PHP on i5/OS:
A Whole New Stack

The following article is reprinted with the permission of IT Jungle, New York, NY. It first appeared in Four Hundred Guru (http://www.itjungle.com/fhg/fhg062707-story01.html) on June 27, 2007

By Erwin Earley

Development and deployment of Web-based applications is one of the most common activities in today's IT world. For development of Web applications a development stack, consisting of an operating system platform, Web server, database engine, and scripting language is often employed. With the advent of PHP on i5/OS V5R4 and OS/400 V5R3, iSeries and System i customers have a whole new way to skin the Web cat while keeping their favored back-end system right in the thick of things.

One of the most popular of these stacks is the LAMP stack, which stands for Linux, Apache, MySQL, and PHP. Since the introduction of Linux in a logical partition, System i customers have been able to employee Web applications utilizing the LAMP set of resources. Another development stack is the WIMP stack (Windows, IIS, MySQL, and PHP). With the introduction of Zend Technologies' PHP in i5/OS in January 2006, System i customers have had a new Web development stack at their disposal, the iADP stack (i5/OS, Apache, DB2, PHP). This stack can be utilized to deploy Web applications natively in i5/OS. These Web applications can access and manipulate i5/OS resources, including DB2 resources, data areas, data queues, etc.

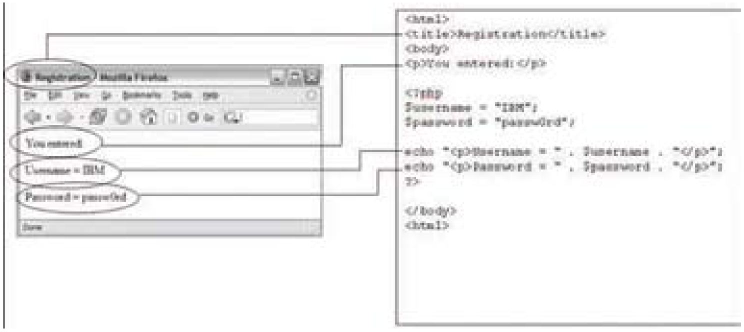
With the announcement of support of MySQL, yet another development stack is made available to System i customers—the iAMP stack (i5/OS, Apache, MySQL, PHP). The iAMP stack facilitates the deployment of open-source applications (such as Wikis, blogs, CRM, e-commerce) natively in i5/OS. This article presents a general introduction to PHP in i5/OS, how the product can be obtained and installed, development environment for the language, and typical uses for the language. This will be the first in an ongoing series of articles covering aspects of both PHP and MySQL in i5/OS.

Introduction to PHP

PHP is an easy to use, open source, platform independent scripting language that has been designed primarily with Web application development in mind. PHP essentially is a data manipulation language for Web-based applications. PHP allows you to imbed business logic inside of the HTML stream that comprises a Web page. It is estimated that there are approximately 2.5 million PHP developers with upwards of 22 million Internet domains leveraging PHP to some degree in their application portfolio.

From a syntax viewpoint, PHP source code is similar in many respects to shell scripts and the C programming language. PHP code is embedded within PHP tags as follows:

```
<?php
    // PHP code
?>
```



Registration - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Done

You entered

Username = IBM

Password = passw0rd

```
<html>
<title>Registration</title>
<body>
<p>You entered:</p>

<?php
$username = "IBM";
$password = "passw0rd";

echo "<p>Username = " . $username . "</p>";
echo "<p>Password = " . $password . "</p>";
?>

</body>
<html>
```

Figure 1. PHP imbedded within HTML

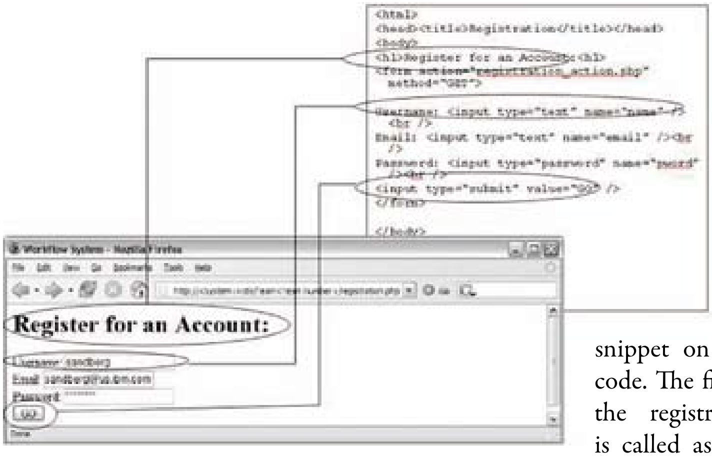
Figure 1 shows a simple example of PHP code embedded inside of HTML.

PHP variables are denoted with the leading '$'. The echo statement outputs the indicated string, while the '.' within the echo statement causes the information on both sides to be concatenated together. You may also notice that HTML tags have been included in the echo statements to cause each output line to be treated as a new paragraph.

Lines of code within PHP are delimited with a semicolon. The PHP engine processes the PHP code and sends the results to the Web browser as HTML.

Perhaps a more interesting example is one in which a Web form is used to request data from the user and then passed to a PHP application. **Figure 2** is a sample Web form.

The <form action> line may be the most important line to discuss from this code snippet. The line causes another file to be executed when a form action (in this case selection of the Go button) is taken. The form action line also defines the Get method, which establishes a "super-global" variable called



```
<html>
<head><title>Registration</title></head>
<body>
<h1>Register for an Account:<h1>
<form action="registration_action.php"
  method="GET">

Username: <input type="text" name="name" />
  <br />
Email: <input type="text" name="email" /><br
  />
Password: <input type="password" name="pword"
  /><br />
<input type="submit" value="GO" />
</form>

</body>
```

Workflow System - Mozilla Firefox
File Edit View Go Bookmarks Tools Help

Register for an Account:

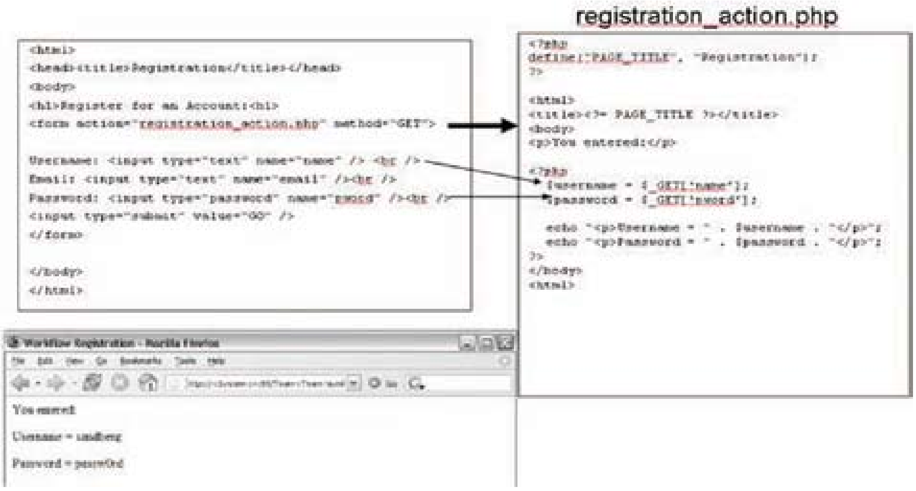
Username: sandberg
Email: sandberg@us.ibm.com
Password: ********
GO

Figure 2. Sample Web form

GET. The PHP processor uses the GET variable to reference the values defined within the HTML form. The values for this example are the values *name*, *email*, and *pword*, as shown in the form field definition lines for Username, Email, and Password.

Let's take a look at the PHP code in file registration_action.php in **Figure 3**.

The code snippet on the left is the HTML code, while the code snippet on the right is the PHP code. The first thing to note is that the registration_action.php code is called as a result of the <form action> from the HTML code. Additionally, note the assignment of values to the $username and $password variables in the PHP code. They are being assigned values from the super-global variable GET, which was assigned within the HTML code. You can view GET as an array with associative keys, where *name*, and *pword* are keys into the array.



```
<html>
<head><title>Registration</title></head>
<body>
<h1>Register for an Account:<h1>
<form action="registration_action.php" method="GET">

Username: <input type="text" name="name" /> <br />
Email: <input type="text" name="email" /><br />
Password: <input type="password" name="pword" /><br />
<input type="submit" value="GO" />
</form>

</body>
</html>
```

registration_action.php

```
<?php
define("PAGE_TITLE", "Registration");
?>

<html>
<title><?= PAGE_TITLE ?></title>
<body>
<p>You entered:</p>

<?php
  $username = $_GET['name'];
  $password = $_GET['pword'];

  echo "<p>Username = " . $username . "</p>";
  echo "<p>Password = " . $password . "</p>";
?>
</body>
<html>
```

Workflow Registration - Mozilla Firefox
File Edit View Go Bookmarks Tools Help

You entered:

Username = sandberg

Password = passw0rd

Figure 3. PHP code

Obtaining and Installing PHP in i5/OS

Obtaining and Installing PHP in i5/OS is a fairly easy process. The first step is to ensure that your system meets the pre-requisites for the Zend Core (PHP Engine) product. The latest information and pre-requisites can be found on the Zend Web site. Most models that run V5R4 or V5R3 are able to run the Zend Core product. These models include the 5XX line (515, 520, 525, 550, 570, and 595) as well as most of the 8XX and 270 line (270, 800, 810, 820, 825, 830, 840, 870, 870, and 890). **Table 1** provides a list of the licensed programs that need to be installed prior to installation of Zend Core.

Once you have ensured that your system meets the pre-requisites, you are ready to obtain and install the Zend Core product. The product itself can be downloaded from the Zend Web site. You will be requested to register for the product before you gain access to the Zend for i5/OS channel. From the channel you will download the Zend Core for i5/OS product as well as the Zend Studio for i5/OS product (discussed in more detail below).

The Zend Core product is distributed as a zip file. (As of publication the name of the zip file is ZendCoreFori5_OS-2.0.1-pase.zip). The zip file contains a number of files including:

- i5Toolkit_Class_README.txt
- LICENSE
- README
- Zcoresavf
- Zend_Core_For_i5OS_2_0_1_Release_Notes-1.txt
- Zend_Core_i5OS_Installation_Guide.pdf
- Zend_Core_User_Guide_i5OS.pdf
- Zend_Navigator_Demo_README.txt

Description	Product Option	Licensed Program
Portable Application Solutions Environment	33	5722SS1
QShell	30	5722SS1
System Openness Includes	13	5722SS1
Digital certificate Manager	34	5722SS1
CCA Cryptographic Service Provider	35	5722SS1
Java Development Kit	5 or 6	57SSJV1
IBM HTTP Server for i5/OS	*BASE	5722DG1
IBM Portable Utilities for i5/OS	*BASE	5733SC1
OpenSSH, OpenSST, Zlib	1	5733SC1

Table 1. Licensed programs pre-requisite to Zend Core

Once you have obtained the product, transfer the ZCoresavf save file to the i5/OS system and install the licensed program it contains. First, create a save file on the target system:

```
CRTSAVF FILE(QGPL/ZCOREPROD) TEXT('Zend Core for
i5/OS Product SAVF')
```

Then transfer the save-file to i5/OS. As an example, with ftp, the command sequence would be similar to the following:

```
ftp
bin
put zcoresavf QGPL/ZCOREPROD
```

Use the Restore License Program command to install the Zend Core product:

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

The installation process will prompt for the definition of a password to be used to access the Web-based administration toolkit for Zend Core (http://:89/ZendCore) as well as a user name and password that you will use when downloading updates. This is the user name and password that you used when you registered for the download from the Zend Web site.

Once you have answered the prompts, the Zend Core product will be installed and configured. The diagram in **Figure 4** shows the product installation and configuration to support PHP in i5/OS.

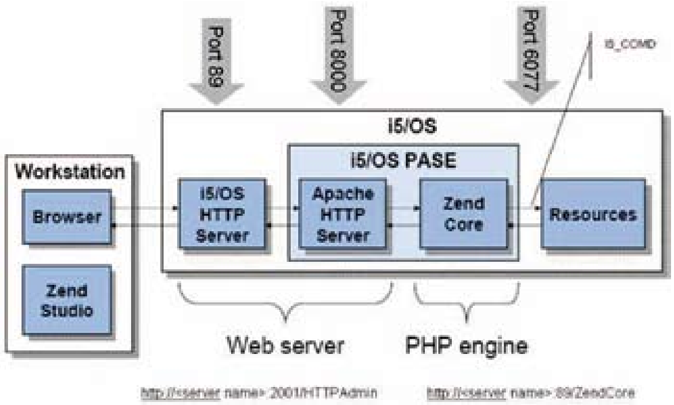


Figure 4. Product Installation and configuration

As you can see from the diagram, in addition to the Zend Core product, the Apache Web Server is also installed in the PASE environment. The i5/OS HTTP Server is configured to forward traffic on port 89 to the Apache Web Server running in the PASE environment. Apache accepts the HTTP request and forwards any PHP code to the Zend Core PHP engine. Zend Core processes the PHP code and forwards requests for i5/OS resources (such as DB2 resources, data queues, data areas, i5/OS commands, and system values) to the I5_COMD process (installed as part of the Zend Core installation) via port 6077. The resulting HTML generated by Zend Core is transferred back to the Apache Web Server which in turn transfers that traffic to the i5/OS HTTP Server, which responds to the browser request from the client.

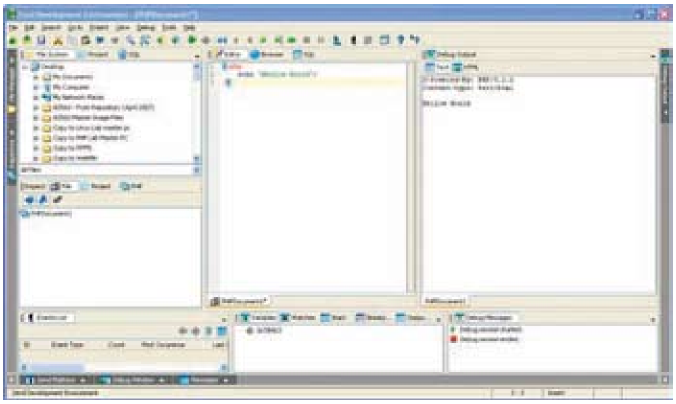


Figure 5. Zend Studio environment

PHP Development Environment

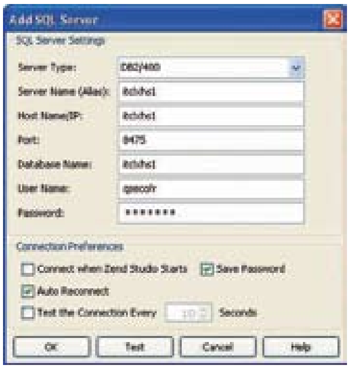
In addition to the Zend Core product, Zend also provides the Zend Studio product for the i5/OS platform. Zend Studio is an Integrated Development Environment that can be used for the development and debugging of PHP applications. Zend Studio is downloaded from the same channel on the Zend Web site as the Zend Core product was obtained from. Zend Studio is available for a number of clients, including Windows, Linux, and Mac OS X. Once the product has been downloaded, it is installed on the client through a typical Windows installer. Starting the Studio provides an environment that looks like **Figure 5**.

```
/* Query file using input from form and deliver results to client */
/* Construct the SQL statement, using lastname as the search substring */
$sql = 'select * from i5schema.employee where lastname like \'JO%\'';
/* Execute the DB2 SQL statement, place results into $stmt */
$stmt = db2_exec($dbh, $sql, array('cursor' => DB2_SCROLLABLE));
/* Print Employee Search Results header and table setup */
print '<b>Employee Records On System i starting with JO</b>';
print '<br><table border=1 cellpadding=5 cellspacing=5>';
/* Iterate through result set, printing one table line per record returned. */
/* Note that the customerNumber field will be an "active field" which will   */
/* $_GET the customerNumber and reinvoke the employee_update.php application*/
   while ($row = db2_fetch_array($stmt)) {
   if (!$row=="") {
      $customerNumber = $row[0];
      $customerName = $row[3];
      $customerFirst = $row[1];
      $workdept = $row[4];
      $job = $row[7];
      $salary = $row[11];
      print '<tr><td
      align=center><a href=employee_update.php?customerNumber='
       .    $customerNumber . '>' . $customerNumber .
'</a><td>'.$customerName.'<td>'.$customerFirst.'<td>'.$workdept.'<td>'.$job.'
<td>'.$salary.'</td></tr>';
   }
}
/* Close table */
print '</table><br>';
/* Print the DB2 SQL statement which was executed - Informational */
print "<p>Echo of dynamically-built sql: ".$sql."</p>";
```

Figure 9. Coding example using APIs

IBM
Premier Business Partner
MIMIX for 99.999% High Availability.
Blair for 100% Satisfaction Guaranteed.

The middle portion of the screen allows you to write and revise PHP source code. The right portion of the screen is where the output/results of running the code locally are displayed. The left portion of the screen provides a file directory tree, as well as project management and an event list for debugging.


Add SQL Server

SQL Server Settings

Server Type: DB2/400
Server Name (Alias): itdxhs1
Host Name/IP: itdxhs1
Port: 8475
Database Name: itdxhs1
User Name: qsecofr
Password: *******

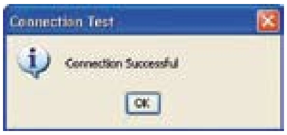
Connection Preferences

□ Connect when Zend Studio Starts ☑ Save Password
☑ Auto Reconnect
□ Test the Connection Every 10 Seconds

Figure 6. Dialog box

One of the neat things about the Studio product is the ability to define an SQL connection within Studio. This can be very useful, as it provides a single point where table structures can be reviewed alongside code that is being written to access/manipulate the data tables.

To create an SQL connection within Zend Studio, you must first change the file/directory view (left portion of the display) to the SQL view. This is accomplished by clicking on the SQL button located on the left side below the tool bar. Once the SQL view is displayed you can add an SQL source by clicking on the Add Server button, located on the left side under the SQL button. At this point, a dialog box (**Figure 6**) will prompt for the connection information. Provide the information indicated for your i5/OS system. You can test the connection information provided with the Test button. If the connection is successful, the response shown in **Figure 7** will be displayed.


Connection Test

Connection Successful

Figure 7. Response

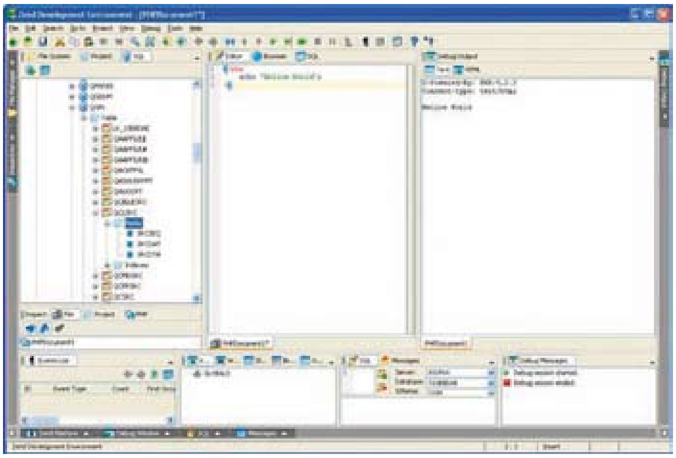


Figure 8. SQL information

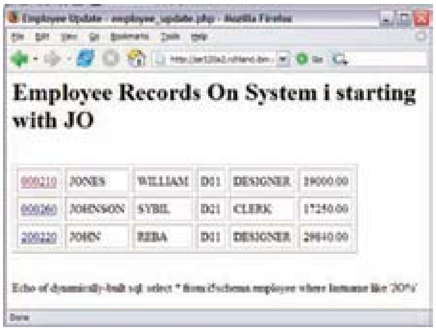
Clicking on the OK button connects to the data source and displays the SQL information shown in **Figure 8**. This interface provides a mechanism to access the field, index, stored procedure, and view definitions for the system.

Typical Uses for PHP in i5/OS

For i5/OS, PHP provides two typical uses. The first will be for development and deployment of customized applications for exploiting i5/OS resident data and procedures. These would typically be developed with the iADP stack (i5/OS, Apache, DB2, PHP) to provide a Web interface into existing i5/OS resources. Zend Core ships with the i5 Toolkit, which includes APIs for accessing such i5/OS resources as server/connection, obtaining results, commit/rollback, fetch, field information, key information, statement, errors, column/procedures, and table information.

One example of using the APIs would be to retrieve records from a table. The example in **Figure 9** uses the db2_exec API to execute an SQL select statement and the db2_fetch_array statement to obtain the resulting records from the select statement:

Executing the PHP code generates HTML output like that shown in **Figure 10**. Since the output of PHP is HTML, the PHP script can output the necessary HTML tags to generate tables and cells and then use the DB2 APIs to loop through records and populate the HTML table. In addition to the DB2 APIs, the i5 Toolkit also includes APIs for command calls, program calls, native file access, system values, and data areas. The other typical use of PHP in i5/OS will be to use the IAMP stack (i5/OS, Apache, MySQL, PHP) to leverage existing open source applications like SugarCRM, ZendCart, and phpBB on i5/OS. Future articles in this publication will discuss both of these uses in more detail.



Employee Records On System i starting with JO

000210	JONES	WILLIAM	D11	DESIGNER	19000.00
000260	JOHNSON	SYBIL	D21	CLERK	17250.00
200220	JOHN	REBA	D11	DESIGNER	29840.00

Echo of dynamically-built sql select * from i5schema.employee where lastname like 'JO%'

Figure 10. HTML output

Summary

This article has provided an overview of PHP on i5/OS including obtaining and installing the PHP engine and development environment, basic PHP syntax, and using PHP to access DB2 resources. I plan to cover more detailed aspects of PHP in i5/OS, including how to write PHP applications that leverage i5/OS resources and how to leverage existing PHP applications in i5/OS, in future articles.

Erwin Earley *is an Advisory Software Engineer in the IBM lab located in Rochester, Minnesota and heads up the Open Source Technologies Center of Competency for System i within the System i Technology Center. At that center, he provides education and enablement services for open source related technologies on System i including Linux , MySQL, and Zend's PHP. Earley currently holds certifications from Red Hat as well as the Linux Professional Institute and is a candidate for the certification with Zend's PHP.*

[Note: Erwin has also published the second in an on-going series of articles on PHP in the August 29, 2007 issue of Four Hundred Guru]