process requires a Zend Network user ID and password (which you created when you registered with Zend to download the product) to access the Zend Network. You can also provide this user ID and password under this option.
The Update via Zend Network menu also enables you to roll back updates.
- ▶ 3. Run Support Tool
The support tool gathers diagnostic data and bundles it into a GZ file. (The location of the file is displayed in a message when you run the tool.) You can then send the file to Zend Support to assist them with problem determination.
- ▶ 5. Service Management menu
The Service Management menu contains options for starting and stopping the Zend subsystem; starting, restarting, and stopping the Apache HTTP Server instances; and starting and stopping the i5_COMD service (which is used by the i5 Toolkit for Zend Core).
- ▶ 90. Signoff
This option signs off your 5250 session.

Zend Core browser-based console

Zend Core includes a PHP application that can be used to edit the PHP configuration. Changes are written to the product php.ini file. It is recommended that you use the console to

configure Zend Core: Hand-editing the php.ini file can be error-prone and result in configuration problems.

To open the Zend Core console, open the following URL in a Web browser, replacing <system_name> with the host name or IP address of your i5 machine:

http://<system_name>:89/ZendCore

Note: Depending on your system's TCP/IP configuration, the console might not load if you use the fully qualified host name (host and domain) of the system in the URL. If this occurs, use the short host name in the URL.

Log in to the console with the password you specified during the installation process. Use the Zend Core Setup Tool to reset or change the password.

At the top of the page is the menu you use to navigate through the console, as shown in Figure 3-9. The top row contains tabs for the main console areas. Most items in the top row, when clicked, display a submenu in the second row that accesses pages in that particular console area.

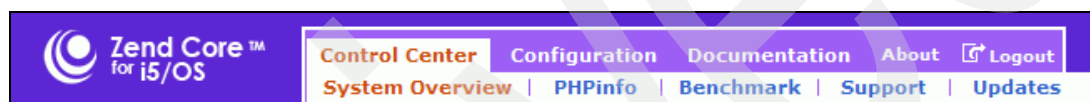


Figure 3-9 The Zend Core console navigation menu

Refer to the product documentation for a detailed list of all console features. Briefly, here is an overview of each tab and the activities that can be performed on those pages:

► **Control Center**

The Control Center area contains information about the product, including the system, loaded PHP modules, installed and available product updates, and how to get support. Of particular interest is the benchmark tool, which you can use to simulate hits to a particular URL and measure response performance.

► **Configuration**

The Configuration tab enables you to change the product configuration, including core PHP modules, PHP extensions, Zend products, server directives, and the Zend Studio Server (which integrates server-side functions with the Zend Studio development tool).

► **Documentation**

The Documentation tab provides access to the integrated product documentation for PHP, Zend Core, and PEAR. There is also a function for searching the documentation.

Many console pages contain controls for saving changes to the configuration and restarting the server. If changes you have made do not appear to take effect after you have clicked the link to restart the server, use the Zend Core Setup Tool to restart the servers or the Zend subsystem.

Click the **Logout** link to exit the Zend Core console.

IBM Web Administration for i5/OS

To manage your IBM HTTP Server for i5/OS instance, use the IBM Web Administration for i5/OS tool. It is a browser-based tool provided with the IBM HTTP Server for i5/OS product. The administrative application is served from the *ADMIN instance of IBM HTTP Server.

To access the tool, first start the *ADMIN instance. You can use System i Navigator or the following i5/OS command:

```
STRTCPSVR SERVER(*HTTP) HTTPSVR(*ADMIN)
```

After the *ADMIN instance is running, use a Web browser to open the following URL, where <server_name> is the host name or IP address of your i5 system:

http://<server_name>:2001/HTTPAdmin

When prompted, enter a user profile name with *IOSYSCFG authority and its associated password.

Note: If your browser uses a pop-up blocker, disable it before using the IBM Web Administration for i5/OS tool or configure it to access pop-up windows for the site. Some functions of the tool do not work correctly if pop-ups are blocked.

Figure 3-10 displays the main page of the Web Administration tool.



Figure 3-10 IBM Web Administration for i5/OS

Tabs at the top of the interface relate to various activities. Selecting certain tabs makes a row of subtabs appear below the main tabs. If you click **Manage** → **HTTP Servers** and select **ZENDCORE** from the Server list (as shown in Figure 3-11), you can display and change your IBM HTTP Server for i5/OS server instance.

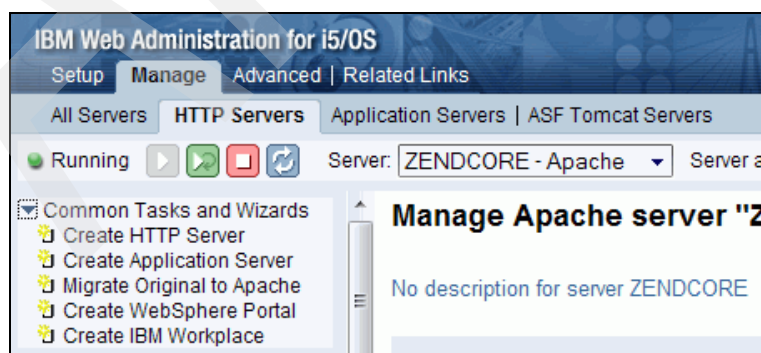


Figure 3-11 Managing the ZENDCORE Apache server instance

You can use the Web Administration tool to create other HTTP server instances.

Tip: Because the ZENDCORE instance forwards all content to the i5/OS PASE Apache HTTP Server, we recommend that you create another server instance for your static content (HTML pages, images, and others). Use the ZENDCORE instance only for PHP files.

Additionally, you can configure the ZENDCORE instance to use secure sockets layer (SSL) or authentication.

For more information about configuring IBM HTTP Server, see *IBM HTTP Server (powered by Apache): An Integrated Solution for IBM eServer™ iSeries™ Servers* (SG24-6716-02).

3.4 Configuring multiple instances

The long history of System i5 (and its predecessors) as a centralized, multi-user machine makes multiple-instance support a common requirement in i5/OS products. The following pages describe a few of the more common scenarios involving multiple instances.

Having the ability to host PHP applications from more than one web server is very important, and is fully supported. There are several ways to implement multiple instances of PHP under i5/OS. Note that the vast majority of configuration required to accomplish each scenario is specific to Apache rather than to i5/OS or the PHP environment. Thus, multiple instance configuration on i5/OS is largely identical to performing the same task on any other operating system that supports the Apache web server. Note that this list of scenarios is not exhaustive; by adding technologies such as virtual hosts one can imagine other variations.

At this time, Zend Core for i5/OS does not support multiple installations of the product on the same system partition. By “multiple installations” we mean the ability to have more than one version of the product installed within the same partition of i5/OS. For example, with IBM WebSphere Application Server it is possible to have versions 5.0, 6.0, and 6.1 all installed simultaneously within a single i5/OS partition. Most customers do not require this functionality, and can achieve the same results via different means (multiple i5/OS partitions, multiple hardware platforms, etc.).

Important: Remember that you should be signed on as QSECOFR when starting and stopping web server instances using the “apachectl” command. Using any other user profile makes it impossible for Zend Core for i5/OS to switch to the NOBODY user profile for the jobs under which PHP scripts execute.

3.4.1 Multiple i5/OS Apache instances, one PHP server

This scenario entails having multiple instances of IBM HTTP Server for i5/OS (i5/OS Apache), all forwarding their requests on to a single instance of Apache/PHP in PASE for i5/OS.

Figure 3-12 shows a representation of this scenario:

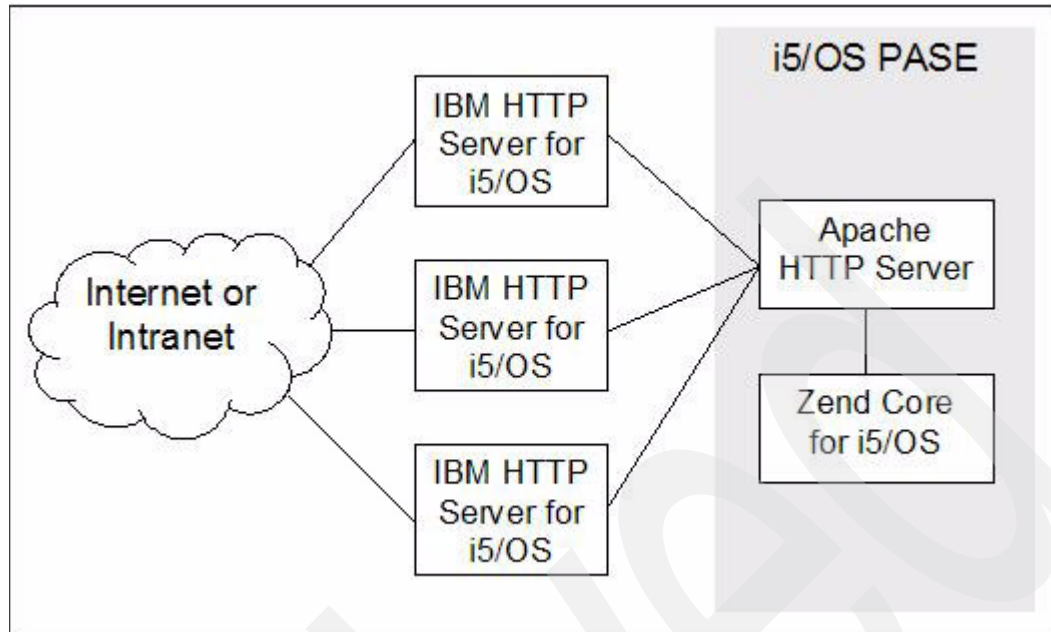


Figure 3-12 Multiple i5/OS Apache instances, one PHP server

In this case, we see three IBM HTTP Server for i5/OS instances. This could be accomplished in various ways - for example, we could have a single IBM HTTP Server for i5/OS instance listen on multiple ports. One example of where this may be a desirable option is when you want to have both HTTP (by default, port 80) and HTTPS (by default, port 443) exposed to the internet/intranet, but want only one PHP server to handle all these requests.

Figure 3-13 shows how this configuration could be done via the HTTP Server Administration utility available on port 2001. In this example we see a second port, in this case port 666, added to our ZENDCORE instance. The default reverse proxy configuration (done by the Zend Core for i5/OS installation program) is already configured for this instance, meaning that all requests received on ports 89 and 666 will be sent on to the internal (PASE for i5/OS) server listening on port 8000.

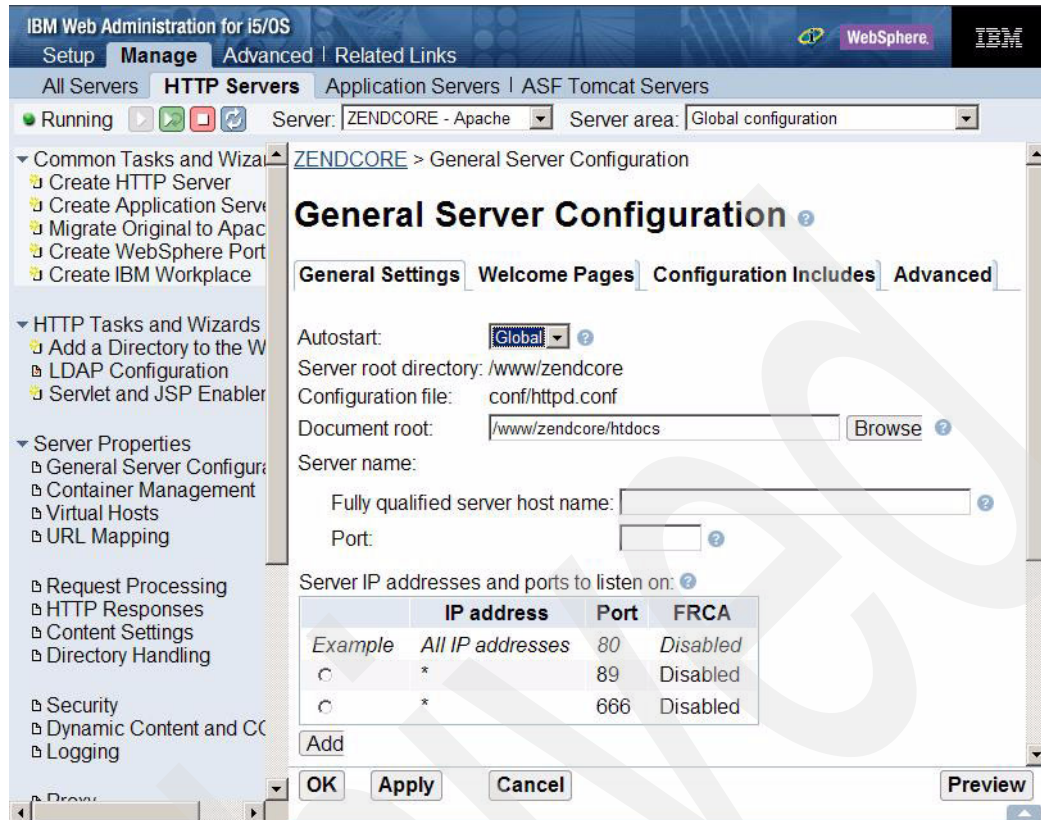


Figure 3-13 Adding a second port (666) to the ZENDCORE instance

3.4.2 Multiple i5/OS Apache instances, multiple PHP servers

This scenario involves having multiple instances of IBM HTTP Server for i5/OS (i5/OS Apache) forwarding requests to multiple instances of Apache/PHP running in PASE for i5/OS. There are several reasons you may want to consider such a scenario: for example, if you are hosting applications for other companies and want to ensure that the applications run in completely separate memory spaces. Another reason someone may consider such a scenario would be to allow a production PHP server to exist on the same physical partition where developers or system administrators are creating PHP applications or environments. Clearly we don't want to bring down the production server when a developer wants to make a change requiring a restart of the PHP environment.

Figure 3-14 shows a diagram outlining how this scenario might be visualized:

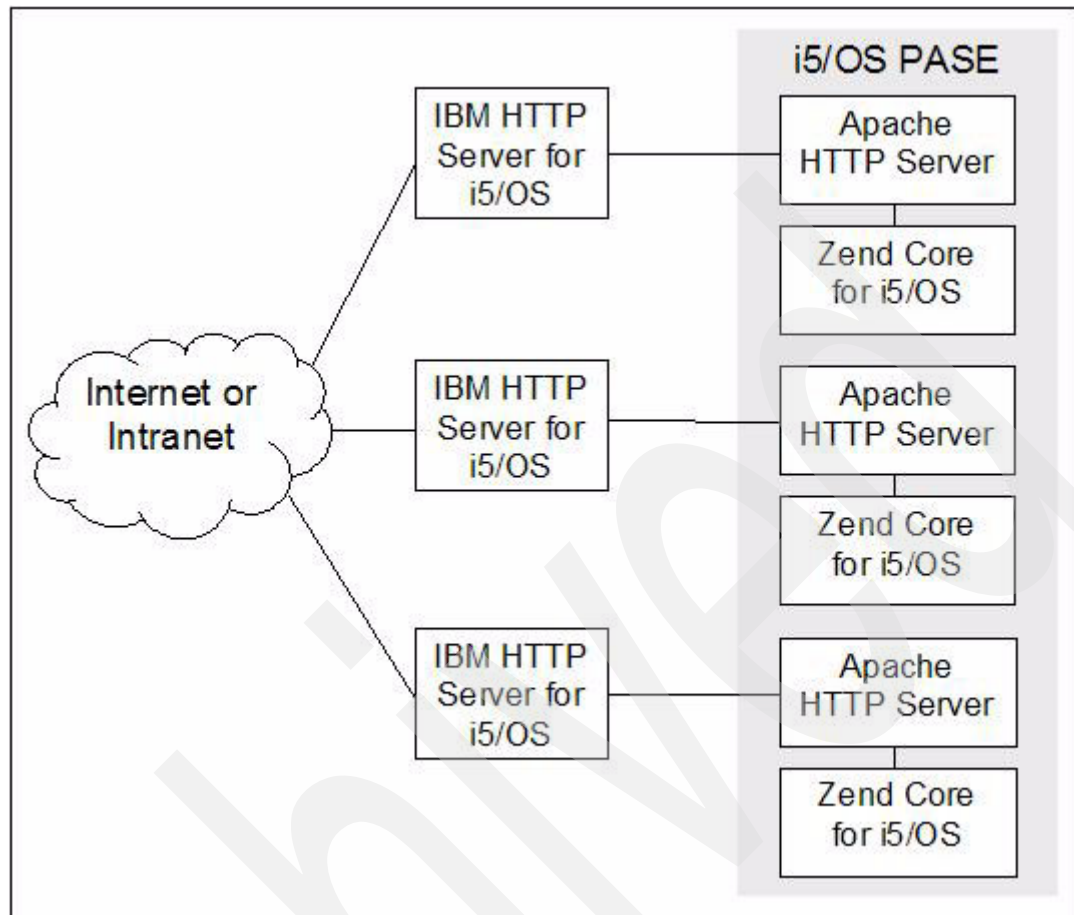


Figure 3-14 Multiple i5/OS Apache instances, multiple PHP servers

Again, there are many different ways one can create configurations to resemble the environment shown in Figure 3-14. We saw in “Multiple i5/OS Apache instances, one PHP server” on page 30 how we could add a second listening port to the existing ZENDCORE server configuration. Of course, another option would be to create a second configuration that can be independently started/stopped, and which listens on a different port.

The easiest way to allow for the creation of a new Apache/PHP instance in PASE for i5/OS is as follows:

First, create a new version of the “php.ini” file, which is found in directory “/usr/local/Zend/Core/etc.” In our case, we copied “php.ini” to a new file called “php2.ini.” Make any desired modifications to the new file.

Next, create a new version of the “httpd.conf” file found in directory “/usr/local/Zend/apache2/conf.” In our case, we created a copy of “httpd.conf” called “httpd2.conf” in the very same directory.

You must now edit the “httpd2.conf” file to make it a unique configuration. The following are some of the lines you should modify:

Example 3-1 Lines to change in httpd2.conf to create new Apache/PHP instance

```

PidFile logs/httpd2.pid
...
  
```

```
DocumentRoot "/www/ZendCore/htdocs2"
...
<Directory "/www/ZendCore/htdocs2">
...
```

Of course, there are others you may want to modify as well, depending on your configuration and your goals.

The following line should be added to “httpd2.conf” to point it at the new “php2.ini” file:

Example 3-2 Lines to add in httpd2.conf to create new Apache/PHP instance

```
### Start of Zend Core
PHPINIDir /usr/local/Zend/Core/etc/php2.ini
```

You should now be able to start this second instance by issuing a command similar to the following:

```
/usr/local/Zend/apache2/bin/apachectl
-f /usr/local/Zend/apache2/conf/httpd2.conf
-k start
```

As a reminder, the instance can then be stopped, started or restarted by adding the “-k” flag as follows:

| | |
|---------|------------|
| Stop | -k stop |
| Start | -k start |
| Restart | -k restart |

Important: Whenever issuing these commands make sure you also specify the “-f” flag and specify which configuration file to use.

This allows you to independently start or stop any IBM HTTP Server for i5/OS or Apache/PHP in PASE for i5/OS instance you like.

3.4.3 One i5/OS Apache instance, multiple PHP servers

In this scenario, we have a single instance of IBM HTTP Server for i5/OS (only one port exposed to the intranet/internet) sending requests to multiple Apache/PHP server instances. Most likely the decision of which Apache/PHP server the request should be routed to will be made based on the URL.

Figure 3-15 shows one way to view this scenario:

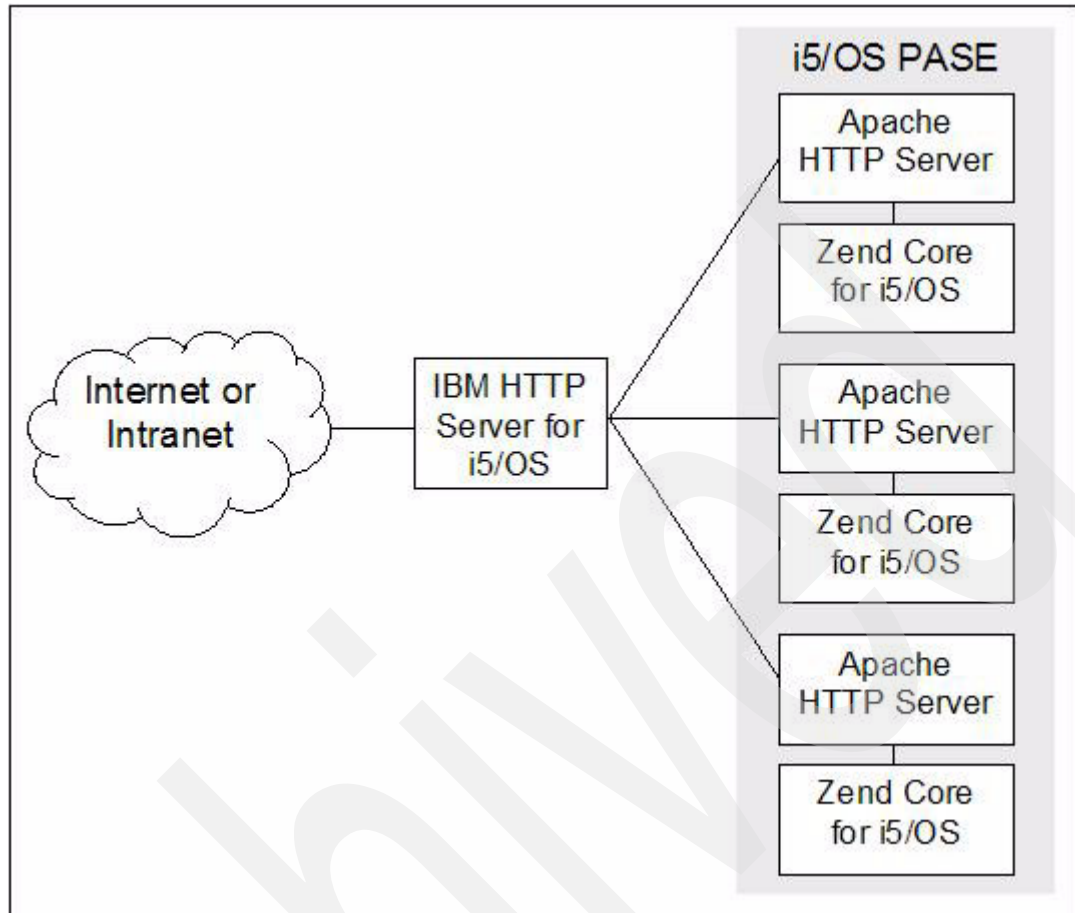


Figure 3-15 One i5/OS Apache instance, multiple PHP servers

To take advantage of both Apache/PHP servers we could configure our IBM HTTP Server for i5/OS instance using the IBM Web Administration for i5/OS tool (port 2001) as shown in Figure 3-16:

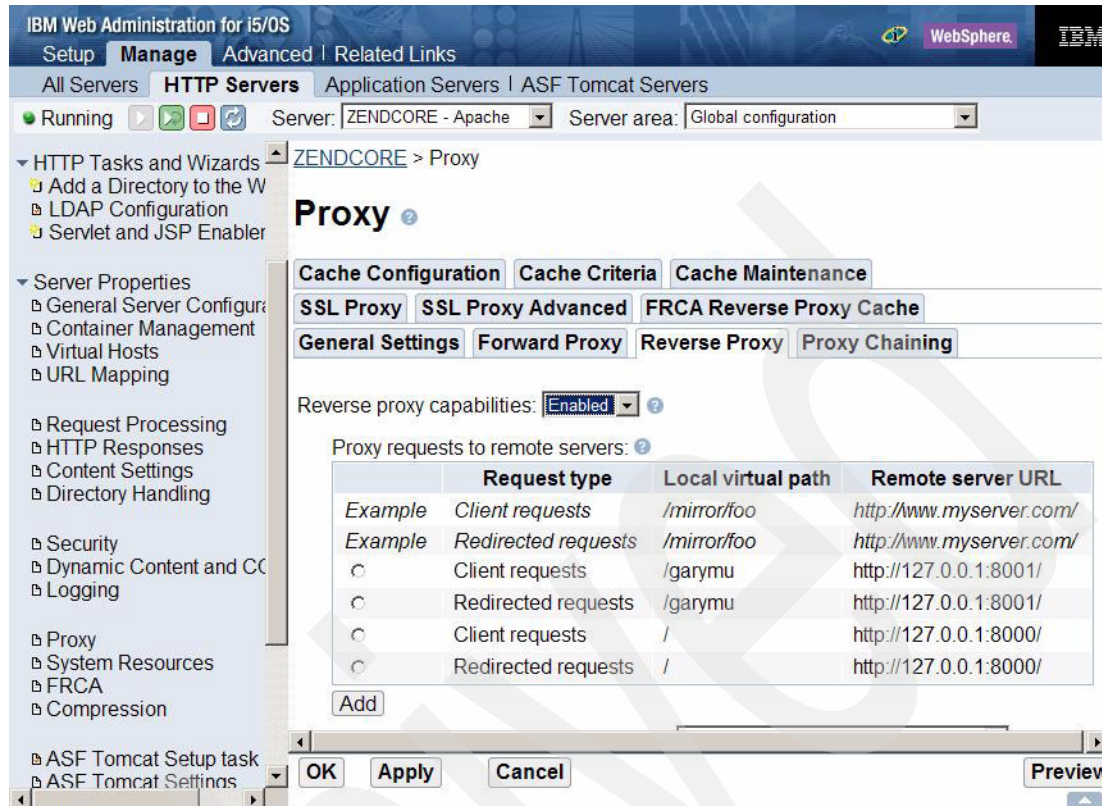


Figure 3-16 Adding second reverse proxy configuration

In this case, any incoming requests that specify “/garymu” on the URL (for example, “http://mysystem.ibm.com/garymu/myfile.php”) would be sent to the Apache/PHP instance in PASE for i5/OS that is listening on port 8001; all other requests (ones that do not specify “/garymu”) will be sent to the default server on port 8000. Note that the two configuration items for “/garymu” and port 8001 must be *above* those for “/” and port 8000, as “/” is a catch all. If it is first in the list, no requests will ever flow down the list to test for other matches.

3.5 Backing up Zend Core for i5/OS

Use the i5/OS save and restore commands to back up and restore the PHP environment.

To save and restore the Zend for i5/OS product libraries and directories, use SAVLICPGM and RSTLICPGM.

Use the Save Object (SAV) and Save Library (SAVLIB) commands for back up, and the Restore Object (RST) and Restore Library (RSTLIB) commands for restore operations. For more information about the commands, see “Systems management: Backup and recovery” in the V5R4 iSeries Information Center at:

<http://publib.boulder.ibm.com/infocenter/iseries/v5r4/topic/rzahg/rzahgbackup.htm>

In particular, you should periodically back up the following objects:

- ▶ /www/zendcore/htdocs/*

This directory (and any subdirectories) contain your PHP and other Web files.

- ▶ `/www/ZendCore/conf/httpd.conf`

If you have made configuration changes to the IBM HTTP Server for i5/OS server instance (ZENDCORE), you might want to back up its configuration file.

- ▶ `/usr/local/Zend/apache2/conf/*`

If you have customized the i5/OS PASE Apache HTTP Server, back up the configuration files in this directory.

- ▶ `/usr/local/Zend/Core/etc/*`

Changes to your Zend Core and PHP configuration are stored in files in this directory.

Archived

Application development

This chapter discusses the use of PHP as both as a Web and general scripting language. We discuss the software and concepts commonly used to develop dynamic applications with PHP and i5/OS data, objects, and programs.

We discuss the following topics in this chapter:

- ▶ PHP and Web development
- ▶ PHP as a command-line scripting language
- ▶ Development tools

4.1 PHP and Web development

PHP is mostly used for developing Web applications. Web applications most often comprise three tiers, which this chapter introduces to give you a starting point for exploration in the rest of the book.

We describe each type of software service in more detail, in the context of Web applications built on an architecture commonly found in IBM environments.

- ▶ The presentation tier

This tier serves as the user interface. A Web server fields requests from users and determines from its configuration whether it can service them itself by retrieving a document from the file system or whether it needs the assistance of another program in the logic tier to complete processing.

- ▶ The logic tier

The logic engine provides an interface to which the Web server can delegate requests. It performs the business logic of the application. It can accept input, perform operations on it, and return dynamically created output to the Web server to pass back to the client.

- ▶ The data tier

A data store manages persisted information and provides an interface to change or retrieve it based on rules and conditions.

Database servers, such as IBM DB2, Informix®, and Cloudscape, provide the services of the data tier. Apache HTTP Server and its IBM-enhanced offering IBM HTTP Server serve the role of the Web server. PHP is used as the business logic engine that most often generates a response, based on information stored in the database, to return to the Web server.

Figure 4-1 shows Web application infrastructure components and the HTTP request-response cycle.

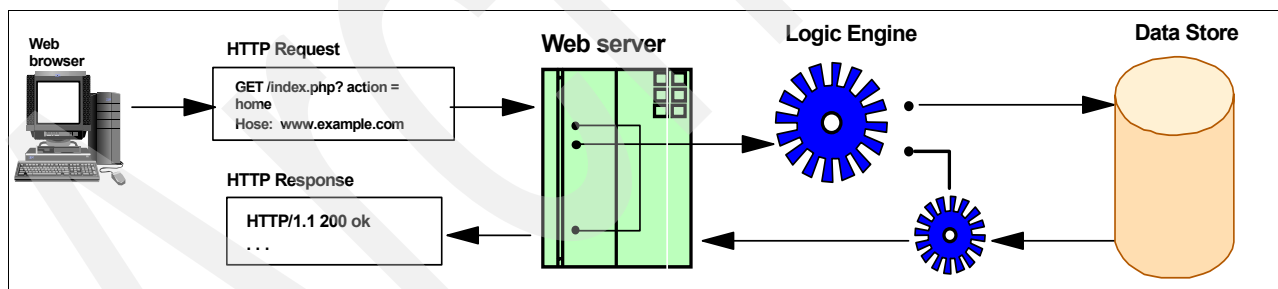


Figure 4-1 Web application infrastructure

4.1.1 Quick Web example

You can start writing Web applications quickly by embedding PHP code within HTML. Although embedded PHP code is less reusable than separate PHP files, the embedded approach is a fast way to develop small Web sites.

Example 4-1 HTML with embedded PHP

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<title>IBM Redbook: Hello World</title>
</head>
```

```

<h3>My first PHP web application</h3>
<p>
<?php echo "Hello, World" ?></p>
<body>
</body>
</html>

```

Put this file in the /www/zendcore/htdocs/ directory using Zend Studio for i5/OS, FTP, or any other technique that transfers files to i5/OS. You can then access your PHP application via a Web browser using the IP address of your Zend Core for i5/OS installation, as illustrated in Figure 4-2.

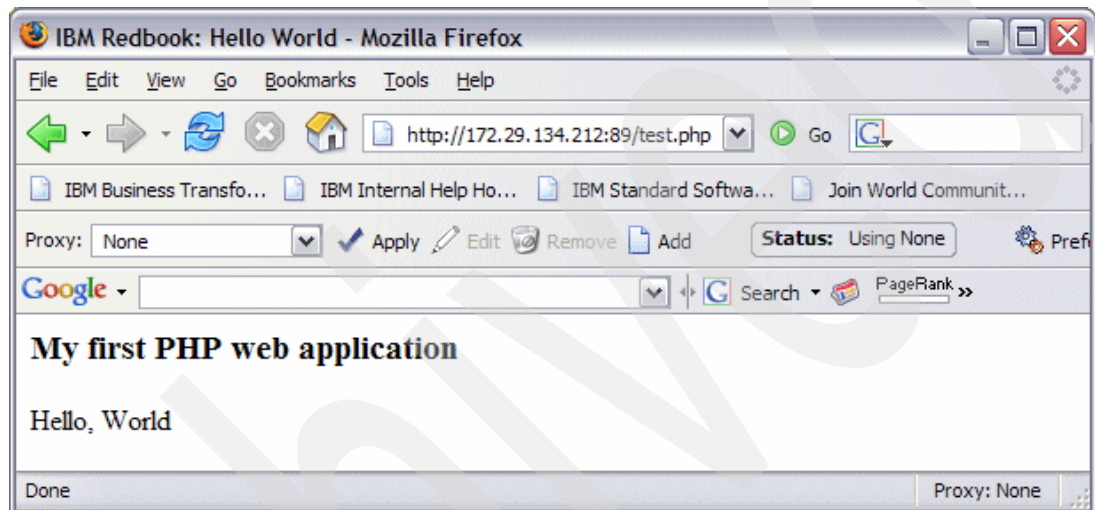


Figure 4-2 Hello world in PHP

That is all you need to do in order to make and test your first PHP application.

4.2 PHP as a command-line scripting language

Although PHP is mostly used for developing Web applications, it can also be used as a command-line scripting language as an alternative to CL. PHP has three command-line modes: interactive, one line at a time, and in scripts.

For details about using PHP as a command-line tool (not specifically for i5/OS), see the following Web address:

<http://www.php.net/manual/en/features.commandline.php>

4.2.1 PHP interactive mode

Interactive mode enables you to test bits of PHP code. To run PHP in interactive mode, type the following command in i5/OS PASE:

```
/usr/local/Zend/Core/bin/php -a
```

When PHP is in interactive mode, you can write PHP statements enclosed in `<?php` and `?>` delimiters. For example:

```
<?php for ($i=0; $i<10; $i++) { print "Counting $i\n"; } ?>
```

Usually PHP output is buffered, so if you want to disable buffering, you can enter following command:

```
<?php while (@ob_end_clean()); ?>
```

To exit from PHP interactive mode, issue following command:

```
<?php exit ?>
```

Figure 4-3 illustrates an example of using PHP in interactive mode.

```
/Q0penSys/usr/bin/-sh

Interactive mode enabled

> <?php while (@ob_end_clean()); ?>
> <?php for ($i=0;$i<10;$i++) { print "Counting $i\n"; } ?>
Counting 0
Counting 1
Counting 2
Counting 3
Counting 4
Counting 5
Counting 6
Counting 7
Counting 8
Counting 9

===> <?php exit ?>

F3=Exit      F6=Print    F9=Retrieve  F11=Truncate/Wrap
F13=Clear    F17=Top     F18=Bottom   F21=CL command entry
```

Figure 4-3 Example of PHP interactive mode

4.2.2 PHP one-line scripts (popularly called oneliners)

To use PHP as a one-line script interpreter, you have to use the -r command line option. For example:

```
/usr/local/Zend/Core/bin/php -r 'for ($i=0;$i<10;$i++) { print "Counting $i\n"; }'
```

When this example is executed, the output will be similar to Figure 4-4.

```
/Q0penSys/usr/bin/-sh

$
> /usr/local/Zend/Core/bin/php -r 'for ($i=0;$i<10;$i++) { print "Counting
$i\n
"; }'
Counting 0
Counting 1
Counting 2
Counting 3
Counting 4
Counting 5
Counting 6
Counting 7
Counting 8
Counting 9
$

====>

F3=Exit      F6=Print    F9=Retrieve  F11=Truncate/Wrap
F13=Clear    F17=Top     F18=Bottom   F21=CL command entry
```

Figure 4-4 Output of PHP one line script

4.2.3 PHP and scripting

To run command-line scripts in PHP, create a file with PHP code as illustrated in Example 4-2, and later execute it as shown (using myfile.php as example):

```
/usr/local/Zend/Core/bin/php myfile.php
```

Example 4-2 PHP scripting example (myfile.php)

```
<?php
for ($i=0;$i<10;$i++)
{
    print "Counting $i\n";
}
?>
```

Output will be similar to that shown in Figure 4-5.

```
/Q0penSys/usr/bin/-sh

kost      myfile.php  pero.php  test.php  testzend.php
$
> /usr/local/Zend/Core/bin/php myfile.php
Counting 0
Counting 1
Counting 2
Counting 3
Counting 4
Counting 5
Counting 6
Counting 7
Counting 8
Counting 9
$

===>

F3=Exit    F6=Print  F9=Retrieve F11=Truncate/Wrap
F13=Clear  F17=Top   F18=Bottom F21=CL command entry
```

Figure 4-5 PHP scripting output

To simply check the syntax of PHP file, you can do it by using the following -l command line option:

```
/usr/local/Zend/Core/bin/php -l myfile.php
```

If no syntax errors are found, the output will be similar to that shown in Figure 4-6.

```
/Q0penSys/usr/bin/-sh

$
> cd /www/ZendCore/htdocs/
$
> /usr/local/Zend/Core/bin/php -l myfile.php
No syntax errors detected in myfile.php
$

===>

F3=Exit    F6=Print  F9=Retrieve F11=Truncate/Wrap
F13=Clear  F17=Top   F18=Bottom F21=CL command entry
```

Figure 4-6 PHP syntax check

4.2.4 Calling PHP from CL

It is possible to call PHP scripts from within CL applications. The following examples show how to invoke a PHP script that sends an e-mail using the built-in “mail” function.

Example 4-3 shows the PHP script that does the actual e-mail send operation. Of course you will want to replace the sender and recipient to actual values, and you must have SMTP configured correctly in your PHP configuration.

Example 4-3 PHP script: emailme.php

```
<?php
// establish "from" user (optional)
ini_set("sendmail_from","sender@example.org");

// Parameters: recipient, subject, body
mail("me@example.org", "Sent from CL", "CL can call PHP scripts.");

?>
```

Example 4-4 shows the CL program (compiled as CALLPHP) that invokes the previous PHP script via PASE for i5/OS.

Example 4-4 CL program: CALLPHP

```
CALL          PGM(QP2SHELL) +
               PARM('/usr/local/zend/Core/bin/php' +
                   '/www/zendcore/htdocs/emailme.php')
```

You can invoke the CALLPHP CL program the same way any other program under i5/OS:

```
CALL CALLPHP
```

Example 4-5 shows the results of calling the CL program.

Example 4-5 Result from invoking CALLPHP CL program

```
From: sender@example.org
To: <me@example.org>
Subject: Sent from CL
```

```
CL can call PHP scripts.
```

4.3 Development tools

Although PHP can be edited with text editors such as Notepad, the use of a powerful Integrated Development Environment (IDE) can greatly speed development. Thanks to PHP's popularity, many IDEs are available. Table 4-1 on page 46 can help you choose the right tool for your needs.

Table 4-1 List of popular PHP development environments

| Name | License | Source control | File transfer | Web address |
|-----------------------|--|--------------------------|--------------------|---|
| Zend Studio for i5/OS | Commercial (free for i5/OS customers who have Zend Core for i5/OS) | CVS/SVN | FTP | http://www.zend.com |
| Eclipse | Free | CVS and SVN(with plugin) | FTP, Web DAV, SFTP | http://www.eclipse.org |
| Maguma | Free/commercial | CVS | FTP | http://www.maguma.com |
| NuSphere PhpED | Commercial | CVS | FTP, SFTP, Web DAV | http://www.nusphere.com |

Many more can be found at the following site:

<http://www.php-editors.com/review/>

There you can find reviews of both commercial and free IDEs as well as editors and feedback by actual users of the products.

4.3.1 Zend Studio for i5/OS

Zend Studio for i5/OS (Figure 4-7) is the default development environment for Zend Core. It comes with an integrated debugger, CVS/SVN, and FTP support.

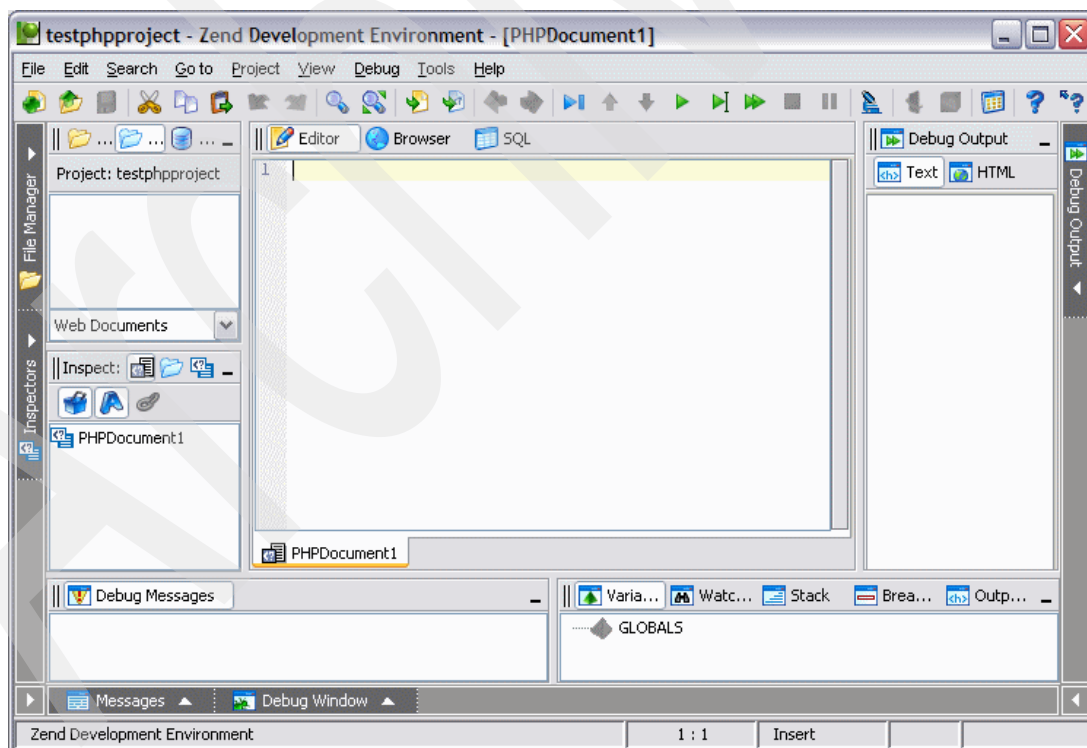


Figure 4-7 Zend Development Environment: Zend Studio for i5/OS

Zend Studio for i5/OS is available for Linux, Mac OS X, and Windows® platforms. Zend Studio for i5/OS is free for customers who downloaded Zend Core for i5/OS, and it is a solid choice for PHP development. Zend Studio for i5/OS can be downloaded from Zend's Web site:

https://www.zend.com/core/oem_registration.php?access_code=IBMi50SZend

For more information you can visit the following Web address:

http://www.zend.com/products/zend_core/zend_for_i5_os

4.3.2 Eclipse

Eclipse is based on an open and extensible platform written in Java. Although Eclipse was originally used almost exclusively for Java development, there are a number of good PHP plug-ins that extend Eclipse to support PHP. Eclipse is available for various platforms including Windows and Linux. Figure 4-8 shows Eclipse with the PHPEclipse plug-in.

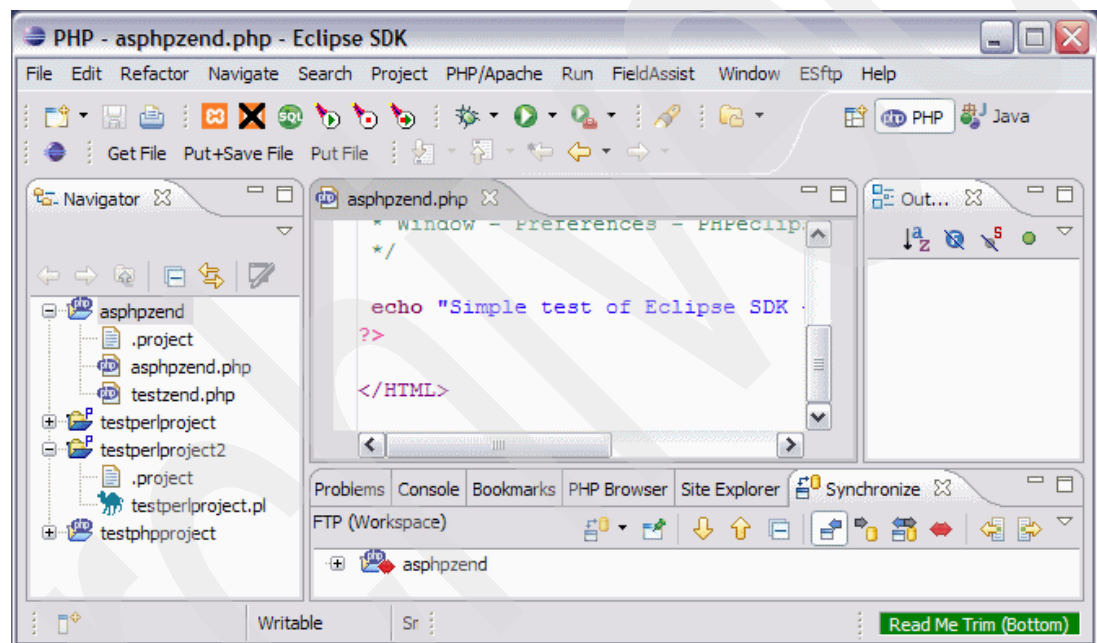


Figure 4-8 Eclipse with PHPEclipse plug-in

There are commercial and free Eclipse plug-ins to support PHP. One of the most popular free PHP plug-ins is PHPEclipse, which can be downloaded from:

<http://phpeclipse.de/>

One of the most popular commercial PHP plug-ins is TruStudio. There is also a free version of TruStudio. You can find out more about TruStudio at:

<http://www.xored.com/trustudio>

4.3.3 NuSphere PhpED

NuSphere PhpED (Figure 4-9) is a robust PHP integrated development environment.

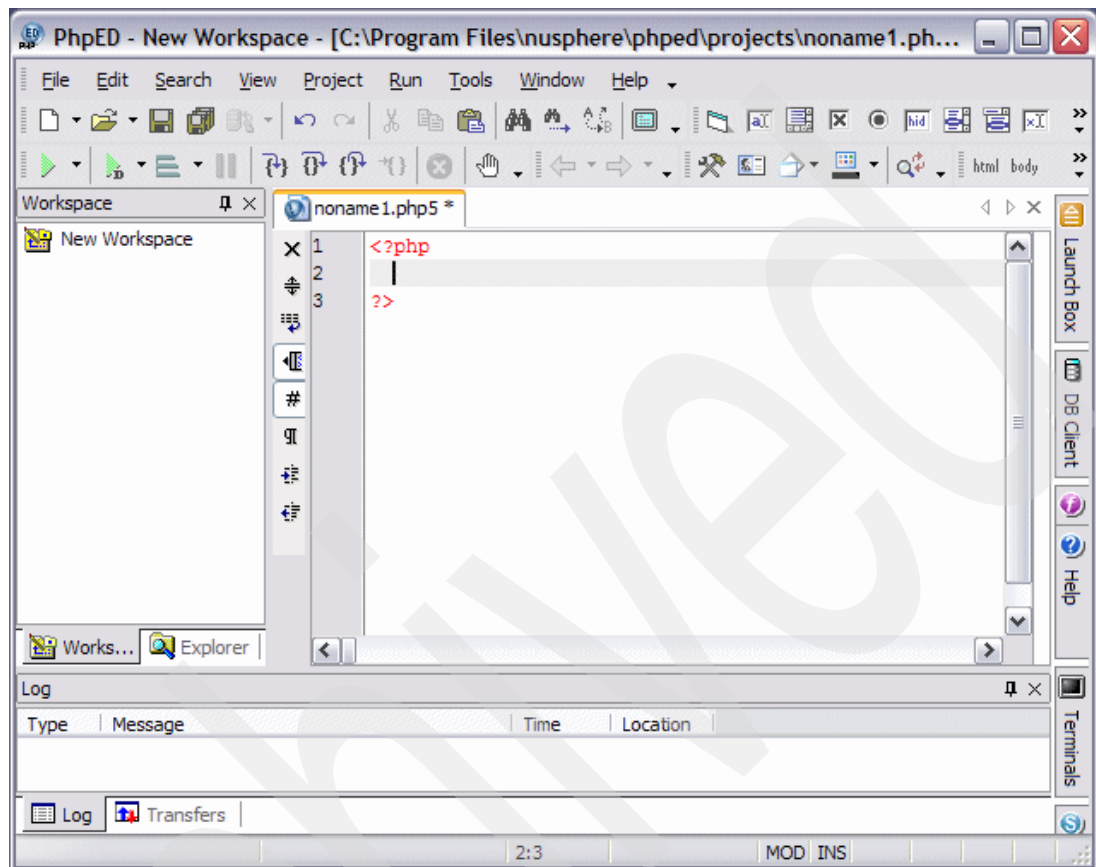


Figure 4-9 NuSphere PhpED

NuSphere PhpED is available only for the Windows platform. An evaluation version can be downloaded from the following Web site:

<http://www.nusphere.com/download.php.ide.htm>

4.3.4 Maguma

Maguma has different versions of their IDE for PHP. There is a free version (Maguma Open Studio, shown in Figure 4-10) as well as a commercial version (Maguma Workbench).

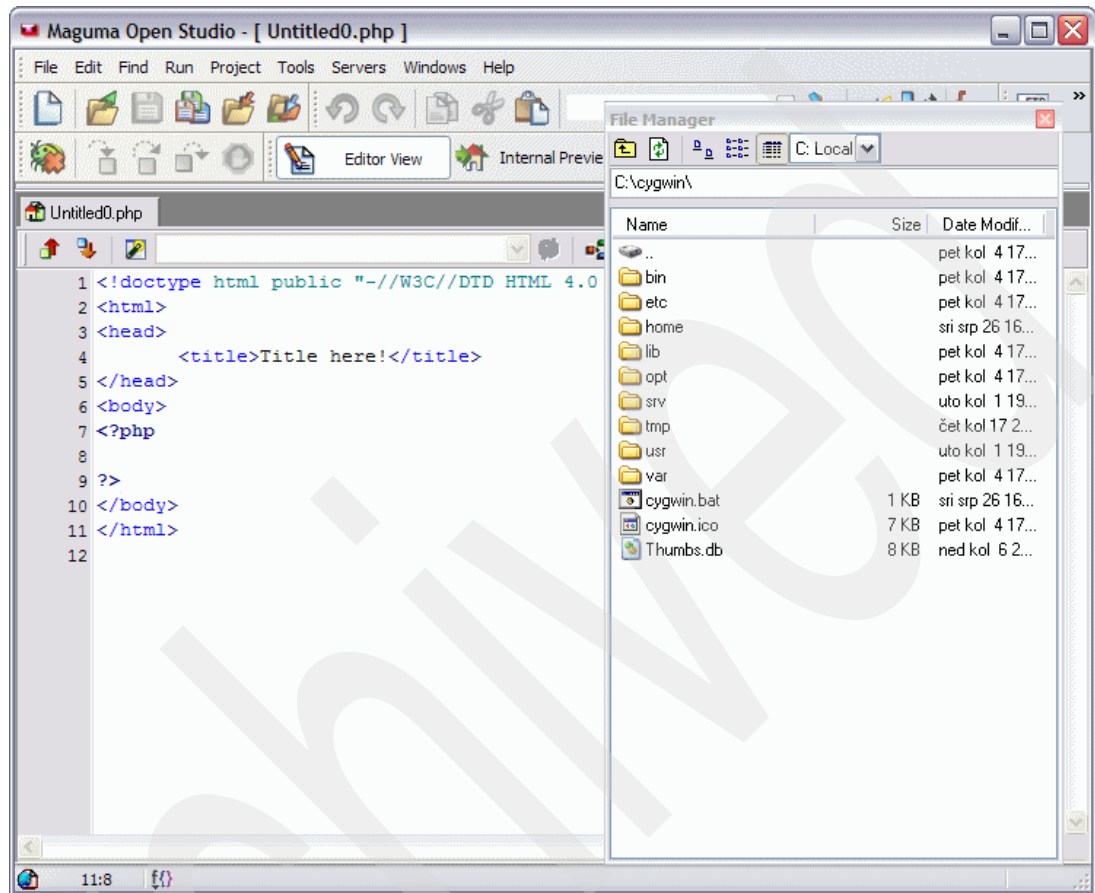


Figure 4-10 Maguma Open Studio

Maguma products are available for Windows and Linux. They can be downloaded from:

<http://www.maguma.com/en/download.html>

4.4 i5 PHP API Toolkit functions

The i5 PHP API Toolkit enables PHP to access native i5/OS resources. With the i5 PHP API Toolkit, you can call specific i5/OS functions or get native access to i5/OS data files.

Important: This function goes by various names; for example, from the Zend configuration menus it is called the “i5_COMD service.”

All i5/OS APIs in the i5 PHP API Toolkit are documented in the Zend Core for i5/OS User Guide. We do not offer details here, but just provide an overview of the functionality provided.

4.4.1 Main functions

Here is an overview of what you can do with documented functions:

- ▶ Connection Management
 - `i5_connect`
 - `i5_close`
 - `i5_adopt_authority`
 - `i5_error`
 - `i5_errormsg`
- ▶ CL calls
 - `i5_command`
- ▶ Program calls
 - `i5_program_prepare`
 - `i5_program_call`
 - `i5_program_close`
- ▶ Data retrieval
 - `i5_fetch_array`
 - `i5_fetch_assoc`
 - `i5_fetch_object`
 - `i5_fetch_row`
 - `i5_info`
 - `i5_field_len`
 - `i5_field_name`
 - `i5_field_scale`
 - `i5_field_type`
 - `i5_list_fields`
 - `i5_num_fields`
 - `i5_result`
- ▶ Native file access
 - `i5_open`
 - `i5_addnew`
 - `i5_edit`
 - `i5_delete`
 - `i5_cancel_edit`
 - `i5_setvalue`
 - `i5_update`
 - `i5_range_from`
 - `i5_range_to`
 - `i5_range_clear`
 - `i5_data_seek`
 - `i5_seek`
 - `i5_bookmark`
 - `i5_free_file`
 - `i5_new_record`
 - `i5_update_record`
 - `i5_delete_record`
 - `i5_get_keys`
- ▶ System Values
 - `i5_get_system_value`

- ▶ Data Areas
 - i5_data_area_create
 - i5_data_area_read
 - i5_data_area_write
 - i5_data_area_delete
- ▶ Data Queue
 - i5_dtaq_prepare
 - i5_dtaq_receive
 - i5_dtaq_send
 - i5_dtaq_close
- ▶ Job Log List
 - i5_jobLog_list
 - i5_jobLog_list_read
 - i5_jobLog_list_close
- ▶ Objects List
 - i5_objects_list
 - i5_objects_list_read
 - i5_objects_list_close
- ▶ Spool File
 - i5_spool_list
 - i5_spool_list_read
 - i5_spool_get_data
 - i5_spool_list_close
- ▶ User Space
 - i5_userspace_create
 - i5_userspace_prepare (new)
 - i5_userspace_get
 - i5_userspace_put (new)

In order to access i5/OS resources, we first need to connect to i5/OS. Example 4-6 illustrates how to connect, which is a requirement when performing most of the i5 PHP API Toolkit calls.

Example 4-6 i5 PHP API Toolkit connection to i5/OS

```

<HTML>
<?php
/* Connect to server */
$conn = i5_connect("localhost", "PHPUSER", "MYPASSWORD");
if (!$conn)
    die("<br>Connection using \"localhost\" with USERID and PASSWORD failed. Error
number = ".i5_errno()." msg=".i5_errormsg())."<br>");
else
    echo "<br>Connection using \"localhost\" with USERID and PASSWORD OK!<br>\n";
/* Close connection */
i5_close($conn);
?>
</HTML>

```

Important: Prior to Version 1.6 it was mandatory for both the user ID and password to be specified in upper case. Starting with V1.6, this restriction was lifted.

When connected (in these examples with user PHPUSER with password MYPASSWORD) we are able to access i5/OS resources. Example 4-7 on page 52 shows how to perform CL calls by

using `i5_command`. It does that by calling Retrieve Network Attributes command (RTVNETA) and then displays its output.

Example 4-7 i5_command example

```
<HTML>
<?php
/* Connect to server */
$conn = i5_connect("localhost", "PHPUSER", "MYPASSWORD");
if (!$conn)
    die("<br>Connection using \"localhost\" with USERID and PASSWORD failed. Error
number = ".i5_errno()." msg=".i5_errormsg())."<br>";
else
    echo "<br>Connection using \"localhost\" with USERID and PASSWORD OK!<br>\n";

/* Call Retrieve Network Attributes command */
$ret = i5_command("rtvneta", array(), array("sysname" => "sysn",
"lclnetid"=>"lclnet"));
if (!$ret) die("<br>rtvneta command failed. errno=".i5_errno()."
msg=".i5_errormsg());
print "<h1><b>Results of \"rtvneta\" command </b></h1><br>" ;
print "System Name : $sysn<br>" ;
print "Local Net ID : $lclnet<br>" ;

/* Close connection */
i5_close($conn);
?>
</HTML>
```

Example 4-8 shows how to perform an i5/OS program call in PHP. First, we prepare the program call by using `i5_program_prepare`, then execute by calling `i5_program_call`. We end by calling `i5_program_close`.

Example 4-8 Program call example

```
<HTML>
<?php
/* Connect to server */
$conn = i5_connect("localhost", "PHPUSER", "MYPASSWORD");
if (!$conn)
    die("<br>Connection using \"localhost\" with USERID and PASSWORD failed. Error
number = ".i5_errno()." msg=".i5_errormsg())."<br>";
else
    echo "<br>Connection using \"localhost\" with USERID and PASSWORD OK!<br>\n";

$description = array(
    array("Name"=>"FIRST", "IO"=>I5_IN, "Type"=>I5_TYPE_CHAR, "Length"=>"15"),
    array("Name"=>"LAST", "IO"=>I5_IN, "Type"=>I5_TYPE_CHAR, "Length"=>"15"),
    array("Name"=>"ACCOUNT", "IO"=>I5_OUT, "Type"=>I5_TYPE_CHAR, "Length"=>"15"),
    array("Name"=>"AMOUNT", "IO"=>I5_INOUT, "Type"=>I5_TYPE_PACKED,
"Length"=>"5.2")
);

$pgm = i5_program_prepare("PHPLIB/INCRAMT", $description);
if (!$pgm) die("<br>Program prepare error. Error number = ".i5_errno()."
msg=".i5_errormsg());
```



```

$parmIn = array(
    "FIRST"=>$_POST["first"],
    "LAST"=>$_POST["last"],
    "AMOUNT"=>$_POST["amount"]
);
$parmOut = array(
    "FIRST"=>"FIRST",
    "LAST"=>"LAST",
    "ACCOUNT"=>"ACCOUNT",
    "AMOUNT"=>"AMOUNT"
);
$ret = i5_program_call($pgm, $parmIn, $parmOut);
if (!$ret) die("<br>Program call error. Error number=".i5_errno()."
msg=".i5_errormsg());
echo "<BR>FIRST :". $FIRST;
echo "<BR>LAST : $LAST";
echo "<BR>AMOUNT : $AMOUNT";
echo "<BR>ACCOUNT : $ACCOUNT";

/* Close program call */
i5_program_close($pgm);

/* Close connection */
i5_close($conn);
?>
</HTML>

```

Example 4-9 is another example of performing i5 calls. This example shows how to get a specific system value. In our example we try to get the value of QDATE.

Example 4-9 Get System Value example

```

<HTML>
<?php
/* Connect to server */
$conn = i5_connect("localhost", "PHPUSER", "MYPASSWORD");
if (!$conn)
    die("<br>Connection using \"localhost\" with USERID and PASSWORD failed. Error
number =".i5_errno()." msg=".i5_errormsg())."<br>";
else
    echo "<br>Connection using \"localhost\" with USERID and PASSWORD OK!<br>\n";

/* Retrieve System DATE value */
print "Date is: ".i5_get_system_value("QDATE");

/* Close connection */
i5_close($conn);

```

For more examples and a full reference of available functions, refer to the Zend Core for i5/OS user guide.

4.5 XML support

One of the major reasons why PHP is so popular is that it is powerful but simple to use. The following short example provides evidence for this statement. First we look at a simple XML file that contains some last and first names (Example 4-10). We called this file myXML.xml.

Example 4-10 Sample XML file containing names and first names

```
<?xml version="1.0" encoding="iso-8859-1"?>
<myXML>
<person>
<name>Anderson</name>
<firstname>Melissa</firstname>
</person>
<person>
<name>Mullen-Schultz</name>
<firstname>Gary</firstname>
</person>
<person>
<name>Kosturjak</name>
<firstname>Vlatko</firstname>
</person>
</myXML>
```

Now have a look at the sample HTML document that contains some PHP script as shown in Example 4-11. After checking for the existence of the file, we load the file using one statement and then simply loop through and print out the elements read from the XML file, which in this case are the names of the authors of this book.

Example 4-11 HTML/PHP document that displays the content of the XML file

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<title>Read an XML file</title>
</head>
<body>
<h3>IBM Redbook authors</h3>
<?php
if (file_exists('myXML.xml')) {
    $x = simplexml_load_file('myXML.xml');
    foreach ($x->person AS $person) {
        printf($person->name . ' ' . $person->firstname . '<br>');
    }
}
else {
    exit('Failed to open myXML.xml.');
```

The SimpleXML extension, which was introduced in PHP 5, simplifies common XML tasks. SimpleXML provides the `simplexml_load_file()` command, used in our examples to load XML data into an array. Example 4-12 on page 55 shows how to implement an RSS feed

parser using `simplexml_load_file()`. It will grab blog posts from technorati tagged with the tag IBM and use RSS to display them.

Example 4-12 RSS implementation using PHP `simplexml_load_file()`

```
<HTML>
<?php
$rssurl="http://feeds.technorati.com/feed/posts/tag/ibm";
foreach(simplexml_load_file($rssurl)->channel->item as $i) {
    echo '<p><a href="'. $i->link. '">'. $i->title. '</a><br />';
    echo $i->description. '</p>';
}
?>
</HTML>
```

Note: Example 4-12 is just an example and is not ready for production. For example, you would likely want to implement caching for production use.

4.6 XML-RPC support

XML-RPC is a very simple remote procedure call protocol. XML-RPC uses XML to encode its calls and HTTP as a transport mechanism. To read more about XML-RPC, we recommend that you reference the following (case-sensitive) address:

<http://en.wikipedia.org/wiki/XML-RPC>

In short, XML-RPC can be used as a simple alternative to Simple Object Access Protocol (SOAP). XML-RPC is best suited for small projects requiring remote procedure calls where SOAP would be too complex to use. Refer to 4.7, “SOAP support” on page 56 for more information about that technology.

Because XML-RPC support is not enabled by default in Zend Core for i5/OS, you must enable the XML-RPC extension via the Zend Administration Panel as illustrated in Figure 4-11.

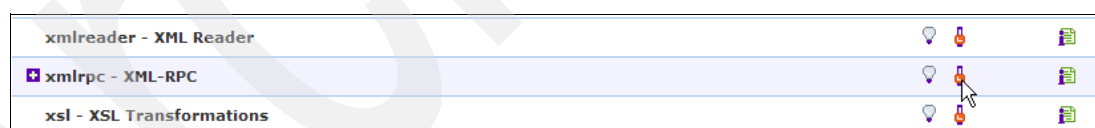


Figure 4-11 XML-RPC extension enablement

The code in Example 4-13 implements a simple XML-RPC service provider with function `hellofunc`, which returns the string `hello world`.

Example 4-13 XML-RPC service provider example

```
<HTML>
<?php
function hellofunc() {
    return "hello world";
}

$xmlrpc = xmlrpc_server_create();
xmlrpc_server_register_method($xmlrpc, 'hellofunc', 'hellofunc');
$r = xmlrpc_server_call_method($xmlrpc, $GLOBALS['HTTP_RAW_POST_DATA'], '');
echo $r;
```

```
xmlrpc_server_destroy($xmlrpc);  
?>  
</HTML>
```

For more information regarding the XML-RPC implementation in PHP, visit:

<http://www.php.net/manual/en/ref.xmlrpc.php>

As an alternative to XML-RPC built-in support in Zend Core, there is also a pure PHP implementation of XML-RPC. You can download XML-RPC for PHP at:

<http://phpxmlrpc.sourceforge.net/>

XML-RPC for PHP is written completely in PHP, and it is very easy to implement XML-RPC services using this library. Note that the code is not officially supported by IBM or Zend.

4.7 SOAP support

Implementing Web services in PHP is possible using SOAP, which is a successor of XML-RPC. SOAP has more features than XML-RPC, but those features come with a price: complexity. For more information about SOAP, visit (case-sensitive):

<http://en.wikipedia.org/wiki/SOAP>

The SOAP extension in PHP helps reduce the complexity of using SOAP. Zend Core for i5/OS already comes with the SOAP extension enabled. Example 4-14 shows how simple it is to write a SOAP service in PHP.

Example 4-14 SOAP service provider example

```
<?php  
function EchoIt($val) {  
    return $val;  
}  
  
$soapsrv = new SoapServer(null, array('location'=>"http://xx.xx.xx.xx/soap.php",  
    'uri'=>"http://xx.xx.xx.xx/"));  
$soapsrv->addFunction("EchoIt");  
$soapsrv->handle();  
?>
```

Example 4-14 demonstrates a SOAP service that returns the value passed in to the service. For more information regarding SOAP implementation in PHP, we recommend:

<http://www.php.net/manual/en/ref.soap.php>

Example 4-15 shows how to create a Web service for an existing RPG program.

Example 4-15 Creating Web service for an existing RPG program

```
<?php  
/*  
This service invokes a RPG program with two parameters  
*/  
class i5_program_service {  
  
    private $conn = false;
```

```

function __construct() {

    $this->conn = i5_connect('127.0.0.1', 'user', 'password'/*, $connection_pa
rameters*/);
    if (!is_resource($this->conn)) {
        throw new SoapFault('i5_program_service', 'Connection to i5 server fa
iled, use i5_errormsg() to get the failure reason');
    }

}

public function service_for_i5_program($var_0, $var_1) {

    $description = Array (
        array ('Name' => 'code', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length
' => '10'),
        array ('Name' => 'name', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length
' => '10'));

    $prog = i5_program_prepare('eacdemo/teststp2', $description, $this->conn);

    if (is_resource($prog)) {

        /* Execute Program */
        $params = array (
            'code' => $var_0, 'name' => $var_1);

        $retvals = array(
            'code' => 'ret_val_1', 'name' => 'ret_val_2');

        $ret = i5_program_call($prog, $params, $retvals) ;

        if ($ret === true) {
            $ret = array($ret_val_1, $ret_val_2);
            return $ret;
        }

        else {
            throw new SoapFault('i5_program_service', 'Failed to call the prog
ram, use i5_errormsg() to get the failure reason');
        }

        if (!i5_program_close ($prog) ) {
            throw new SoapFault('i5_program_service', 'Failed to free program
resource handle, use i5_errormsg() to get the failure reason');
        }
    }

    else {
        throw new SoapFault('i5_program_service', 'Program prepare failed, use
i5_errormsg() to get the failure reason');
    }
}

```

```
    }

    function __destruct() {
        if (!i5_close($this->conn)) {
            // Failed to disconnect from i5 server, use i5_errormsg() to get the f
            ailure reason
        }
    }

}

ini_set('soap.wsdl_cache_enabled', '0');
$server = new SoapServer('wsdl_service.wsdl');
$server->setClass('i5_program_service');
$server->handle();

?>
```



Database access

This chapter provides details on ways to access data stored on i5/OS from PHP applications. We focus on data stored in DB2 for i5/OS and briefly discuss MySQL.

5.1 DB2

IBM DB2 is an advanced relational database management system that adheres to open standards and is capable of managing both large-scale data and high-speed transactions. DB2 data server provides a high-performance and robust environment for all types and sizes of databases and applications that run on it.

DB2 data server offers database solutions that run on all platforms, including AIX, Sun, HP-UX, Linux, Windows, i5/OS, and z/OS® on both 32-bit and 64-bit environments. The DB2 family is a consistent set of relational database management systems (RDBMS) that utilizes shared technologies and a common application programming interface.

5.1.1 DB2 access options

The DB2 for PHP extension communicates with DB2 using the Call Level Interface (CLI). The interfaces are PHP extensions, which is part of PECL (PHP Extension Community Library) written in C compiled with DB2 libraries.

To use the DB2 extension, you must specify the relational database name when connecting to DB2 for i5/OS, not the IP system name/IP address.

Use the following commands to display/change RDB directory entries:

DSPRDBDIRE - Display RDB Directory Entries

With the proper authorization you can change the name with the following command:

WRKRDBDIRE - Work with Relational Database Directory Entries

Tip: Some valuable information about accessing DB2 for i5/OS:

- a. `db2_connect` can specify an RDB Directory Entry that is really a remote one (one i5 has an RDB directory entry that points to another i5's IP address or domain name), and thus PHP can access data on another System i, even one that does not have PHP or Zend installed. This remote method does use DRDA®.
- b. If you want to connect to the System i on which Zend is installed, leave the system name blank.

There are three main extensions of PHP that you can use to write applications with DB2:

- ▶ IBM_DB2
- ▶ PDO_ODBC
- ▶ Unified ODBC

Important: The IBM_DB2 and Unified ODBC extensions are already shipped and installed with Zend Core for i5/OS. There is no need to download any code or perform any configuration changes to begin using this support.

IBM recommends using IBM_DB2 or PDO_ODBC to get the best results out of the IBM DB2 database.

Restriction: PDO_ODBC is not currently supported with DB2 for i5/OS.

IBM_DB2

A good source of documentation about these APIs provided by the `ibm_db2` extension is available at:

<http://www.zend.com/manual/ref.ibm-db2.php>

Documentation about the `ibm_db2` extension for the Linux, UNIX®, and Windows platforms can be found at the following address. Note that even though the information is not geared toward i5/OS, much of it is accurate and applicable:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp?topic=/com.ibm.db2.udb.apdv.php.doc/doc/t0023484.htm>

Also, see the IBM Redbook *Developing PHP Applications for IBM Data Servers*, SG24-7218, for excellent information about the `ibm_db2` extension, including much valuable technical detail that will not be repeated here. The book is available at:

<http://www.redbooks.ibm.com/abstracts/sg247218.html?Open>

The `ibm_db2` extension of PHP provides the interface to connect from PHP to IBM DB2, Cloudscape, and Apache Derby databases. This extension provides mechanisms for application developers to issue SQL queries, work with large objects, call stored procedures, use persistent connections, and use prepared SQL statements. It also works on PHP releases below Version 5. Unlike `PDO_ODBC`, `ibm_db2` is based on traditional procedural programming and performs better when compared to Unified ODBC functions. `ibm_db2` provides built-in functions for getting details about the DB2 database server and client by querying system tables, which provide lots of information about the DB2 database management system.

Tip: If you understand the C programming language and at any time want to see how the driver actually works, you can view the source code at the following URL:

http://viewcvs.php.net/viewvc.cgi/pecl/ibm_db2/

i5/OS considerations

If you create a connection to DB2 for i5/OS (with `db2_connect`) without specifying a user ID or password, the database is accessed within the same process that the PHP script is executing. Often (but not always) this is one of several jobs in the “ZEND” subsystem that looks something like the jobs shown in Example 5-1.

Example 5-1 Jobs under which PHP scripts execute

```

                                Work with Active Jobs                                RCHXXXXX
                                12/15/06 15:20:03
CPU %:      3.4      Elapsed time:  00:00:00      Active jobs:   311

Type options, press Enter.
  2=Change  3=Hold  4=End  5=Work with  6=Release  7=Display message
  8=Work with spooled files 13=Disconnect ...

Current
Opt  Subsystem/Job  User      Type  CPU %  Function      Status
    ZENDCOREAP     NOBODY    BCI    .0     PGM-httpd     TIMW
    ZENDCOREAP     NOBODY    BCI    .0     PGM-httpd     TIMW
    ZENDCOREAP     NOBODY    BCI    .0     PGM-httpd     TIMW
    ZENDCOREAP     NOBODY    BCI    .0     PGM-httpd     TIMW

Parameters or command
===>
```

More...

F3=Exit F5=Refresh F7=Find F10=Restart statistics
F11=Display elapsed data F12=Cancel F23=More options F24=More keys

If you create the connection by passing in a user ID and password, then so-called “server mode” is used. In this case, the database is accessed through a QSQSRVR job.

See 7.1.4, “User profiles” on page 89 for more details on user profiles and PHP on i5/OS.

Attention: Column names in the SQL statement must be in upper-case unless you are using the option of DB2_ATTR_CASE to DB2_CASE_LOWER in the db2_connect() function.

List of APIs

The following is the list of APIs supported by the ibm_db2 extension. They are placed into common groups for convenience.

Table 5-1 *ibm_db2 APIs*

| API | Description |
|-------------------------------|--|
| Server/connection APIs | |
| db2_bind_param | Binds a PHP variable to an SQL statement parameter. |
| db2_client_info | Returns an object with properties that describe the DB2 database client. |
| db2_close | Closes a database connection. |
| db2_connect | Returns a connection to a database. |
| db2_cursor_type | Returns the cursor type used by a statement resource. |
| db2_exec | Executes an SQL statement directly. |
| db2_execute | Executes a prepared SQL statement. |
| db2_prepare | Prepares an SQL statement to be executed. |
| db2_pconnect | Returns a persistent connection to a database. |
| db2_server_info | Returns an object with properties that describe the DB2 database server. |
| db2_statistics | Returns a result set listing the index and statistics for a table. |
| Results | |
| db2_free_result | Frees resources associated with a result set. |
| db2_next_result | Requests the next result set from a stored procedure. |
| db2_result | Returns a single column from a row in the result set. |
| Commit/rollback | |
| db2_autocommit | Returns or sets the AUTOCOMMIT state for a database connection. |
| db2_commit | Commits a transaction. |
| db2_rollback | Rolls back a transaction. |

| API | Description |
|--------------------------|---|
| Fetch | |
| db2_fetch_array | Returns an array, indexed by column position, representing a row in a result set. |
| db2_fetch_assoc | Returns an array, indexed by column name, representing a row in a result set. |
| db2_fetch_both | Returns an array, indexed by both column name and position, representing a row in a result set. |
| db2_fetch_object | Returns an object with properties representing columns in the fetched row. |
| db2_fetch_row | Sets the result pointer to the next row or requested row. |
| Field information | |
| db2_field_display_size | Returns the maximum number of bytes required to display a column. |
| db2_field_name | Returns the name of the column in the result set. |
| db2_field_num | Returns the position of the named column in a result set. |
| db2_field_precision | Returns the precision of the indicated column in a result set. |
| db2_field_scale | Returns the scale of the indicated column in a result set. |
| db2_field_type | Returns the data type of the indicated column in a result set. |
| db2_field_width | Returns the width of the current value of the indicated column in a result set. |
| Key information | |
| db2_foreign_keys | Returns a result set listing the foreign keys for a table. |
| db2_primary_keys | Returns a result set listing primary keys for a table. |
| Statement | |
| db2_free_stmt | Frees resources associated with the indicated statement resource. |
| Errors | |
| db2_conn_error | Returns a string containing the SQLSTATE returned by the last connection attempt. |
| db2_conn_errormsg | Returns the last connection error message and SQLCODE value. |
| db2_stmt_error | Returns a string containing the SQLSTATE returned by an SQL statement. |
| db2_stmt_errormsg | Returns a string containing the last SQL statement error message. |

| API | Description |
|--------------------------|--|
| Column/procedure | |
| db2_column_privileges | Returns a result set listing the columns and associated privileges for a table. |
| db2_columns | Returns a result set listing the columns and associated metadata for a table. |
| db2_procedure_columns | Returns a result set listing stored procedure parameters. |
| db2_procedures | Returns a result set listing the stored procedures registered in a database. |
| db2_special_columns | Returns a result set listing the unique row identifier columns for a table. |
| Table information | |
| db2_num_fields | Returns the number of fields contained in a result set. |
| db2_num_rows | Returns the number of rows affected by an SQL statement. |
| db2_table_privileges | Returns a result set listing the tables and associated privileges in a database. |
| db2_tables | Returns a result set listing the tables and associated metadata in a database. |

i5/OS-specific functionality

There are a number of configurable attributes that can be set on the `db2_connect()` call. The entire list is contained in Table 5-2. Note that these attributes are specific to i5/OS, as you can tell from the “i5_” prefix on the attribute keys.

An example of how you would pass in these attributes is shown in the following code snippet:

```
$options = array("i5_lib"=>"mylibrary", "i5_commit"=>DB2_I5_TXN_READ_UNCOMMITTED);
$i5 = db2_connect($i5localhost, $i5user, $i5password, $options);
```

Table 5-2 Configurable attributes on db2_connect() call

| Key | Value |
|-----------|---|
| i5_lib | <p>A character value that indicates the default library that will be used for resolving unqualified file references. This is not valid if the connection is using system naming mode.</p> <p>Note: if this attribute is not passed in you must fully qualify your table name with the DB2 for i5/OS schema (library): for example, “myschema.mytable.”</p> |
| i5_naming | <p>DB2_I5_NAMING_ON - Turns on DB2 UDB CLI System i naming mode. Files are qualified using the slash (/) delimiter. Unqualified files are resolved using the library list for the job.</p> <p>DB2_I5_NAMING_OFF - Turns off DB2 UDB CLI default naming mode, which is SQL naming. Files are qualified using the period (.) delimiter. Unqualified files are resolved using either the default library or the current user ID.</p> |

| Key | Value |
|--------------|---|
| i5_dbc_alloc | <p>DB2_I5_DBCS_ALLOC_ON - Turns on DB2 6X allocation scheme for DBCS translation column size growth.</p> <p>DB2_I5_DBCS_ALLOC_OFF - Turns off DB2 6X allocation scheme for DBCS translation column size growth.</p> |
| i5_commit | <p>The SQL_ATTR_COMMIT attribute should be set before the SQLConnect(). If the value is changed after the connection has been established, and the connection is to a remote data source, the change does not take effect until the next successful SQLConnect() for the connection handle.</p> <p>DB2_I5_TXN_NO_COMMIT - Commitment control is not used.</p> <p>DB2_I5_TXN_READ_UNCOMMITTED - Dirty reads, nonrepeatable reads, and phantoms are possible.</p> <p>DB2_I5_TXN_READ_COMMITTED - Dirty reads are not possible. Nonrepeatable reads, and phantoms are possible.</p> <p>DB2_I5_TXN_REPEATABLE_READ - Dirty reads and nonrepeatable reads are not possible. Phantoms are possible.</p> <p>DB2_I5_TXN_SERIALIZABLE - Transactions are serializable. Dirty reads, non-repeatable reads, and phantoms are not possible.</p> |
| i5_date_fmt | <p>DB2_I5_FMT_ISO - The International Organization for Standardization (ISO) date format yyyy-mm-dd is used. This is the default.</p> <p>DB2_I5_FMT_USA - The United States date format mm/dd/yyyy is used.</p> <p>DB2_I5_FMT_EUR - The European date format dd.mm.yyyy is used.</p> <p>DB2_I5_FMT_JIS - The Japanese Industrial Standard date format yyyy-mm-dd is used.</p> <p>DB2_I5_FMT_MDY - The date format mm/dd/yyyy is used.</p> <p>DB2_I5_FMT_DMY - The date format dd/mm/yyyy is used.</p> <p>DB2_I5_FMT_YMD - The date format yy/mm/dd is used.</p> <p>DB2_I5_FMT_JUL - The Julian date format yy/ddd is used.</p> <p>DB2_I5_FMT_JOB - The job default is used.</p> |

| Key | Value |
|----------------|---|
| i5_date_sep | <p>DB2_I5_SEP_SLASH - A slash (/) is used as the date separator. This is the default.</p> <p>DB2_I5_SEP_DASH - A dash (-) is used as the date separator.</p> <p>DB2_I5_SEP_PERIOD - A period (.) is used as the date separator.</p> <p>DB2_I5_SEP_COMMA - A comma (,) is used as the date separator.</p> <p>DB2_I5_SEP_BLANK - A blank is used as the date separator.</p> <p>DB2_I5_SEP_JOB - The job default is used.</p> |
| i5_time_fmt | <p>DB2_I5_FMT_ISO - The International Organization for Standardization (ISO) time format hh.mm.ss is used. This is the default.</p> <p>DB2_I5_FMT_USA - The United States time format hh:mmxx is used, where xx is AM or PM.</p> <p>DB2_I5_FMT_EUR - The European time format hh.mm.ss is used.</p> <p>DB2_I5_FMT_JIS - The Japanese Industrial Standard time format hh:mm:ss is used.</p> <p>DB2_I5_FMT_HMS - The hh:mm:ss format is used.</p> |
| i5_time_sep | <p>DB2_I5_SEP_COLON - A colon (:) is used as the time separator. This is the default.</p> <p>DB2_I5_SEP_PERIOD - A period (.) is used as the time separator.</p> <p>DB2_I5_SEP_COMMA - A comma (,) is used as the time separator.</p> <p>DB2_I5_SEP_BLANK - A blank is used as the time separator.</p> <p>DB2_I5_SEP_JOB - The job default is used.</p> |
| i5_decimal_sep | <p>DB2_I5_SEP_PERIOD - A period (.) is used as the decimal separator. This is the default.</p> <p>DB2_I5_SEP_COMMA - A comma (,) is used as the date separator.</p> <p>DB2_I5_SEP_JOB - The job default is used.</p> |

| Key | Value |
|-------------------|--|
| i5_query_optimize | <p>DB2_FIRST_IO - All queries are optimized with the goal of returning the first page of output as fast as possible. This goal works well when the output is controlled by a user who is most likely to cancel the query after viewing the first page of output data. Queries coded with an OPTIMIZE FOR nnn ROWS clause honor the goal specified by the clause.</p> <p>DB2_ALL_IO - All queries are optimized with the goal of running the entire query to completion in the shortest amount of elapsed time. This is a good option when the output of a query is being written to a file or report, or the interface is queuing the output data. Queries coded with an OPTIMIZE FOR nnn ROWS clause honor the goal specified by the clause. This is the default.</p> |
| i5_fetch_only | <p>DB2_I5_FETCH_ON - Cursors are read-only and cannot be used for positioned updates or deletes. This is the default unless SQL_ATTR_FOR_FETCH_ONLY environment has been set to SQL_FALSE.</p> <p>DB2_I5_FETCH_OFF - Cursors can be used for positioned updates and deletes.</p> |

Code examples

Some simple code examples follow. These are snippets, and not full programs.

Example 5-2 shows how to create a prepared statement, bind a parameter, execute it, and then iterate through the result set.

Example 5-2 Use of prepared statement

```
...
$options = array("i5_lib"=>"QIWS");
$i5 = db2_connect($db, $user, $password, $options);

if (!$i5) :
die ("Problems connecting to the database.");
endif;

/* Construct the SQL statement, using CDTLMT as the search substring */
$sql = "select * from QCUSTCDT where CDTLMT > ?";

/* Prepare, bind and execute the DB2 SQL statement */
$stmt= db2_prepare($i5, $sql);
$lower_limit = 1000;
db2_bind_param($stmt, 1, "lower_limit", DB2_PARAM_IN);
db2_execute($stmt);
...
/* Execute the statement again with a new input parameter */
$upper_limit = 99999;
db2_bind_param($stmt, 1, "upper_limit", DB2_PARAM_IN);
db2_execute($stmt);
```

Example 5-3 on page 68 shows how to use the db2_procedures API to get a list of stored procedures available in i5/OS.

Example 5-3 Get list of stored procedures

```
...
$procName = '%';
$schemaName = '%';

$procs = db2_procedures($conn, '', $schemaName, $procName);
if (!$procs){
    print ('Error in db2_procedures() call');
} else {
    print '<h1>Procedures found in schema ' . $schemaName . ' on System i</h1>';
    print '<h2>Using db2_procedures and db2_fetch_array</h2>';
    print '<br><table border=1 cellpadding=5 cellspacing=5>';
    print '<tr><td
align=center>procedure_cat<td>procedure_schem<td>procedure_name<td>num_input_param
s<td>num_output_params<td>num_result_sets<td>remarks</td></tr>';
    while ($row = db2_fetch_array($procs)) {
        if (!$row=="") {
            $procedure_cat = $row[0];
            $procedure_schem = $row[1];
            $procedure_name = $row[2];
            $num_input_params = $row[3];
            $num_output_params = $row[4];
            $num_result_sets = $row[5];
            $remarks = $row[6];
            print '<tr><td align=center>' . $procedure_cat . '<td>' . $procedure_schem.
            '<td>' . $procedure_name . '<td>' . $num_input_params . '<td>' . $num_output_params . '<td>' .
            $num_result_sets . '<td>' . $remarks . '</td></tr>';
        }
    }
    print '</table><br>';
}
...
```

PDO_ODBC

PDO (PHP Data Objects) is an object-oriented, standards-based data access method in PHP, where you can use the same methodology to query the database and fetch data from the supported databases. The PDO_ODBC extension is the implementation of PDO specification. When compiled with DB2 libraries, you can use it to access DB2, Cloudscape, and Apache Derby databases. This extension provides a mechanism to connect to both local cataloged and non-cataloged databases. For a local cataloged database, PDO_ODBC obtains the database server details from the client machine. For a non-local cataloged database, the full details of the remote database are specified in the connection URL. PDO_ODBC also provides access to advanced features of DB2, such as persistent connections, prepared SQL statements, large objects, and stored procedures. It provides better performance compared to Unified ODBC functions.

As mentioned previously, PDO_ODBC is not currently supported with DB2 for i5/OS.

Unified ODBC

Unified ODBC was the only method for PHP to talk with DB2 before the ibm_db2 extension was released. As with the ibm_db2 extension, Unified ODBC interacts with DB2 using native CLI calls. It uses the same PHP methods to interact with different databases even if the underlying mechanism is different; thus, ODBC offers greater database independence than

extensions designed for a specific DBMS. However, this API cannot be used to call stored procedures in DB2. In addition, you will achieve better performance by using the `ibm_db2` extensions, because they are optimized for DB2 access.

Example 5-4 illustrates a sample of Unified ODBC usage.

Example 5-4 Unified ODBC and DB2 for i5/OS

```
<HTML>
<?php
# connect to rdb directory entry called 'mysystem' ...
# ...using 'kost' as username and 'newpass' as password
$dbh=odbc_connect('mysystem','kost','newpass');

# check if connected
if (!$dbh) {
    die ('Connection Failed: ' . $dbh);
}

# execute the query
$sql="SELECT Host,User FROM schema.user";
$res=odbc_exec($dbh,$sql);
if (!$res) {
    die ("Error in SQL: $sql");
}

# output the results
while (odbc_fetch_row($res))
{
    print odbc_result($res,"Host").':';
    print odbc_result($res,"User").'<br />';
}

# close the connection
odbc_close($dbh);
?>
</HTML>
```

To read more about the Unified ODBC extension and its features, refer to:

<http://www.php.net/manual/en/ref.uodbc.php>

5.1.2 Performance considerations

In general, we recommend that you use the `ibm_db2` driver to access DB2 data whenever possible. It provides the best overall performance, but is not as database-neutral as either `PDO_ODBC` or `Unified ODBC`.

Many of the performance recommendations are similar to those for other programming languages that access DB2 for i5/OS. For example, most of the considerations a developer would keep in mind when using `JDBC™` support in Java would apply to PHP.

Prepare once, execute many

The most efficient way to avoid full database path opens is to employ the so-called “prepare once, execute many” programming paradigm. Ideally, an SQL statement that is executed more than once should be prepared just once, and then reused for consecutive executions.

For an example of using a prepared statement see Example 5-2 on page 67.

Connect as NOBODY user profile

You will realize greater performance by connecting to DB2 for i5/OS as NOBODY rather than as a specific user. This enables the database access to occur within the same i5/OS job as the PHP script is executing. You can do this by passing in empty strings ("") for the user ID and password on the db2_connect call.

Running under a specific user profile causes database access to occur via a QSQSRVR job, which adds a degree of overhead.

See "i5/OS considerations" on page 61 for more details.

Transaction isolation levels

Because the isolation level determines how data is locked and isolated from other processes while the data is being accessed, you should select an isolation level that balances the requirements of concurrency and data integrity. The isolation level that you specify is in effect for the duration of the unit of work.

DB2_I5_TXN_SERIALIZABLE is the most restrictive and protected transaction isolation level, and it incurs significant overhead. Many workloads do not require this level of isolation protection. A given application might never update the underlying data or be run with other concurrent updaters. In that case, the application would not have to be concerned with dirty, non-repeatable, or phantom reads problems. (Refer to the DB2 product documentation for more details.) DB2_I5_TXN_READ_UNCOMMITTED would probably suffice in such cases. Review your isolation level requirements and adjust them appropriately to increase performance.

Persistent connections

Persistent connections can dramatically improve overall performance, but must be used with caution.

The main advantage of using a persistent connection is that it avoids much of the initialization and teardown normally associated with getting a connection to the database. When db2_close() is called against a persistent connection, the call always returns TRUE, but the underlying DB2 client connection remains open and waiting to serve the next matching db2_pconnect() request.

One main area of concern with persistent connections is in the area of commitment control. You should use persistent connections only when you leave automatic commitment control ("autocommit") turned on. Mixing managed commitment control and persistent connections can result in unknown transaction states if errors occur. Consider the following scenario:

1. Request A gets a persistent connection, turns off autocommit, and performs various database update (insert/update/delete) operations.
2. Request A encounters an error, and neglects to either commit or roll back the transaction. The db2_close() operation is called. Note that not all desired updates were performed.
3. Request B requests a persistent connection, and is given the same one just used by Request A. It performs several database updates, and commits its changes.
4. The incomplete database changes performed by Request A were also committed when Request B committed its changes. Thus, we now have a database with inconsistent and incomplete transactions.

5.1.3 Troubleshooting

We know of no tracing functionality built in to the `ibm_db2` driver. A few of the places you should look when problems are occurring include:

- ▶ i5/OS job logs, message queues, and so forth
- ▶ PHP log files (`php_error_log` is probably the best place to look)
- ▶ Application-generated messages (either sent to the browser or put in a separate log file)

An excellent source of debug and troubleshooting information for DB2 for i5/OS not specifically related to PHP can be found in the IBM Redbook *SQL Performance Diagnosis on IBM DB2 Universal Database for iSeries*, SG24-6654. It can be found at:

<http://www.redbooks.ibm.com/Redbooks.nsf/RedbookAbstracts/sg246654.html?Open>

5.2 MySQL

MySQL is a common Open Source SQL database management system. It is developed and distributed by MySQL AB (<http://www.mysql.com>), a company that builds its business by providing services around MySQL.

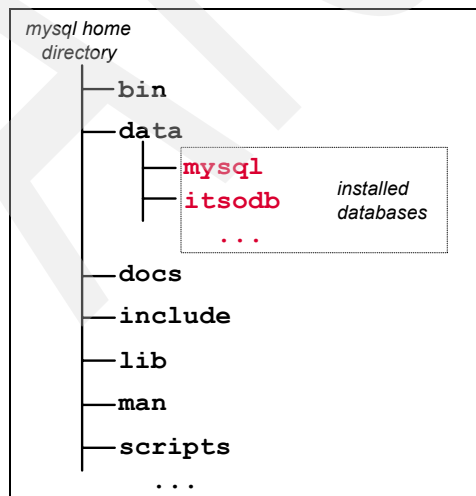
MySQL first was developed for UNIX and Linux applications. It became popular when Internet service providers (ISP) discovered that MySQL could be offered free of charge to their Internet customers, and was able to provide all the storage and retrieval functionality a dynamic Web application needs. It was also advantageous that ISPs mostly use Linux or UNIX together with Apache as their favorite Web server environment. However, MySQL is in use as an integrated database in many applications on almost every operating system.

5.2.1 MySQL architecture

MySQL is a client-server architecture based on TCP/IP. The MySQL server waits for connections on a specified port and responds to SQL statements from a client.

5.2.2 MySQL directory structure

MySQL can be installed as one or more databases within one server. For each database, MySQL creates a directory that holds the data, indexes, and other related information.



The figure at left shows the directory structure.

In our example, the installed databases are *mysql*, which by default holds the security information, and our sample database *itsodb*. For each database there is a directory that contains three files per table. Files with the “MYD” extension contain the table data. Files with the “MYI” extension contains the table’s indexes. Files with the `frm` extension contain the table’s structure definition known as the *schema*.

5.2.3 Installation and configuration

These instructions will help you understand how to download, install, and run MySQL in the i5/OS PASE environment.

Restriction: When this document was written, MySQL was not officially supported under i5/OS.

1. Create the user profile `mysql` by issuing the following command in i5/OS:

```
CRTUSRPRF USRPRF(MYSQL) STATUS(*DISABLED) LCLPDMGT(*NO) TEXT('MySQL user id')
```

Note that MySQL profile is a disabled ID, so it cannot be used to sign on to the system but can be used by MySQL.

2. Go to <http://www.mysql.com> and download the AIX 5.2 (POWER™ 64-bit) version of MySQL 5.0 to your workstation. At the time of this writing, `mysql-max-5.0.24a-aix5.2-powerpc-64bit.tar.gz` was the name of the file for Max version of MySQL, and `mysql-standard-5.0.24a-aix5.2-powerpc-64bit.tar.gz` was the name of the file for Standard version of MySQL. In our case, we used the Max version.
3. Extract the file on your workstation in order to get the TAR archive as illustrated in Figure 5-1. We used a GPL-licensed tool called “7-zip,” which is free to download and use. You can download it at the following Web address:

<http://www.7-zip.org>

Note: WinRAR does not work for extracting .gz archives because it automatically opens the .tar archive as well.

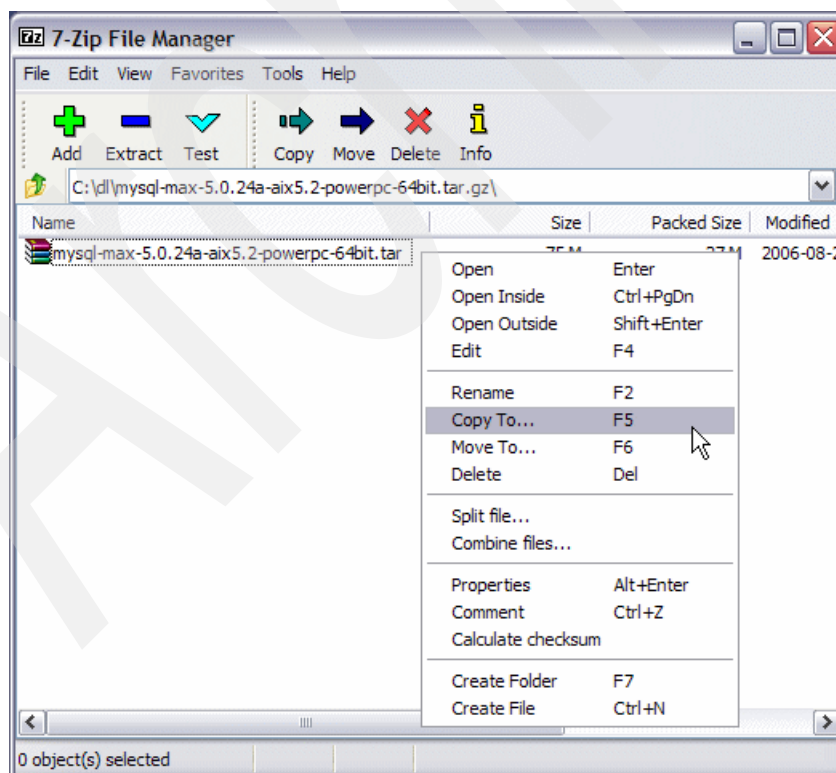


Figure 5-1 Extracting TAR archive using 7-Zip

4. When you have the .tar file, transfer it to the /usr/local directory on i5/OS (probably using FTP).
5. Log on to a 5250 session as user QSECOFR.
6. Go to the PASE for i5/OS environment by issuing the following command:
`call qp2term`
7. Change to directory /usr/local by issuing the command:
`cd /usr/local`
8. Untar the file with command `tar xvf mysql-max-5.0.24a-aix5.2-powerpc-64bit.tar` as illustrated in Figure 5-2.

```

/Q0penSys/usr/bin/-sh

$
> cd /usr/local
$
> ls
Zend
mysql-max-5.0.24a-aix5.2-powerpc-64bit.tar
$
> tar xvf mysql-max-5.0.24a-aix5.2-powerpc-64bit.tar
x mysql-max-5.0.24a-aix5.2-powerpc-64bit
x mysql-max-5.0.24a-aix5.2-powerpc-64bit/bin
x mysql-max-5.0.24a-aix5.2-powerpc-64bit/bin/comp_err, 1379704 bytes, 2695
me
dia blocks.
x mysql-max-5.0.24a-aix5.2-powerpc-64bit/bin/replace, 1377369 bytes, 2691
med
ia blocks.

===>

F3=Exit      F6=Print     F9=Retrieve   F11=Truncate/Wrap
F13=Clear    F17=Top      F18=Bottom    F21=CL command entry

```

Figure 5-2 Untarring the MySQL TAR file

9. Create a symbolic link called mysql to the new directory (which in our case is called "mysql-max-5.0.24a-aix5.2-powerpc-64bit") for easier directory navigation with the following command:
`ln -s mysql-max-5.0.24a-aix5.2-powerpc-64bit mysql`
10. Create the MySQL system table by issuing the following command:
`scripts/mysql_install_db --user=mysql`
11. Now you can start the MySQL server with the following command (as shown in Figure 5-3 on page 74):
`bin/mysqld_safe --user=mysql &`

```

> bin/mysqld_safe --user=mysql &
[1] 114232
$ Starting mysqld daemon with databases from /usr/local/mysql/data

===>

F3=Exit      F6=Print      F9=Retrieve    F11=Truncate/Wrap
F13=Clear     F17=Top       F18=Bottom     F21=CL command entry

```

Figure 5-3 Starting MySQL

12. To verify that installation was successful, issue the following command:

```
bin/mysqlcheck -u root mysql
```

You should get output similar to that shown in Figure 5-4.

```

> bin/mysqlcheck -u root mysql
mysql.columns_priv      OK
mysql.db                OK
mysql.func              OK
mysql.help_category     OK
mysql.help_keyword      OK
mysql.help_relation     OK
mysql.help_topic        OK
mysql.host              OK
mysql.proc              OK
mysql.procs_priv        OK

===>

F3=Exit      F6=Print      F9=Retrieve    F11=Truncate/Wrap
F13=Clear     F17=Top       F18=Bottom     F21=CL command entry

```

Figure 5-4 mysqlcheck output

If you get error messages similar to those shown in Figure 5-5 it probably means that the MySQL server is not running.

```

> bin/mysqlcheck -u root mysql
bin/mysqlcheck: Got error: 2002: Can't connect to local MySQL server through
socket '/tmp/mysql.sock' (2) when trying to connect
$

===>

F3=Exit      F6=Print      F9=Retrieve    F11=Truncate/Wrap
F13=Clear     F17=Top       F18=Bottom     F21=CL command entry

```

Figure 5-5 mysqlcheck failed output

13. In order to utilize MySQL with PHP in Zend Core you must enable the MySQL Extensions in the Zend Core control center. You can do that also by uncommenting (removing the “#” character) from the following lines in /usr/local/Zend/Core/etc/php.ini file:

```
extension=mysql.so  
extension=mysqli.so
```

14. To stop the MySQL server, run the following command (from the /usr/local/mysql directory):

```
bin/mysqladmin -u root shutdown
```

Upon successful shutdown, there will be no messages.

15. For better security we recommend that you set a password for the root MySQL user. You can do that by typing the following command (from the /usr/local/mysql directory):

```
bin/mysqladmin -u root password newpass
```

This command will change the password for user “root” to “newpass.”

Note: After you set a password for the root MySQL user, you must specify the password while working with the database (on behalf of user root). For example, to shut down the server, specify following command:

```
bin/mysqladmin -u root -pnewpass shutdown
```

16. We recommend that you read the UNIX Post-Installation Procedures in the MySQL 5.0 Reference Manual at the following address:

<http://dev.mysql.com/doc/refman/5.0/en/unix-post-installation.html>

5.2.4 Access options

MySQL can be used with the following extensions of PHP:

- ▶ MySQL/MySQLi
- ▶ PDO_MYSQL
- ▶ Unified ODBC

The source code for all the extensions is available for download at the PECL Web site:

<http://pecl.php.net/>

We recommend using the MySQL interface in order to get the best results and performance.

Note: Zend for iOS is shipped with all these extensions. Therefore, there is no need to download them. To use, make sure you enable the desired extension in the Zend Administration Panel and restart the Web server.

MySQL/MySQLi

The mysql and mysqli extensions of PHP provide interfaces to connect from PHP to MySQL databases. These extensions provide mechanisms for application developers to issue SQL queries, work with large objects and use persistent connections. These extensions also work with PHP releases below Version 5. Unlike PDO_ODBC, mysql and mysqli are based on traditional procedural programming and perform better compared to Unified ODBC. The main difference between mysql and mysqli is that mysqli is a more advanced (“improved”) extension of mysql and supports MySQL functions provided by MySQL 4.1 and above.

To read more about the mysql extension and its features, refer to the following Web site:

<http://www.php.net/manual/en/ref.mysql.php>

Example 5-5 shows an example of mysql extension usage.

Example 5-5 mysql extension example

```
<HTML>
<?php
# connect to MySQL DBMS using 'root' as username and 'kost' as password
$dbh = mysql_connect('localhost', 'root', 'kost');

# check if connected
if (!$dbh) {
    die('mysql connect failed: '.mysql_error());
}

# select database and do the query
mysql_select_db("mysql");
$sql='select Host,User from user';
$res = mysql_query($sql);
if (!$res) {
    die('Error: $sql: '.mysql_error());
}

# output the query results
while ($r = mysql_fetch_assoc($res)) {
    print $r['Host'].':';
    print $r['User'].'<br />';
}

# free the query resources
mysql_free_result($res);

# close connection to MySQL DBMS
mysql_close($dbh);
?>

</HTML>
```

An example of mysqli usage is illustrated in Example 5-6. To read more about the mysqli extension and its features, refer to:

<http://www.php.net/manual/en/ref.mysqli.php>

Example 5-6 Procedural syntax of mysqli extension

```
<HTML>
<?php
# connect to MySQL system database called 'mysql' ...
# ...using 'root' as username and 'kost' as password
$dbh = mysqli_connect('localhost', 'root', 'kost', 'mysql');

# check if connected
if (!$dbh) {
    die('mysqli connect failed: '.mysqli_connect_error());
}
```



```

# execute the query
$sql = 'select Host,User from user';
$res = mysqli_query($dbh,$sql);
if (!$res) {
    die("Error: $sql: ".mysqli_error($dbh));
}

# output the query results
while ($r = mysqli_fetch_assoc($res)) {
    print $r['Host'].':';
    print $r['User'].'<br />';
}

# free the query resources
mysqli_free_result($res);

# close connection to MySQL DBMS
mysqli_close($dbh);
?>
</HTML>

```

PDO_MYSQL

PDO (PHP Data Objects) is an object-oriented, standards-based data access method in PHP, where you can use the same methodology to query the database and fetch data from the supported databases. The PDO_ODBC extension is the implementation of PDO specification. It provides better performance compared to Unified ODBC functions.

Unified ODBC

Unified ODBC uses the same PHP methods to interact with different databases even if the underlying mechanism is different. It therefore provides a level of database independence to your code, at the price of less functionality and reduced performance.

The free source code for all of the extensions is available for download at the PECL Web site:

<http://pecl.php.net/>

5.3 Migrating from MySQL to DB2

Although MySQL has evolved into a quite robust and full-function database, it is still significantly behind DB2 for i5/OS in almost every category.

See *Developing PHP Applications for IBM Data Servers*, SG24-7218, for excellent information about how to migrate from MySQL to DB2. Although the book is not specific to i5/OS, it is still a very helpful guide.

The IBM DB2 Migration Toolkit (MTK), for example, is free software available from <http://www.ibm.com/software/data/db2/migration/mtk/>. The DB2 MTK currently supports conversions of MySQL databases to DB2 SQL syntax, but it does not support a DB2 for i5/OS target. An upcoming version of the DB2 MTK supports converting MySQL to DB2 for i5/OS SQL syntax. See “Online resources” on page 119 for a link to the IBM DB2 Migration Toolkit Web page for the latest version information. The DB2 Migration Toolkit is best suited to convert long SQL scripts. This tool is not designed for directly converting the SQL statements

embedded in PHP code. However, you can copy individual SQL statements into the DB2 MTK Translator utility to ease conversion of complex MySQL SQL statements.

Another helpful resource is the “DB2 for i5/OS - Porting Information” Web site at:

<http://www.ibm.com/servers/enable/site/db2/porting.html>

Find a document highlighting some of the differences between DB2 for i5/OS and MySQL at:

<http://www.ibm.com/developerworks/systems/library/es-path2php/>

5.4 When to choose MySQL or DB2

Because DB2 for i5/OS is included at no extra cost with the i5/OS operating system, the most compelling reason to use MySQL on i5/OS is when you are porting an application from another system to System i, and that application is closely tied to MySQL. Even then you might want to consider modifying the application to take advantage of DB2.

Java Bridge support

This chapter provides details about the Java Bridge support found in the Zend Platform for i5/OS. It describes the functionality of the Java Bridge, then provides some examples of how it could be used as a method for accessing i5/OS resources via the IBM Toolbox for Java.

6.1 Overview

The Java Bridge support makes it possible to call Java programs from within PHP. This enables you to directly access existing Java applications you might have written already, or simply i5/OS resources (data queues, printers, and so on) by interfacing with the IBM Toolbox for Java. By doing so you avoid the development time and code overhead of using a technology such as Web services to expose or consume existing functionality.

More information about the Java Bridge support is available at:

http://www.zend.com/products/zend_platform/in_depth/php_java_integration

Additional information about the IBM Toolbox for Java can be found at:

<http://www.ibm.com/eserver/iserries/toolbox/>

6.2 Java Bridge configuration

To use the Java Bridge functionality, some additional configuration is required.

Important: The Java Bridge functionality is available only if the Zend Platform for i5/OS product is installed. It is not contained in Zend Core for i5/OS.

6.2.1 Main configuration file: javamw.rc

The main configuration file for the Java Bridge support is as follows:

`/usr/local/Zend/Platform/bin/javamw.rc`

Restriction: Do not attempt to edit this file using a PC-based editor such as Notepad. This will corrupt the file. We recommend that you use the i5/OS editor (EDTF).

The main change you will likely have to make to this file is to add entries to the Java classpath. For example, we added the IBM Toolbox for Java as follows:

```
...  
CLASSPATH=$BINDIR/javamw.jar:/jt400/jt400.jar:$CLASSPATH  
...
```

6.2.2 Java Bridge status

After you have installed both Zend Core for i5/OS and Zend Platform for i5/OS successfully, you can verify that the support is active by accessing the following URL in a browser:

`http://MYSERVER:89/Zend Platform/server/index.php?frame=java`

Figure 6-1 on page 81 shows an example of the page that opens.

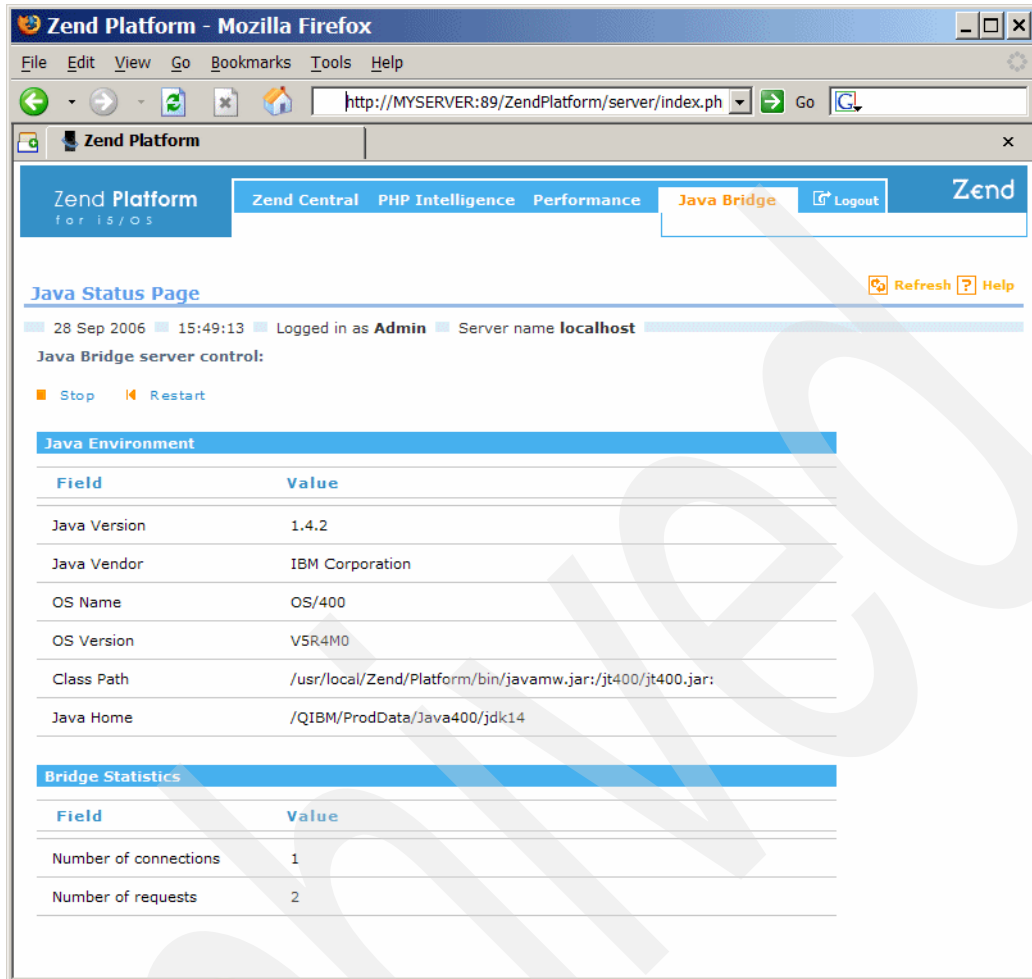


Figure 6-1 Java Bridge status page

From this screen we can see that our classpath change in fact worked.

We can stop and restart the Java Bridge support from this screen. You can also perform these actions from an i5/OS PASE terminal with the following commands (first change into the correct directory with `cd /usr/local/Zend/Platform`).

| | |
|----------------|------------------------------------|
| Start | <code>bin/javamw.rc start</code> |
| Stop | <code>bin/javamw.rc stop</code> |
| Restart | <code>bin/javamw.rc restart</code> |
| Status | <code>bin/javamw.rc status</code> |

6.3 Sample programs

This section contains some simple programs to show how to test and take advantage of the Java Bridge functionality found in Zend Platform for i5/OS.

6.3.1 Simple test program

The program in Example 6-1 on page 82 shows how to get all of the Java system properties and iterate through them, printing all key/value pairs out to the screen.

Example 6-1 Print all Java system properties

```
<html>
<head><title>Java System Properties</title></head>
<?php

// Get java.lang.System instance
$system = new Java("java.lang.System");

// Get the properties.
// In PHP this will be put into an array of key-value pairs
$properties = $system->getProperties();

// Iterate through properties and write out to screen
foreach($properties as $key=>$value) {
    echo "<br>";
    echo $key . " : " . $value;
}

?>

</html>
```

The output from running this PHP program looks similar to Figure 6-2.

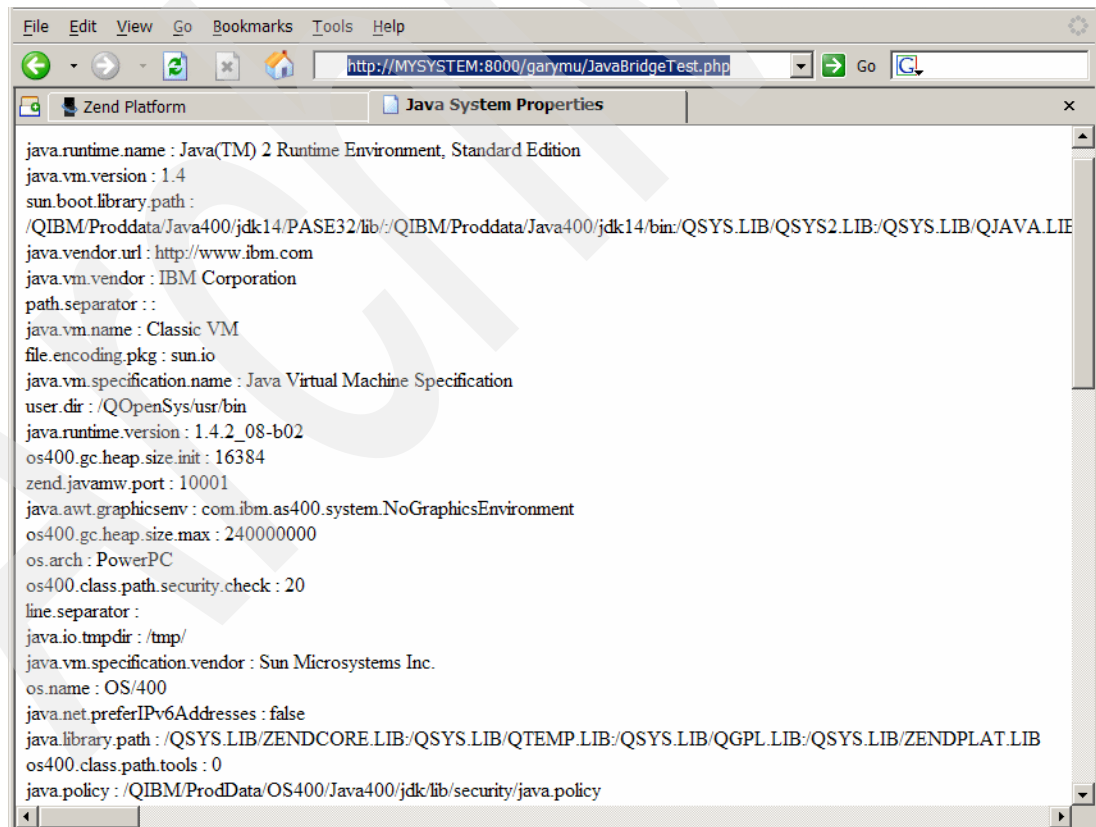


Figure 6-2 Output from running Java system properties program

6.3.2 IBM Toolbox for Java and PHP

This section shows two sample programs that access i5/OS resources via the IBM Toolbox for Java.

Restriction: Only a few Toolbox functions were executed when this chapter was written. Comprehensive testing of all functionality has *not* been done.

For example, at the time this chapter was written the ability to run commands on i5/OS via the `ibm.com®.as400.access.CommandCall` class did not work.

Message queue test program

Example 6-2 shows a simple program to display the contents of an i5/OS message queue. You should first make sure that there are messages in the queue to see output.

Example 6-2 Display all messages in i5/OS message queue

```
<html>
<head><title>Message Queue Test</title></head>
<?php

// Get an AS400 connection object
$as400 = new Java('com.ibm.as400.access.AS400', 'localhost', 'myuserid',
'mypassword');

// Create the MessageQueue object
$mq = new Java('com.ibm.as400.access.MessageQueue', $as400,
'/QSYS.LIB/QUSRSYS.LIB/GARYMU.MSGQ');

// Get the number of messages in the queue and print it out
$numMessages = $mq->getLength();
echo '<br>Number of messages: ' . $numMessages ;

// If we have messages, print each; otherwise indicate no messages exist
if (0 < $numMessages) {
    $messages = $mq->getMessages(-1, 0);

    foreach($messages as $value) {
        echo '<br>Message = ' . $value;
    }
} else {
    echo '<br>No messages in queue.';
}

// Disconnect all services
$as400->disconnectAllServices();

?>

</html>
```

The output from this program looks similar to Figure 6-3 on page 84 (if there are messages in the queue).

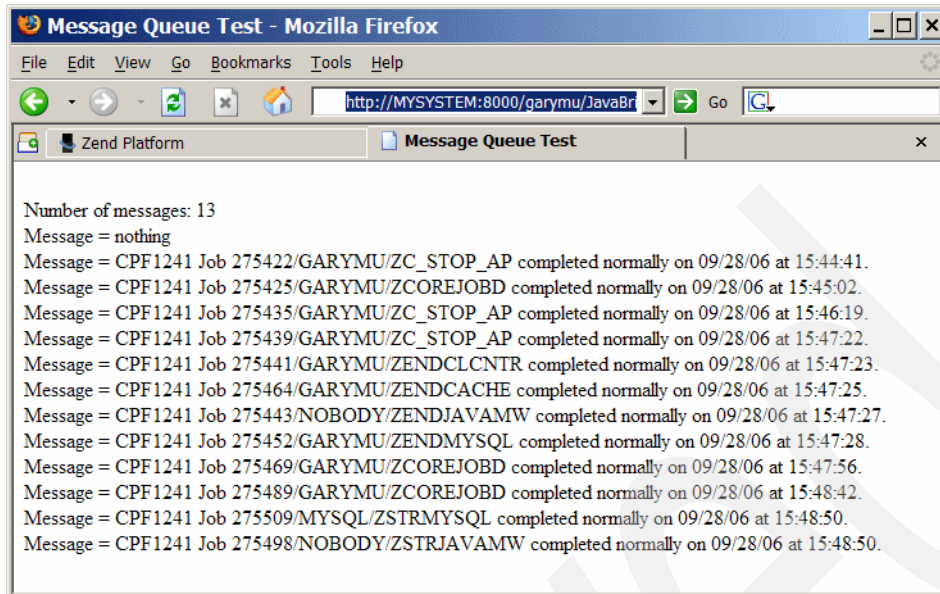


Figure 6-3 Output from running Java Toolbox message queue program

Data queue test program

Example 6-3 shows how to post and peek entries from an i5/OS data queue. We test for the existence of the data queue - if it doesn't exist, we create it. If it does, we clear its contents.

Example 6-3 Manipulate i5/OS data queue

```
<html>
<head><title>Data Queue Test</title></head>
<?php

// Get an AS400 connection object
$as400 = new Java('com.ibm.as400.access.AS400', 'localhost', 'myuserid',
'mypassword');

// Get a data queue object
$dq = new Java('com.ibm.as400.access.DataQueue', $as400,
"/QSYS.LIB/GARYMU.LIB/GARYDATAQ.DTAQ");

// Create a data queue object in i5/OS if it doesn't exist; otherwise clear its
contents
if (!$dq->exists()) {
    $dq->create();
} else {
    $dq->clear();
}

// Write a string to the data queue containing the current time
$dq->write("Current date and time: " . date("F j, Y, g:i a"));

// Peek the data - this does not remove the entry from the data queue
$entry = $dq->peek();
$data = $entry->getString();

// Write out the data to the browser
```



```

echo '<br>Data queue data: ' . $data;

// Delete the data queue
$dq->delete();

// Disconnect all services
$as400->disconnectAllServices();
?>

</html>

```

Figure 6-4 shows the output from the program in Example 6-3 on page 84.

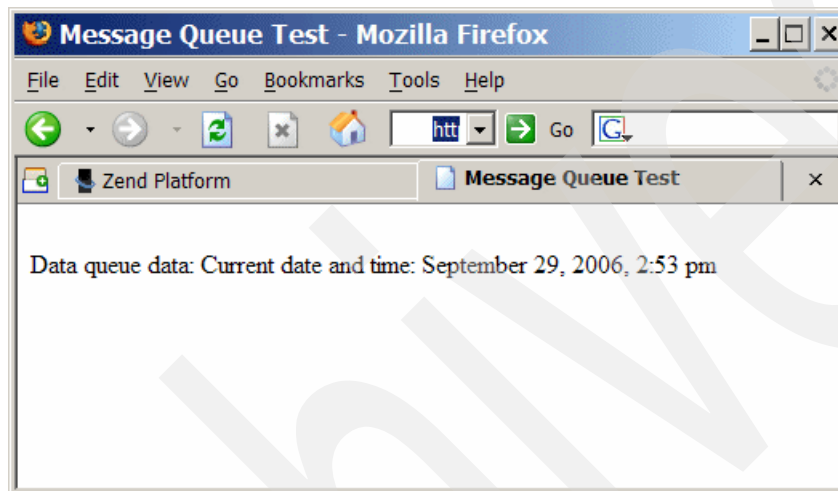


Figure 6-4 Output from running Java Toolbox data queue program

6.4 Troubleshooting tips

This section contains some simple troubleshooting tips that make it easier to diagnose problems with the Java Bridge.

6.4.1 php_error_log file

The `php_error_log` file is probably the first place to look for information. It is found in the `/usr/local/Zend/Core/logs` directory.

Example 6-4 shows the error that is logged when attempting to instantiate an object using a nonexistent constructor. The last line shows the row of the PHP file that caused the error.

Example 6-4 Sample error in `php_error_log` file

```

[29-Sep-2006 14:41:19] PHP Fatal error: Uncaught exception 'JavaException' with
message 'Java Exception java.langInstantiationException: No matching constructor
found
java.langInstantiationException: No matching constructor found
    at java.lang.Throwable.<init>(Throwable.java:195)
    at java.lang.Exception.<init>(Exception.java:41)
    at java.langInstantiationException.<init>(InstantiationException.java:37)
' in /www/zendcore/htdocs/garymu/JavaBridgeDataQueuePost.php:9

```

```
Stack trace:  
#0 /www/zendcore/htdocs/garymu/JavaBridgeDataQueuePost.php(9): *No  
Class!*->java('com.ibm.as400.a...', Object(com.ibm.as400.access.AS400))  
#1 {main}  
    thrown in /www/zendcore/htdocs/garymu/JavaBridgeDataQueuePost.php on line 9
```

6.4.2 var_dump() function

This built-in PHP function is very helpful in determining what type of variable PHP gets back from a Java method call. For example, the call `$properties = $system->getProperties();` found in Example 6-1 on page 82 returns a PHP Array, even though the actual method in Java returns a Properties object.



Security

This chapter provides details about unique security considerations for PHP on i5/OS.

7.1 Security considerations

Securing your PHP applications on i5/OS requires some changes to the default environment. This section discusses some of the different areas you should scrutinize to ensure that your business applications that are written in PHP are secure.

Of course, there are many aspects to successfully securing data and applications on any platform, including i5/OS. The IBM Redbook *IBM System i Security Guide for IBM i5/OS Version 5 Release 4*, SG24-6668, is an excellent overall guide to help implement security on i5/OS:

<http://www.redbooks.ibm.com/abstracts/sg246668.html?Open>

7.1.1 SSL configuration

SSL (secure sockets layer) helps encrypt the data sent from your Web server to the user's browser. This makes it extremely difficult, if not impossible, for a third party to intercept the data stream and extract valuable information as it passes through the Internet. You must also ensure that proper authentication and authorization are in place to make sure you know who is accessing what parts of your Web server.

The main twist with using SSL together with Zend Core for i5/OS is that the SSL configuration must be done on the Apache server (called "zendcore") running in i5/OS. You cannot configure the "inner" Apache server (which runs in PASE for i5/OS) to do the SSL work.

For detailed information about configuring SSL on i5/OS, see *IBM HTTP Server (powered by Apache): An Integrated Solution for IBM eSeries iSeries Servers*, SG24-6716, available at:

<http://www.redbooks.ibm.com/abstracts/sg246716.html?Open>

7.1.2 Access to directory structure

Several directories are created when Zend Core for i5/OS installs:

- ▶ /usr/local/Zend
- ▶ /www/zendcore

We now discuss some of the important files in these directories that you should consider "locking down."

Configuration files

The main PHP configuration file (php.ini) is located in directory /usr/local/Zend/Core/etc. This file (and all others in this directory) can be edited by any user. You should probably modify the access attributes so that only specific administrators can change the files.

The same is true for the internal Apache server: Its configuration file (httpd.conf) is located in the directory /usr/local/Zend/apache2/conf.

The external Apache server configuration file (httpd.conf) is located in the /www/zendcore/conf directory. By default, *PUBLIC does not have write authority to this file.

PHP source files

By default, Zend Core for i5/OS has its document root as /www/zendcore/htdocs. All users have full access to all files in this directory - thus, any i5/OS user can by default create content that will be served up by the PHP server. Also, users can modify any existing scripts under this directory. It is very important that you change the default security attributes for this directory, and all directories and files contained within it.

7.1.3 Reverse proxy

There are two key reasons for using a proxy HTTP server:

- The first reason is to improve performance.

A proxy server is commonly used to pre-cache static Web pages into memory at the time the server starts. This allows the pages to be sent to the requesting client without having to retrieve the content from the file system.

The second way performance can be improved is by having one proxy server that can forward requests to a number of other HTTP servers who are each configured to serve a given set of Web pages or Web applications. This can help by balancing the load of many requests across a number of internal servers.

- The second feature of a proxy server is to improve security.

By using a proxy server, you can control access outside your firewall and Demilitarized Zone (DMZ). This enables you to keep a production-level server inside your secured internal network. It also lets you hide internal servers completely from the requesting clients so they cannot know a server's name, IP address, and so forth. You can also use a proxy server to log activity and prevent denial of service attacks.

To clarify what a “reverse” proxy is: A reverse proxy accepts requests from the client and forwards them onto another server, receives the response from that other server, and returns the results to the client. The reverse proxy is the only server the client interfaces with.

ZENDCORE server instance

As discussed in Chapter 3, “Administration” on page 11, a full instance of the IBM HTTP Server for i5/OS (called ZENDCORE) is installed by default by the Zend Core for i5/OS installation process. It listens on port 89, and forwards requests to the internal Apache server (which is running in PASE for i5/OS) on port 8000.

However, because these two HTTP servers are running on the same hardware platform, we do not achieve any of the security benefits (mainly a DMZ) that often are associated with a reverse proxy configuration. Therefore we strongly advise you to create a full DMZ as described in *IBM HTTP Server (powered by Apache): An Integrated Solution for IBM eSeries iSeries Servers*, SG24-6716. It is available at:

<http://www.redbooks.ibm.com/abstracts/sg246716.html?Open>

7.1.4 User profiles

As outlined in Chapter 3, “Administration” on page 11, a user profile called NOBODY is created by the Zend Core for the i5/OS installation procedure. All PHP jobs run under this user profile, and it is not possible to specify that any given incoming request be executed under any other user profile.

When using the `ibm_db2` driver and the Toolkit to access i5/OS data and resources, you can specify that the connection is made with a specific user profile. If no user ID and password is provided, the connection will be made using the NOBODY profile. If a user ID and password is provided, then the operations that are specified for that connection (for example, run an SQL query, or execute an i5/OS program) are executed under the specified user profile. Access to i5/OS objects follows normal i5/OS object security rules.

7.1.5 More information

Visit these Web sites for more information about security considerations.

The following site describes Web security in general:

<http://www.w3.org/Security/Faq/>

This information is specific to PHP:

<http://www.php.net/manual/en/security.php>



Performance

This chapter provides details about performance tuning of PHP on i5/OS.

Note: This section is far from comprehensive. Much more information will become available over the upcoming months and years as Zend for i5/OS gains wide acceptance in the marketplace.

8.1 Performance tuning

Tuning PHP for optimal performance on i5/OS is not a simple task because numerous components and layers are involved in providing the support. We discuss each of these in more detail in this section. In most instances, we do *not* provide low-level performance tuning tips, but rather pointers to other excellent sources of information. We only get into specifics when the topic is unique to the PHP on i5/OS support.

8.1.1 Hardware

Of course, if your System i is underpowered, PHP will not perform well. Currently, the IBM Systems Workload Estimator (WLE, at <http://www-912.ibm.com/estimator/index.html>) does not provide support for estimating PHP workloads on i5/OS.

However, the Estimator does provide support for modeling WebSphere workloads. We recommend using a non-EJB workload as a rough guideline for PHP. Figure 8-1 shows recommended parameters.

IBM Systems Workload Estimator - Mozilla Firefox

File Edit View Go Bookmarks Tools Help del.icio.us

http://www-912.ibm.com/wle/EstimatorServlet?postcache=111648

IBM Systems Workload Estimator

United States [select] Terms of use

Search

Home Products Services & solutions Support & downloads My account

Version: 2008.5.frc.1 29-Nov-08 www-912

IBM Systems Workload Estimator

Workload Selection Workload Definition Help/Tutorials

WAS #1

Partition Name: Main #1
i5/OS™ - V5R4
No LPAR

WAS Workload Definition

A visit is a group of transactions from a given user.

1. How many visits per hour are anticipated during the busiest time of the day? 0.0

2. How many web requests occur during a typical visit? 0.0

3. How many active users are anticipated at the busiest time of the day? *CALC

A typical web application consists of multiple logical layers. In this application, how are each of the following layers implemented?

4. Presentation JSPs

5. Business Logic Servlets

6. Data Access Direct JDBC

Most WebSphere applications access a database.

7. Is the database for this application:

☒ On the same partition as the WebSphere application

☐ On a separate partition or system

☐ Not used

Back Continue

About IBM Privacy Contact

Adblock Now: 23° F Wed: 31° F Thu: 22° F Fri: 32° F Sat: 29° F Sun: 24° F

Figure 8-1 Recommended parameters for estimating PHP workload using Workload Estimator

Important: Using Workload Estimator in this way to model PHP behavior is very rough, and should only be used as a starting point for sizing. Your actual results might vary significantly.

8.1.2 i5/OS

It is very important that the “core” i5/OS operating system be well tuned. There is a wealth of information about how to do this. For many valuable links, visit:

<http://www.ibm.com/servers/eserver/support/series/planning/perfcapacity.html>

A Redpaper is available that provides a nice overview of the main performance analysis tools that are available. It is called *IBM eSeries iSeries Performance Management Tools*, REDP-4026, and is available at:

<http://www.redbooks.ibm.com/abstracts/redp4026.html?Open>

8.1.3 i5/OS PASE

Little performance tuning is available for the i5/OS PASE environment. Most tuning of this environment flows directly from how the core operating system, i5/OS, is configured.

8.1.4 DB2 for i5/OS

In addition to the configuration of i5/OS itself, tuning DB2 for i5/OS is probably the other main area that requires attention in the way of performance. As with base i5/OS, there exists a great deal of information about this important topic, including:

- ▶ DB2 for i5/OS tuning: *Preparing for and Tuning the SQL Query Engine on DB2 for i5/OS*, SG24-6598

<http://www.redbooks.ibm.com/abstracts/sg246598.html?Open>

- ▶ *SQL Performance Diagnosis on IBM DB2 Universal Database for iSeries*, SG24-6654

<http://www.redbooks.ibm.com/abstracts/sg246654.html?Open>

8.1.5 Apache

Good information about tuning Apache for maximum performance on i5/OS is available in *IBM HTTP Server (powered by Apache): An Integrated Solution for IBM eSeries iSeries Servers*, SG24-6716, which can be found at:

<http://www.redbooks.ibm.com/abstracts/sg246716.html?Open>

8.1.6 Zend Core for i5/OS

Fortunately, there is a great deal of existing information about making Zend Core perform on various operating systems, and much of that wisdom “ports” directly to i5/OS.

The Zend Developer Zone (<http://devzone.zend.com>) has many articles and tutorials about PHP performance. A Google search will reveal many other current articles related to performance tuning tips for PHP.

Archived

Troubleshooting

This section focuses on identifying and resolving problems in the PHP environment.

An inherent difficulty that arises with Web technologies in general is that they are usually comprised of multiple components. This affects the process of troubleshooting a Web environment because it can be hard to determine in which component the problem is occurring. Figure 9-1 shows several places where failure can occur.

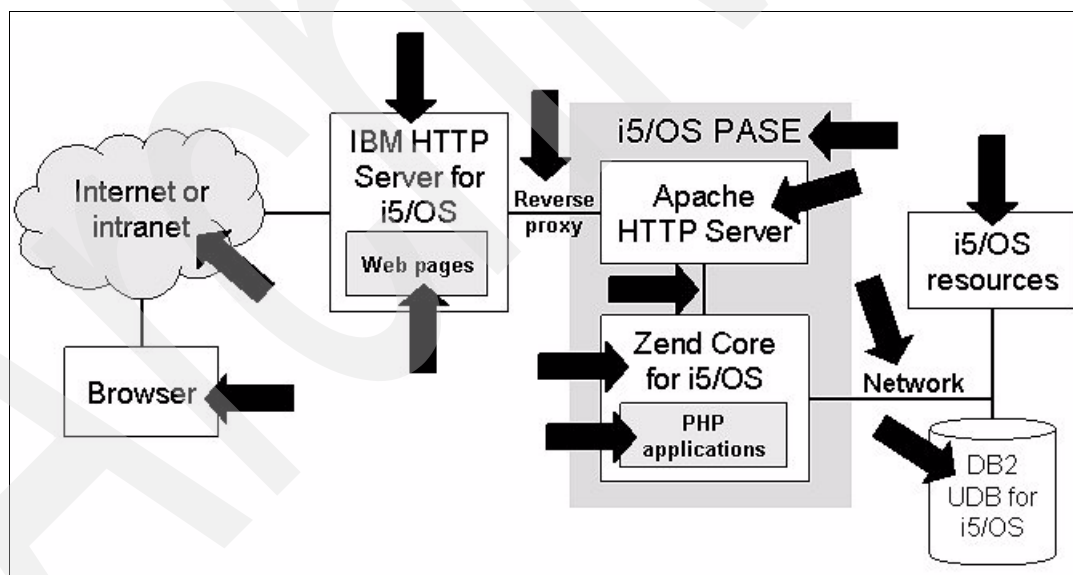


Figure 9-1 Troubleshooting a multiple-component Web topology

Note: Although it might seem that the Web environment is fragile, it is important to note that if you have correctly configured and tested your environment, the vast majority of failures you see occur as a result of application errors, not because of the environment componentry.

When troubleshooting a multiple-component topology, you first need to narrow down the problem to a particular component. Determine how far a request gets in the topology, and you have identified the best place to start looking for a problem. This can be tricky considering that most symptoms of a problem first surface in the browser. Work your way through the topology from the beginning. Construct a series of questions and related tests that help you determine whether a particular component is working correctly.

Here are examples of the types of questions to ask to locate a problem:

- ▶ Can you view similar content in the browser?
- ▶ Can you ping the server?
- ▶ Can you access static Web pages from your HTTP server?
- ▶ Can you access other dynamic pages from your HTTP server?
- ▶ Can you access other PHP applications from the same Zend Core installation?
- ▶ If your application accesses external resources, such as databases and files, can you manually access them?

9.1 Before you start

Working your way through a Web topology and testing each component can be tedious and time-consuming. Often, you can reduce your troubleshooting time significantly by ruling out the most common problems.

By far the most common problem is incorrect application code, so it is helpful to verify that the application code is correct. If the application works in the development environment but not in the test environment, the application code is likely not the cause of the problem. If not, you should debug the application. Zend Studio for i5/OS provides a good debugger for PHP applications. Debugging is covered in the application development chapter.

Next, make sure all the servers have been started. See the administration chapter for more information about checking that the appropriate jobs are running. If one or more servers are stopped, start them. Nothing is more frustrating than to be well into your troubleshooting routine and discover that the problem could have been resolved in a few minutes if you had first checked that the servers were running.

Another good first step to take is to verify that the latest fixes have been applied to the software stack. Your problem might have been encountered by someone else and addressed with a fix. If you later determine that finding the cause of the problem is beyond your abilities, the first question any support organization asks is whether you are current with your fixes.

9.2 Troubleshooting the browser

Problems with the browser are actually quite rare. One of the benefits of using a server-side technology such as PHP is that the browser performs little or no logic processing, and the response is sent to the browser as a static Web page. However, if the browser is responsible for processing any client-side application code, such as JavaScript, check that the code is correct. Most modern browsers include utilities that check scripting code. Also keep in mind that code might not be cross-browser compatible. That is, make sure that the code works in a couple of different browsers, such as Microsoft® Internet Explorer® and Mozilla Firefox.

If your application uses a technology that is not part of the default browser installation (such as plug-ins for special types of media), make sure that any necessary software has been installed.

Most browsers allow users to disable client-side technologies, such as JavaScript, Java applets, and cookies. If your application depends on these, be sure they are not disabled. (It is also helpful to inform your users that these technologies have to be enabled to use your application.)

Some browsers or workstation-based firewalls can add code to Web pages to block pop-ups, remove ads, and disable client-side technologies. This inserted code can cause problems with Web applications. Browsers typically enable you to view the source code for the currently loaded page. Check the source to see whether code has been added that might affect your application.

9.3 Troubleshooting the Web servers

The first thing to check with the Web servers is to make sure that the TCP/IP ports they use are not in conflict with some other application. The characteristic symptom of a port conflict is a server that starts but does not remain running. Use the NETSTAT *CNN command and F14 to view your system's port usage.

Specify option 8 (Display jobs) to show what jobs are using a particular port. Figure 9-2 shows that the I5_COMD job uses port 6077. Note that the default port is 6078; this can be changed using the ZCMENU options.

Work with TCP/IP Connection Status

Type options, press Enter.
 3=Enable debug 4=End 5=Display details 6=Disable
 8=Display jobs

Display Jobs Using Connection

Connection type : *TCP
 Local address : *
 Local port : 6077
 Remote address : *
 Remote port : *

Type options, press Enter.
 5=Work with job

| Opt | Name | User | Number | Opt | Name |
|-----|---------|------|--------|-----|------|
| 8 | I5_COMD | QTCP | 292914 | | |

| Local Port | Idle Time | State |
|------------|-----------|--------|
| 515 | 145:05:43 | Listen |
| 657 | 145:05:53 | Listen |
| 2001 | 000:28:52 | Listen |
| 2301 | 145:05:59 | Listen |
| 3000 | 145:06:03 | Listen |
| 4111 | 145:04:03 | Listen |
| 5544 | 145:05:25 | Listen |
| 5555 | 145:05:14 | Listen |
| 6077 | 042:24:01 | Listen |
| 8000 | 000:10:43 | Listen |
| 8470 | 145:05:10 | Listen |
| 8471 | 145:05:09 | Listen |

Figure 9-2 Displaying jobs that use a port

You should also check the IBM HTTP Server for i5/OS configuration to ensure that you are using the correct URL, including port number, to invoke the application. URLs are typically based on a Web page's location, relative to the Web server's document root directory structure; however, aliases and redirections can be configured to mask the document root structure. You also might check the system's TCP/IP configuration with the Configure TCP/IP (CFGTCP) command to ensure that the host name you are using in the URL is correct.

Another item to check in the configuration of the Web servers is that the IBM HTTP Server for i5/OS server forwards requests to the PASE Apache HTTP Server using the appropriate port number, which defaults to 8000.

LDAP Configuration
 Servlet and JSP Enablement

Server Properties

Tools
 Display Configuration File
 Edit Configuration File
 Directive Index
 Real Time Server Statistics

General Settings Forward Proxy Reverse Proxy Proxy Chaining

Reverse proxy capabilities: Enabled

Proxy requests to remote servers:

| | Request type | Local virtual path | Remote server URL |
|---------|---------------------|--------------------|--------------------------|
| Example | Client requests | /mirror/foo | http://www.myserver.com/ |
| Example | Redirected requests | /mirror/foo | http://www.myserver.com/ |
| | Client requests | / | http://127.0.0.1:8000/ |
| | Redirected requests | / | http://127.0.0.1:8000/ |

Figure 9-3 Default port numbers for the Web servers

Check the logs for the Web servers for any error messages. Because both Web servers are based on the Apache HTTP Server, their logging functions are similar. By default, the Web servers gather information into two logs: the error log and the access log. The value of the error log is obvious. The access log records server resources being accessed. You can use the access logs for both Web servers to ensure that a request arrived at both servers. If the request is logged in the IBM HTTP Server for i5/OS access log and not in the PASE Apache HTTP Server access log, then you know the problem must lie in the linkage between the two servers.

You can also configure the servers to use custom logs, based on whatever criteria you decide. For more information about this and the Web server logs in general, see the product documentation:

- ▶ IBM HTTP Server for i5/OS
<http://publib.boulder.ibm.com/infocenter/iserics/v5r4/topic/rzaie/rzaieconfiglogs.htm>
- ▶ Apache HTTP Server
<http://httpd.apache.org/docs/2.0/logs.html>

9.4 Troubleshooting Zend Core

The log files for Zend Core are located in the `/usr/local/Zend/Core/logs` directory. There are two PHP logs:

- ▶ `php_error_log`: This log contains error messages for PHP scripts.
- ▶ `ini_modifier_log`: The configuration for Zend Core is stored in the `php.ini` file. Changes to this file (that is, the configuration) are logged in the `ini_modifier_log` file. You can check this log for recent changes to Zend Core that might affect your application.

There are two logs for the internal Apache server located in the `/usr/local/Zend/apache2/logs` directory:

- ▶ `access_log`: This log contains a list of all accesses made to the server. It (depending on configuration) lists the requesting IP address; the type of request (GET, PUT); the resource (URL) requested; and the return codes.
- ▶ `error_log`: This log contains a list of errors that have occurred in the internal Apache server running in PASE for i5/OS.

Error handling and logging are configured in the Zend Core console. Click the **Configuration** tab and then the PHP tab. Expand the **Error Handling and Logging** section, which is shown in Figure 9-4.

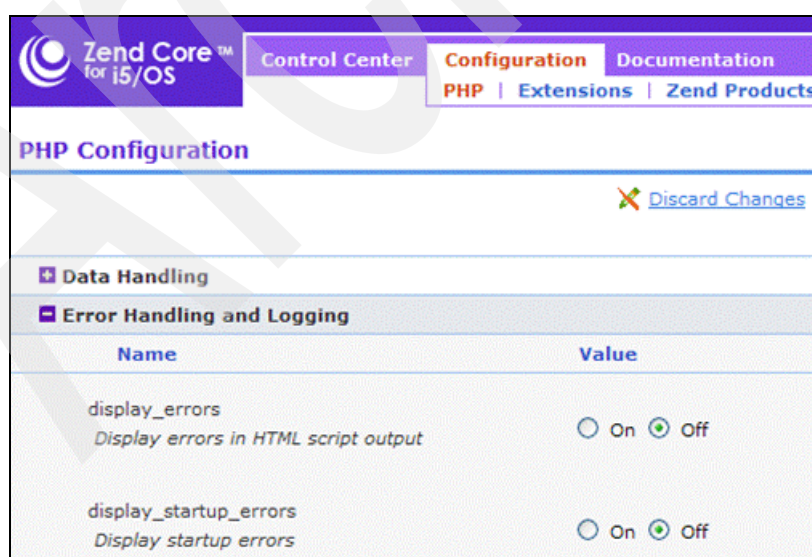


Figure 9-4 Error handling and logging settings in the console

The console contains several settings. These are important settings you should know:

- ▶ `display_errors`: If set to **On**, error messages are written to the browser. This feature can be very helpful in determining when an error is occurring during application processing.
- ▶ `display_startup_errors`: By default, startup errors are suppressed. If you have problems starting the server, change this setting to **On**.
- ▶ `error_log`: Specifies the path and name of the error log file.
- ▶ `error_reporting`: Sets the level of error reporting. By default, this property is set to the highest level (ALL). For production-level systems, the value can be reduced.
- ▶ `log_errors_max_len`: Sets the maximum size of the log files. When log files reach this size, a new file is started.

For more information about PHP logs, see “Error Handling and Logging Functions” in the PHP documentation:

<http://www.php.net/manual/en/ref.errorfunc.php>

If your application uses a particular PHP extension, use the console to check that the extension is enabled. Click the **Configuration** tab and then the **Extensions** tab. Extensions are listed with corresponding icons that communicate their status. A red exclamation mark, as shown in Figure 9-5, denotes an error with an extension.

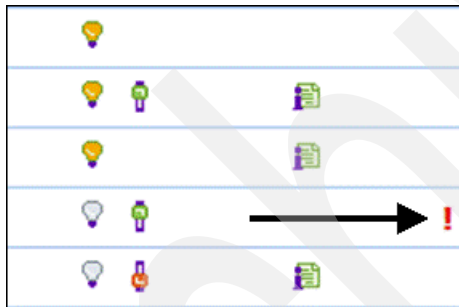


Figure 9-5 Icons that represent extension status

If you suspect that requests from the PASE Apache HTTP Server is not reaching Zend Core, check the Zend directives in the Apache configuration file, `httpd.conf`, which is located in the `/usr/local/Zend/apache2/conf` directory. The Zend directives are at the end of the file. Periodically back up this file so you have something to use for comparison.

9.5 Troubleshooting application resources

If your application accesses system resources, such as files and databases, there are several techniques you can use to troubleshoot this area. Many use standard i5/OS troubleshooting facilities. For more information about i5/OS logs and resources, see “Troubleshooting” in the iSeries Information Center:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/rzahb/rzahbrtrbshool.htm>

9.5.1 Working with object authorities

Often resource access fails if the user profile the application uses does not have sufficient authority to the resource. This includes resources such as libraries, files, and programs. Use the Display Object Authority (DSPOBJAUT) command to check authorities for i5/OS objects,

and the DSPAUT (Display Authority) command for objects in the Integrated File System (IFS). The user profile you use for access (either one you specify or the default, NOBODY) usually needs at least read (*R) authority.

Figure 9-6 shows the authorities for the PHPTRB library. Note that *PUBLIC user authority is set to *EXCLUDE, and only the user profile that owns the object can change it. In this case, you should grant *READ authority to the NOBODY user profile (or a non-default profile your application uses).

| Display Object Authority | | | |
|--|-------|------------------|-------------|
| Object | : | PHPTRB | Owner . . |
| Library | : | QSYS | Primary gro |
| Object type | : | *LIB | ASP device |
| Object secured by authorization list | | | |
| User | Group | Object Authority | |
| *PUBLIC | | *EXCLUDE | |
| QDFTOWN | | *CHANGE | |

Figure 9-6 Displaying authorities for PHPTRB.LIB

The Grant Object Authority (GRTOBJAUT) command can be used to authorize a user profile to a particular resource.

9.5.2 Working with i5/OS job logs

Most Zend Core jobs do not log messages to the i5/OS job logs. Rather, the errors are written to the product log files. However, if your PHP application accesses a resource service that runs in its own job, you might want to look for messages in the job log.

To access a job log, perform the following steps:

1. Use the Work with Active Job (WRKACTJOB) command to display running jobs. The subsystem (SBS) parameter can be used to display only the running jobs in a particular subsystem.
2. Specify option 5 (Work with Jobs) in front of the job.
3. On the Work with Job display, specify option 10 (Display Job Log) on the command line.

Tip: In the job log display, position the cursor over a message and press F1 for more details about the message.

The Work with Jobs display has other options that can be useful in troubleshooting, such as viewing the command stack.

9.5.3 Working with spooled files

If a job ends abnormally, error messages are written to spooled files on an output queue, QEZJOBLOG. These files are very helpful in tracking down database access problems.

Run the WRKSPLF SELECT(<user_profile>) command to view messages for jobs that are no longer running (where <user_profile> is the user profile under which the job was running). Specify option 5 (Display) in front of the file you want to view.

In Figure 9-7, the PHPUSER user profile is used for the database connections, which is specified by the QSQSRVR user data. These files (QPJOBLOG) contain messages about accessing the database.

Work with All Spooled Files

Type options, press Enter.

1=Send 2=Change 3=Hold 4=Delete 5=Display 6=Release 7=Messages
8=Attributes 9=Work with printing status

| Opt | File | User | Device or Queue | User Data | Sts | Total Pages | Cur Page | Copy |
|-----|----------|---------|-----------------|-----------|-----|-------------|----------|------|
| — | QPJOBLOG | PHPUSER | QEZJOBLOG | QSQSRVR | RDY | 2 | | 1 |
| — | QPJOBLOG | PHPUSER | QEZJOBLOG | QSQSRVR | RDY | 2 | | 1 |
| — | QPJOBLOG | PHPUSER | QEZJOBLOG | QSQSRVR | RDY | 2 | | 1 |
| — | QPJOBLOG | PHPUSER | QEZJOBLOG | QSQSRVR | RDY | 2 | | 1 |
| — | QPJOBLOG | PHPUSER | QEZJOBLOG | QSQSRVR | RDY | 2 | | 1 |
| — | QPJOBLOG | PHPUSER | QEZJOBLOG | QSQSRVR | RDY | 2 | | 1 |
| 5 | QPJOBLOG | PHPUSER | QEZJOBLOG | QSQSRVR | RDY | 2 | | 1 |
| — | QPJOBLOG | PHPUSER | QEZJOBLOG | QSQSRVR | RDY | 2 | | 1 |
| — | QPJOBLOG | PHPUSER | QEZJOBLOG | QSQSRVR | RDY | 2 | | 1 |

Figure 9-7 Spooled files on the PHPUSER output queue

9.5.4 Working with the history log

In some cases, you might need a system-wide view to determine the source of a problem. The i5/OS history log (QHST) contains messages from various activities on the system. This historical information is helpful for determining when jobs started and stopped and if some event was responsible for a job ending. Also, messages about hardware problems are logged. Figure 9-8 shows a sample history log.

Display History Log Contents

Subsystem ZEND in library ZENDCORE starting.

Subsystem ZEND started.

Job 291064/QTCP/I5_CMD started on 09/03/06 at 20:51:54 in subsystem ZEND in

Job 291065/ZENDADMIN/ZC_STR_ASJ started on 09/03/06 at 20:51:55 in subsystem

Job 291066/ZENDADMIN/ZC_STR_PRN started on 09/03/06 at 20:51:55 in subsystem

Job 291067/ZENDADMIN/ZCOREJOB0 started on 09/03/06 at 20:51:55 in subsystem Z

Job 291069/QTMHHTTP/ZENDCORE started on 09/03/06 at 20:51:55 in subsystem QHT

Job 291067/ZENDADMIN/ZCOREJOB0 ended on 09/03/06 at 20:51:55; 1 seconds used;

Job 291066/ZENDADMIN/ZC_STR_PRN ended on 09/03/06 at 20:51:55; 1 seconds used

Sep 3 20:51:55 prngd[5842]: prngd 0.9.29 (12 Jul 2004) started up for user u

Sep 3 20:51:55 prngd[5842]: have 7 out of 2000 filedescriptors open

Job 291065/ZENDADMIN/ZC_STR_ASJ ended on 09/03/06 at 20:51:56; 1 seconds used

Job 291077/QSECOFR/ZENDCOREAP started on 09/03/06 at 20:51:56 in subsystem ZE

Job 291084/QTMHHTTP/ZENDCORE started on 09/03/06 at 20:51:57 in subsystem QHT

Job 291085/QTMHHTTP/ZENDCORE started on 09/03/06 at 20:51:57 in subsystem QHT

More...

Press Enter to continue.

Figure 9-8 The history log (QHST)

The history log on production systems might be extremely large. When displaying the history log, you want to display only the messages for a certain time period. Type the Display Log (DSPLOG) command and press F4 to enter the time period.

9.5.5 Working with message queues

Message queues are associated with i5/OS user profiles. Sometimes, useful troubleshooting information is written to these queues. Use with `WRKMSGQ MSGQ(<user_profile>)`, where `<user_profile>` is the name of the user profile.

The following list shows user profiles whose message queue you might want to check:

- ▶ **QSYSOPR:** System administrative messages are logged here.
- ▶ **QTCP:** Messages about TCP/IP and the i5 Toolkit for PHP are written here.
- ▶ **NOBODY:** By default, Zend Core and the PASE Apache HTTP Server run under this user profile. Additionally, if no user profile is specified on a connection, system access occurs under the NOBODY profile.
- ▶ **QTMHHTTP:** By default, the IBM HTTP Server for i5/OS runs under this user profile.

9.5.6 Using a debugger

Although the main goal of a debugger is to identify errors in application code, it can also be used to examine resource access. By monitoring return values, you can identify which accesses fail or return an unexpected error. Use the Zend Studio for i5/OS debugger to step through the application code and verify resource access.

9.5.7 Logging Toolkit activity

This setup creates a LOGFILE in the ZENDCORE library where the Toolkit is going to write requests received from the PHP scripts and messages received from i5/OS system.

- ▶ Toolkit log file setup:
`ADDLIBLE ZENDCORE`
`CFGGEAC LIB(ZENDCORE) LOGLEV(4) JOBLLOG(*YES)`
- ▶ To turn off the Toolkit logging, run the following command:
`CFGGEAC LIB(ZENDCORE) LOGLEV(0) JOBLLOG(*NO)`

Archived



Globalization

This chapter provides details about the support of non-English languages when running PHP on i5/OS, which is somewhat complex because PHP currently does not support Unicode. We show concrete examples of how to enable the Cyrillic alphabet, which is interesting in that it forces the use of atypical code pages/CCSIDs.

We also discuss what types of non-English character data can be queried from DB2 and used within PHP.

10.1 Overview

This section provides a brief overview of existing internationalization support in PHP 5, and then a description of each “layer” involved in providing PHP support in i5/OS and how it influences internationalization support.

10.1.1 PHP internationalization support

PHP 5 does not support Unicode, which makes internationalization support more complicated. According to Zend, this support will be added in the next release (PHP 6), but until then, it is necessary to find other ways to provide this functionality.

There is support for UTF-8 (see <http://tools.ietf.org/html/rfc3629> for the detailed specification). You can find a good overview at <http://en.wikipedia.org/wiki/Utf-8>. UTF-8 support is known as CCSID 1208 on i5/OS.

To configure PHP to use UTF-8 it is necessary to modify the `php.ini` file (found in `\usr\local\Zend\Core\etc`). Edit this file and add the following line under the existing (commented out) entry:

```
default_charset = "UTF-8";
```

This should enable pages written in languages such as English, Spanish, German, or French to continue working without change.

10.1.2 Layers on i5/OS

Given the way PHP support is implemented on i5/OS, there are several components that “touch” the data as it is passed between the browser and the actual PHP page under execution. It is very important to understand how each of these components influences national language support.

i5/OS

Traditionally, i5/OS has always been an EBCDIC-based machine. The vast majority of applications running on the box use data that is encoded in an EBCDIC-coded character set ID (CCSID). For a list of i5/OS-supported CCSIDs, visit:

<http://publib.boulder.ibm.com/infocenter/iserics/v5r4/index.jsp?topic=/nls/rbagsccsidcdepgscharsets.htm>)

Many CCSID transformations can and do occur automatically when running applications under i5/OS. For example, Java is a Unicode-based language. When EBCDIC data is extracted from DB2 for i5/OS via the JDBC driver, an automatic transformation occurs from EBCDIC to Unicode. Many similar such transformations occur, often unbeknownst to the application developer and user.

PHP runs in the IBM i5/OS Portable Application Solutions Environment (PASE) under i5/OS. When an i5/OS PASE shell is created from i5/OS, many globalization-related configuration settings of the i5/OS job are passed to the i5/OS PASE environment.

For a full discussion of i5/OS globalization see:

<http://publib.boulder.ibm.com/infocenter/iserics/v5r4/index.jsp?topic=/nls/rbagsglobalmain.htm>

i5/OS PASE

The i5/OS PASE environment is an integrated runtime that makes it easy to run AIX applications from i5/OS. The following is a simplified view of how i5/OS PASE interacts with i5/OS and the System i hardware:

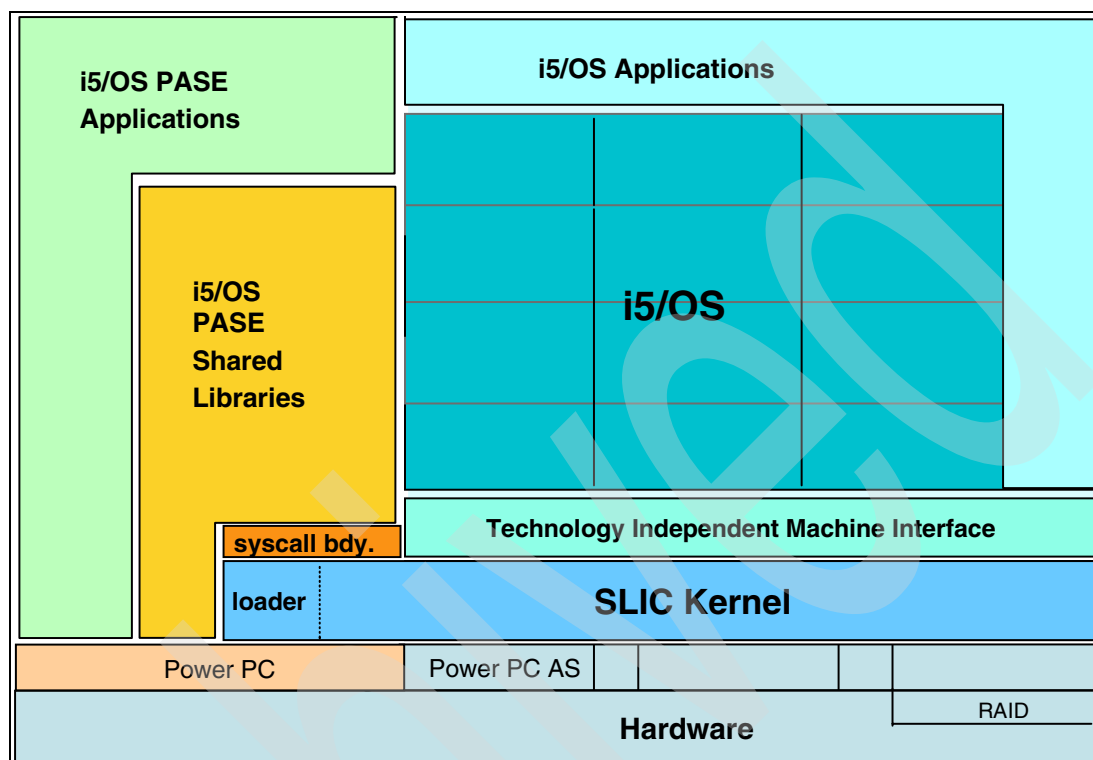


Figure 10-1 Simplified view of i5/OS PASE

PHP is implemented as an i5/OS PASE application that is initiated from i5/OS. Because of this, the i5/OS PASE environment that PHP runs in inherits its globalization from the i5/OS job that started it. For example, first issue the following command from i5/OS:

```
CHGJOB LANGID(RUS) CNTRYID(RU) CCSID(1025)
```

This changes the job's language and country IDs, and the job's CCSID to 1025 (which is the primary EBCDIC CCSID for Cyrillic).

Then start an i5/OS PASE terminal:

```
CALL PGM(QP2TERM)
```

As shown in Example 10-1, several globalization environment variables have been set automatically (in bold).

Example 10-1 Environment variables in new i5/OS PASE environment

```
$
> env
_=/QOpenSys/usr/bin/env
LANG=ru_RU
PASE_LANG=ru_RU
QIBM_PASE_DESCRIPTOR_STDIO=T
PATH=/QOpenSys/usr/bin:/usr/ccs/bin:/QOpenSys/usr/bin/X11:/usr/sbin:./usr/bin
ROWS=17
```

```

QIBM_DESCRIPTOR_STDIN=CRLN=Y
COLUMNS=129
PASE_PATH=/QOpenSys/usr/bin:/usr/ccs/bin:/QOpenSys/usr/bin/X11:/usr/sbin:./usr/bin
LC__FASTMSG=true
LOGNAME=QSECOFR
LOCPATH=/usr/lib/nls/loc
PASE_LC__FASTMSG=true
QIBM_IFS_OPEN_MAX=66000
QIBM_USE_DESCRIPTOR_STDIO=I
QIBM_PASE_CCSID=915
PASE_SHELL=/QOpenSys/usr/bin/sh
SHELL=/QOpenSys/usr/bin/sh
PASE_LOCPATH=/usr/lib/nls/loc
HOME=/home/QSECOFR
PASE_TZ=UTC0
PASE_NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lib/nls/msg/%L/%N.cat:/usr/lib/nls/msg
/ru_RU/%N:/usr/lib/nls/msg/ru_RU/%N.cat
PWD=/
TZ=UTC0
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lib/nls/msg/%L/%N.cat:/usr/lib/nls/msg/ru_RU/%
N:/usr/lib/nls/msg/ru_RU/%N.cat
$

```

==>

| | | | |
|-----------|----------|-------------|----------------------|
| F3=Exit | F6=Print | F9=Retrieve | F11=Truncate/Wrap |
| F13=Clear | F17=Top | F18=Bottom | F21=CL command entry |

We see that i5/OS PASE is running with CCSID 915, which happens to be one of the main ASCII CCSIDs for Cyrillic.

For a full discussion of i5/OS PASE, see:

http://publib.boulder.ibm.com/infocenter/iserics/v5r4/index.jsp?topic=/rzalf/rzalf_intro.htm

PHP

The most important configuration parameter from a language perspective in PHP is the `default_charset` directive. Some plug-ins, such as the DB2 and ODBC drivers, use the value of this directive to determine what sort of data transformation is required.

This directive can be found in the `php.ini` file, which by default on i5/OS is located in the `/usr/local/Zend/Core/etc` directory. Setting this directive causes PHP to add the configured character set to be sent down to the browser in the Content-Type header.

Apache

For the most part, no special configuration of the Apache environments is required to modify globalization settings. The HTTP servers largely act as a “pipe,” passing the PHP-generated data straight through.

If you use straight HTML files in addition to PHP files, you might have to make sure that the Content-Type header is set correctly. You can do this manually by adding the code to the header section of each HTML file:

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```


An alternate way to accomplish this is:

```
<?php
    header('Content-Type: text/html; charset=utf-8');
?>
```

Note that this configuration is for UTF-8.

Important: Make sure that your PHP file outputs the Content-Type header before any actual data intended for display is written out. In other words, you must do this within the header section and not the body.

DB2 for i5/OS

Encoding character data in DB2 can be a complex topic. As mentioned previously, depending on the way the data is encoded in DB2 for i5/OS, and what format the consuming application wants the data in, transformations can occur.

Table 10-1 shows three ways to encode Russian (Cyrillic alphabet) data in DB2 for i5/OS.

Table 10-1 Ways of encoding Russian in DB2 for i5/OS

| Encoding | SQL data type | CCSID | Description |
|----------|--------------------|-------|---|
| EBCDIC | CHAR/VARCHAR | 1025 | EBCDIC Cyrillic Multilingual |
| UTF-8 | CHAR/VARCHAR | 1208 | Unicode, growing, represented as UTF-8 as defined in the Unicode Standard |
| Unicode | GRAPHIC/VARGRAPHIC | 13488 | Unicode UTF-16 as defined in the Unicode Standard |

We could create a single table containing columns of these data types using the DDL shown in Example 10-2.

Example 10-2 Creating table in i5/OS with three Russian character encodings

```
CREATE TABLE MYLIBRARY/CYRILLIC (
    EBCDIC CHAR ( 100) CCSID 1025 NOT NULL WITH DEFAULT,
    UTF8 CHAR ( 100) CCSID 1208 ,
    UNICODE GRAPHIC (100 ) CCSID 13488 NOT NULL WITH DEFAULT)
```

One way to populate the new table is by using the Operations Navigator database functions from a Windows workstation that has the Russian language installed (**Control Panel** → **Regional and Language Options** → **Languages** → **Details...** → **Add**).

There are two main methods of extracting data from DB2 for i5/OS into your PHP application, as detailed in Chapter 5, “Database access” on page 59: using either the DB2 or the ODBC driver. As this book was being written, both of these drivers worked correctly with data encoded as detailed in Table 10-1. This occurred only after performing the configuration steps detailed in the next section.

10.2 Globalization configuration (single-byte)

This section details the configuration required to enable PHP to display non-English languages and extract such data from DB2 for i5/OS. We continue the use of Russian in these

examples and focus on using UTF-8 to enable the support, which is perhaps the most popular method used.

Restriction: Currently it is mandatory to be signed on with the QSECOFR user profile (not just a profile having *SECOFR authority) for the following support to work. This is due to the way Apache security code is written.

10.2.1 i5/OS

As described earlier, change the i5/OS job to be configured for the Russian language:

```
CHGJOB LANGID(RUS) CNTRYID(RU) CCSID(1025)
```

This is the only thing that has to be done in i5/OS. Remember that you have to be signed on as QSECOFR to make this work.

10.2.2 i5/OS PASE

We need to manually change some settings in the i5/OS PASE environment to configure UTF-8 support, because the default mappings used by the operating system when the new i5/OS PASE environment is started are ASCII rather than UTF-8.

Start an i5/OS PASE terminal:

```
CALL PGM(QP2TERM)
```

Issue the following commands to change from an ASCII environment to a UTF-8 one:

```
export QIBM_PASE_CCSID=1208
export PASE_LANG=RU_RU
export LANG=RU_RU
```

You can change PASE_NLSPATH and NLSPATH, although they are not mandatory modifications.

You should now be able to execute the `env` command and see your changes reflected in the i5/OS PASE environment.

10.2.3 PHP

Edit the main PHP configuration file, either via the administration console or directly by editing it manually.

We use Notepad to edit the file, which you can find (`php.ini`) in the following directory:

```
/usr/local/Zend/core/etc
```

You will find a `default_charset` directive there that is commented out. Copy it, remove the comment (semicolon) from the front, and change the value to "UTF-8". It should look like this:

```
default_charset = "UTF-8"
```

10.2.4 Apache

Special configuration is not required in the Apache environment. However, we cannot use the menu options provided by Zend (GO ZENDCORE/ZCMENU) to stop and start the PHP environment because they do not correctly configure the i5/OS PASE environment.

Fortunately, we can take advantage of an easy-to-use command called **apachectl**.

First, issue the following command:

```
cd /usr/local/Zend/apache2/bin
```

Now you can issue the following commands:

| | |
|----------------|---------------------------|
| Start | apachectl -k start |
| Stop | apachectl -k stop |
| Restart | apachectl -k restart |

This should get the PHP server going with the desired configuration.

Tip: You can embed the following code in a PHP file to see the i5/OS PASE environment settings that the PHP engine is running in:

```
<?php
    phpinfo (INFO_ENVIRONMENT);
?>
```

10.2.5 DB2 for i5/OS

After you have performed all of the configuration steps outlined in this section, you should be able to extract text data from DB2 for i5/OS that is encoded in EBCDIC, UTF-8, or Unicode and have it display correctly (as long as the target browser has the correct fonts available).

Archived



Advanced development topics

This chapter discusses some more advanced application development topics. They might be a bit advanced for beginning PHP developers, but will possibly be useful as skills, experience, and application complexity increase.

11.1 Implementation of templates

Using templates leads to cleaner PHP code and more agile applications, because display and application logic are separated. This enables you to change either the application logic or the look and feel of the application without one affecting the other. Templates do have drawbacks, though. As you can see in the following simple examples, a template-based solution contains more code than one without templates. We do not use templates to write less code, but to keep the display and application logic separate.

This example aims to show how to separate display from application logic by using PHP with an HTML template. In Example 11-1 we define the template file that we named `Template.tpl`.

Example 11-1 HTML Template file

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title><?php echo $template->doctype ?></title>
</head>
<body>
<?php echo $template->contentTitle; ?>
<?php echo $template->content; ?>
</body>
</html>
```

The next step consists of creating a class called `Template` with the appropriate functions (see Example 11-2). Our main method, `show($doc)`, receives the name of a document to be displayed as an argument. It sets the path, which is empty in our example, and then executes the template by using the PHP function `include()`.

After that it calls the function `showFooter()`, which prints a line with the current date and the current working directory.

Example 11-2 Definition of class `Template` (`TemplateClass.php`)

```
<?php
class Template {

    public $template_dir;

    function show($doc) {
        $template = $this;
        include($this->template_dir.$doc);
        $this->showFooter();
    }

    function showFooter() {
        printf("<br><br><br>Today's date: %s | Current directory: %s", date('d. M
Y'), getcwd());}
    }
?>
```

Now all that is needed is the main document, which first includes the template class file. Then it creates a new instance of the class `Template` and assigns values to the class attributes.

Finally, the function `show('Template.tpl')` displays the template file, parsing and replacing the values according to what has been set in the file.

Example 11-3 TemplateMain.php, our main file

```
<?php
    include('TemplateClass.php');

    $template = new Template;
    $template->doctitle = 'PHP using an HTML template';
    $template->template_dir = '';
    $template->contentTitle = '<h3>PHP using an HTML template</h3>';
    $template->content = 'Hello, World';
    $template->show('Template.tpl');
?>
```

Example 11-4 shows a more complicated example that combines templates with retrieval of an i5/OS spooled file.

Example 11-4 Combining templates with retrieval of an i5/OS spooled file

```
$conn = i5_connect("127.0.0.1", "${USERNAME}", "${PASSWORD}" );
if ($conn === false)
{
    //echo "FAIL : Failed to connect to server <br>";
    die(i5_errormsg());
}

$spool = i5_spool_list(/*array("username"=>"${USERNAME}", "outq"=>"${OUTQ}",
"userdata"=>"${USERDATA}"),$conn*/);
if ($spool === false)
{
    //echo "Fail: failed to create a spooled file list. <br>";
    die(errormsg());
}

$spool_file = i5_spool_list_read($spool);
if ($spool_file === false)
{
    // echo "Fail: failed to get spooled file data from the queue. <br>";
    die(errormsg());
}

$data = i5_spool_get_data($spool_file['SPLFNAME'], $spool_file['JOBNAME'],
    $spool_file['USERNAME'], $spool_file['JOBNBR'],
    "*LAST" );

if ($data === false)
{
    // echo "Fail: failed to get the data from the spooled file. <br>";
    die(i5_errormsg());
}
// echo "<pre> $data </pre><br>";

$close_spool = i5_spool_list_close($spool);
```

```

if ($close_spool === false)
{
    //print ("FAIL : Failed to free spooled file list resource. <br>");
    i5_errormsg();
}

$close = i5_close($conn);
if ($close === false)
{
    //echo "FAIL : Failed to disconnect from server :$i5_server_ip <br>";
    i5_errormsg();
}
${END}

```

If you are interested in templates but do not want to write a solution yourself, you can find plenty of free template solutions and we recommend them rather than spending time writing yet another template engine. You can find some of them on PEAR:

- ▶ HTML_Template_Flexy (http://pear.php.net/package/HTML_Template_Flexy)
- ▶ HTML_Template_PHPLIB (http://pear.php.net/package/HTML_Template_PHPLIB)
- ▶ HTML_Template_IT (http://pear.php.net/package/HTML_Template_IT)
- ▶ HTML_Template_Sigma (http://pear.php.net/package/HTML_Template_Sigma)
- ▶ HTML_Template_Xipe (http://pear.php.net/package/HTML_Template_Xipe)

In addition to those listed on PEAR, we recommend that you look at the following PHP template solutions:

- ▶ Smarty (<http://smarty.php.net/>)
- ▶ TinyButStrong (<http://www.tinybutstrong.com/>)

If you are more interested in code maintainability and portability, we suggest that you refer to the next section, where you can read more advanced techniques and start learning about recommended design practice when building your PHP project.

11.2 Model View Controller (MVC) design and frameworks

Though PHP offers a quick turnaround, this can be its downfall, leading to quickly built but unmaintainable applications. You should strive for maintainability, extensibility, and reusability in your design, especially if you plan to build a large PHP project.

MVC architecture is a design that can help improve maintainability, extensibility, and reusability. This architecture splits the application into three distinct components with certain responsibilities. This technique decouples, or minimizes, the dependency between any two components by establishing three distinct areas of concern:

▶ The view

The display of the current application state is handled by HTML templates that contain a small amount of PHP code to display data retrieved from the controller. There is no business logic on these pages, only calls to our model classes to obtain data structures based on user input from the controller. Attributes that are present in the model or the request delegated by the controller determine whether messages or forms are displayed.

▶ The controller

The controller is the single entry point to our application that includes each of our display pages. This thus controls the behavior and data supplied to each of our PHP view pages.

The controller also implements security precautions, such as validating and cleansing user input before passing it to the model to be processed.

► The model

The model consists of our application business logic and objects. This is usually implemented as a collection of PHP 5 classes. It uses an intermediary data access object that makes calls to a database interface. The database interface translates commands and provides them to the runtime implementation regardless of the actual database or PHP API for communicating with it. Regardless of the exact database interface, the model objects receive data in the same format and do not need to change if the data tier changes.

In reality, each of these logical separations overlaps a bit into the others. For example, many PHP applications have some controller code in the view pages and some model class names mentioned in the display pages, despite a strong MVC approach.

Luckily, there are MVC frameworks that can help you with MVC design in your future PHP applications. Examples include:

- CakePHP (<http://www.cakephp.org/>)
- Agavi PHP MVC Framework (<http://www.agavi.org/>)
- Phrame (<http://www.phrame.org/>)
- Claw (<http://claw.tigris.org/>)
- The Ismo PHP Framework (<http://ismo.sourceforge.net/>)
- PHP on TRAX (<http://www.phpontrax.com/>)
- Symfony (<http://www.symfony-project.com/>)
- Livepipe: Pipeline Framework (<http://livepipe.net/pipeline/>)
- Zend Framework (<http://framework.zend.com>)

You can find more MVC frameworks for PHP and their quick comparison at:

http://www.phpwact.org/php/mvc_frameworks

When this book was written, most of the listed MVC frameworks did not support IBM DB2 as data source, but did support MySQL or PostgreSQL. In order to work with IBM DB2 databases, you should use ODBC connectivity if possible.

In short, although PHP and Web applications are gaining prominence as an excellent way to build applications quickly, it should be paired with time-tested software development principles to build flexible and maintainable applications.

If you want to know more about the best methods to design your Web application, a good resource for planning your PHP application architecture is Sherri Wheeler's "Mature Design Theory in Web Development":

<http://www.zend.com/php/design/mature-design.php>

A high-level, theoretical discussion of application architecture called "What is a software architecture?," written by Peter Eeles, is available in the Rational® Edge:

<http://www.ibm.com/developerworks/rational/library/feb06/eeles/>

11.3 Tips and tricks

The following important tips might help you develop and administer PHP code.

11.3.1 Accessing system environment variables using predefined arrays

By default, system environment variables cannot be accessed using `$_ENV` or `$_HTTP_ENV_VARS` arrays in Zend Core for i5/OS. You should configure the `variable_order` option in `php.ini` in order to get access to the mentioned array, or you can just use `getenv()` PHP functions to get the value of specific environment variables. Find more information about the `getenv()` function at:

<http://www.php.net/manual/en/function.getenv.php>

11.3.2 Reverse proxy issues

Because Zend Core for i5/OS is implemented using a reverse proxy, some environment variables in `$_SERVER` (or `$_HTTP_*_VARS` equivalents) will become meaningless. For example, `SERVER_NAME` will show `127.0.0.1` and `HTTP_HOST` will show `127.0.0.1:8000`.

To preserve the original, useful `SERVER_NAME` and `HTTP_HOST` (`www.example.org`), add the following directive to your reverse proxy server's configuration (`/www/zendcore/conf/httpd.conf` by default):

```
ProxyPreserveHost On
```

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information about ordering these publications, see “How to get IBM Redbooks” on page 120. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *Bringing PHP to Your IBM eSeries iSeries Server*, REDP-3639
- ▶ *Developing PHP Applications for IBM Data Servers*, SG24-7218
- ▶ *IBM HTTP Server (powered by Apache): An Integrated Solution for IBM eSeries iSeries Servers*, SG24-6716
- ▶ *IBM System i Security Guide for IBM i5/OS Version 5 Release 4*, SG24-6668
- ▶ *Preparing for and Tuning the SQL Query Engine on DB2 for i5/OS*, SG24-6598
- ▶ *SQL Performance Diagnosis on IBM DB2 Universal Database for iSeries*, SG24-6654

Other publications

These publications are also relevant as further information sources:

- ▶ *Beginning PHP and MySQL 5: From Novice to Professional, 2nd Edition*, W. Jason Gilmore, ISBN 1590595521
- ▶ *Learning PHP 5*, David Sklar, ISBN 0596005601
- ▶ *PHP Cookbook, 2nd Edition*, Adam Trachtenberg, et al, ISBN 0596101015

Online resources

These Web sites are also relevant as further information sources:

- ▶ Zend site for i5/OS
http://www.zend.com/products/zend_core/zend_for_i5_os
- ▶ IBM site for Zend Core
<http://www-304.ibm.com/jct03002c/software/data/info/zendcore/>
- ▶ Site with many useful PHP links
<http://www.ibm.com/developerworks/opensource/library/os-php-read/>
- ▶ Pair J2EE with PHP to implement a common Web application infrastructure
http://www.ibm.com/developerworks/websphere/techjournal/0505_krook/0505_krook.html
- ▶ Use PHP on System i: An overview, sample applications, and one group's experiences
<http://www.ibm.com/developerworks/systems/library/es-path2php/>

- ▶ DB2 for i5/OS - Porting Information
<http://www.ibm.com/servers/enable/site/db2/porting.html>
- ▶ IBM Migration Toolkit
<http://www.ibm.com/software/data/db2/migration/mtk/>

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Abbreviations and acronyms

| | | | |
|------------------|--|-------------------|--|
| API | Application Programming Interface | RSTLIB | Restore Object (RST) and Restore Library |
| BDB | BerkeleyDB | RSTLICPGM | Restore Licensed Program |
| CCSID | Coded Character Set ID | RTVNETA | Retrieve Network Attributes command |
| CFGTCP | Configure TCP/IP | SAV | Save Object |
| CLI | Call Level Interface | SAVLIB | Save Object (SAV) and Save Library |
| CLR | Common Language Runtime | SDO | Service Data Objects |
| CMS | Content Management System | SOA | Service Oriented Architectures |
| CPAN | Comprehensive Perl Archive Network | SSL | Secure Sockets Layer |
| cURL | client URL | UDB | Universal Database |
| DAS | Data Access Service | V5R4 | Version 5 Release 4 |
| DDL | Data Definition Language | WRKACTJOB | Work with Active Jobs |
| DLTLICPGM | Delete Licensed Program | WRKAUT | Work with Authority |
| DML | Data Manipulation Language | WRKPTFGRP | Work with PTF Groups |
| DMZ | Demilitarized Zone | WRKUSRJOB | Work with User Jobs |
| DSPOBJAUT | Display Object Authority | WRKUSRPRF | Work with User Profiles |
| DSPPTF | Display PTF | ZENDCOREAP | Zend Core jobs |
| GRTOBJAUT | Grant Object Authority | | |
| GUI | Graphical User Interface | | |
| IBM | International Business Machines Corporation | | |
| IDE | Integrated Development Environment | | |
| ISAM | Index Sequential Access Method | | |
| ISO | International Organization for Standardization | | |
| ISP | Internet Service Provider | | |
| ITSO | International Technical Support Organization | | |
| LDAP | Lightweight Directory Access Protocol | | |
| LICPGM | Licensed Program | | |
| MVC | Model View Controller | | |
| OO | Object Oriented | | |
| PASE | Portable Application Solutions Environment | | |
| PEAR | PHP Extension and Application Repository | | |
| PECL | PHP Extension Community Library | | |
| RDBMS | Relational Database Management System | | |
| RST | Restore Object | | |

Archived

Index

Numerics

32-bit 60
64-bit 60

A

apachectl 34

C

CL call - Toolkit 52
column names 62
connection - Toolkit 51
controller 117

D

database solution 60
DB2 family 60
DB2_ATTR_CASE 62
DB2_CASE_LOWER 62
db2_connect 64
Demilitarized Zone 89
DMZ 89
DSRDBDIRE 60

E

environment 60
extension 3

F

firewalls 97

H

httpd.conf 88

I

i5_command 52

J

JavaScript 97

L

library 64
logic 40

M

Microsoft Internet Explorer 97
Mozilla Firefox 97
multiple installations 30
multiple instances 30

MVC 117
mysql 75
MySQL AB 71
mysqli 75

N

NETSTAT 97
NOBODY user profile 70

O

ODBC 60
open source 71

P

PDO_ODBC 60, 68
Perl 8
persisted information 40
Personal Home Page/Form Interpreter 8
PHP 5 8, 117
PHP community 8
PHP Data Objects 68, 77
php.ini 88
PHP/FI 8
Practical Extraction and Reporting Language 8
program call - Toolkit 52
programming interface 60
programming language 3

Q

QSQRVR 62, 70

R

RDBMS 60
Redbooks Web site 120
 Contact us x
relational database management systems 60
reliability 3
reverse proxy 25, 89
RSS parser 54

S

schema 64, 71
security 71
server mode - ibm_db2 62
SimpleXML extension 54
SOAP 55–56
SSL 88
stability 3

T

Toolkit

- CL call 51
- connection 51
- program call 52

U

- upper-case 62

V

- V5R3 4
- V5R4 4

W

- WRKRDBDIRE 60

X

- XML 54
- XML-RPC 55

Z

- z/OS 60
- Zend engine 8
- ZENDCORE 89



PHP: Zend for i5/OS



Redbooks

**Learn how to install
and administer**

**Discover valuable
development tips and
advice**

**Security,
globalization, Zend
Platform for i5/OS,
and more!**

This IBM Redbook will help you install, configure, and become productive with PHP on System i using Zend Core for i5/OS and Zend Platform for i5/OS. If you are evaluating PHP, this book will also help by providing background information and comparisons with other tools.

Some of the topics addressed include:

- ▶ Installation, configuration and administration
- ▶ Application development
- ▶ Strategies to access DB2 for i5/OS and MySQL data from your PHP applications
- ▶ Security, performance, troubleshooting, and more

Emphasis has been placed on highlighting i5/OS-specific functions and features in this book rather than those generic to PHP on all platforms.

**INTERNATIONAL
TECHNICAL
SUPPORT
ORGANIZATION**

**BUILDING TECHNICAL
INFORMATION BASED ON
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:
ibm.com/redbooks**

SG24-7327-00

ISBN 0738489875