IBM i and Zend PHP for old folks – 01 getting a connection, your first script

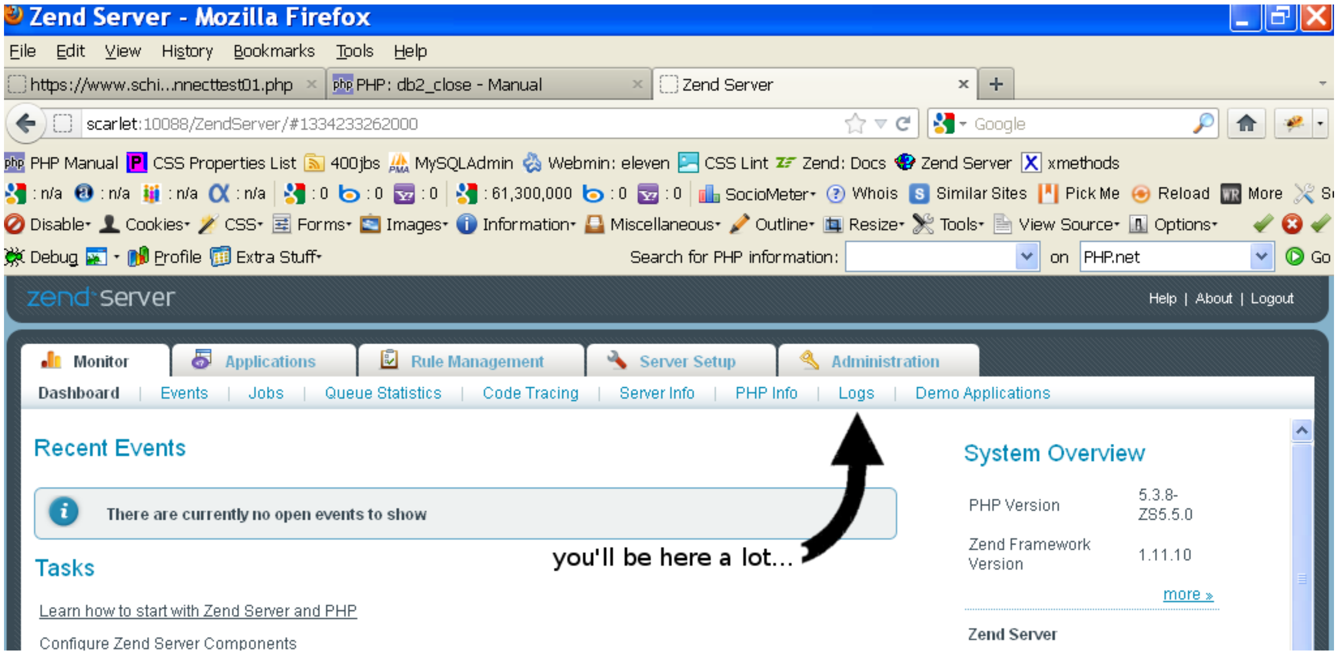
louwilkinson1@gmail.com

Setup:

This is for an experienced IBM midrange person just trying to get something going with PHP to see if they want to delve farther (you will).

First get the young kid in your shop (or your neighbor's middle school kid) to set up apache with Zend on your iSeries (AS400, IBM i...whatever you call it at your place). Next, get them to give you the password, bribing them as appropriate, to get on the ZendServer monitor/dashboard.

Go to the server and log on, click on the logs tab. Get comfortable...you'll be here a lot.


Zend Server - Mozilla Firefox
scarlet:10088/ZendServer/#1334233262000
Monitor
Applications
Rule Management
Server Setup
Administration
Dashboard | Events | Jobs | Queue Statistics | Code Tracing | Server Info | PHP Info | Logs | Demo Applications
Recent Events
There are currently no open events to show
Tasks
Learn how to start with Zend Server and PHP
Configure Zend Server Components
you'll be here a lot...
System Overview
PHP Version 5.3.8-ZS5.5.0
Zend Framework Version 1.11.10
more »
Zend Server

If you have a pretty standard install, then the actual files / scripts that you'll be serving / running will be in (from the IFS root) **/www/zendsvr/htdocs/** or anything below that. I like to make subfolders under **htdocs** to keep the various pieces and projects straight, so go ahead and make yourself a folder under **htdocs** called "**testing**". That's where will put our first script.

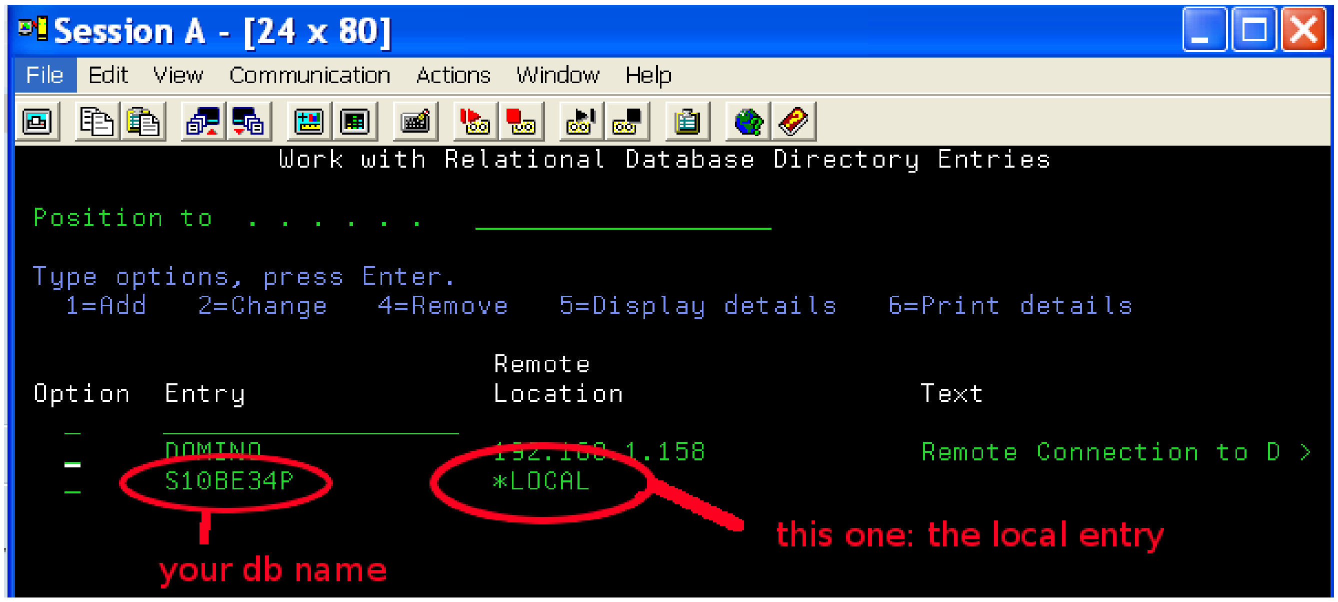
Now...open up **/www/zendsvr/htdocs/testing/connecttest01.php** (yes...01 because there'll be more) in some editor..even wordpad...we don't have to get fancy yet...this is going to be so unbelievably easy you'll wonder why it took you so long, I promise.

Your first script:

At this point we're going to start with some bad practices...but we'll clean them up before the end, so don't worry. Get **connecttest01.php** in an editor and type:

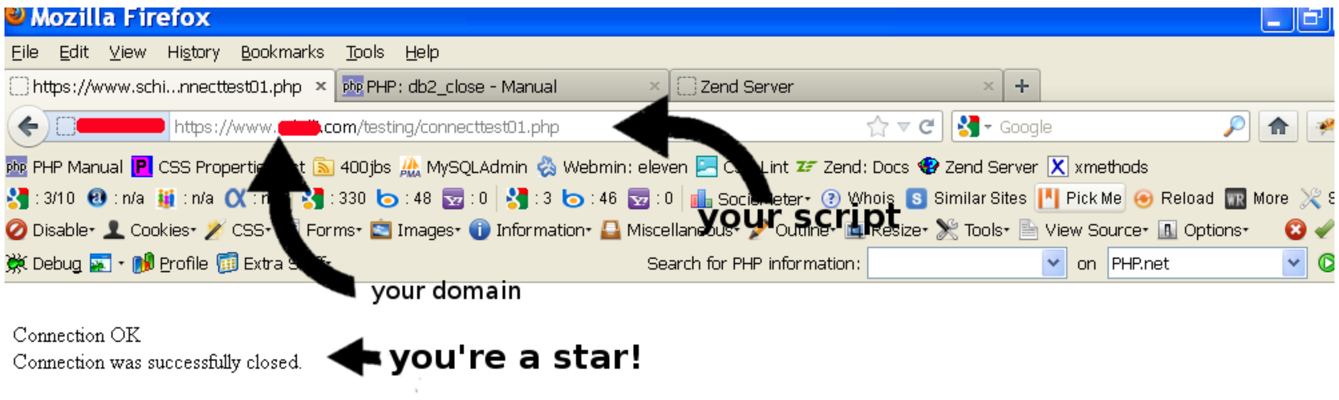
```
<?php
      define('DATABASE', "yourdb");
      define('USER', "qpgmr");
      define('PASSWORD',"qpgmrpassword");

      $conn = db2_connect(DATABASE, USER, PASSWORD);

      if (!$conn) {
            print "<br />\n no connection made";
      }
      else {
            echo "<br />\n Connection OK ";
      }

      $closed = db2_close($conn);

      if ($closed) {
            print "<br />\n Connection was successfully closed.";
      }
      else {
            print "<br />\n nope...not closed yet.";
      }
?>
```

Replace "*yourdb*" with your database name (wrkrdbdire if you need to) and "*qpgmrpassword*" with QPGMR's password.


```
                     Work with Relational Database Directory Entries

 Position to  . . . . . .   __________________

 Type options, press Enter.
   1=Add   2=Change   4=Remove   5=Display details   6=Print details

                                    Remote
 Option  Entry                      Location                 Text

   _     __________________
   _     DOMINO                     192.168.1.158            Remote Connection to D >
   _     S10BE34P                   *LOCAL
```
Session A - [24 x 80]
File Edit View Communication Actions Window Help
your db name
this one: the local entry

Bring up your browser and try to run your script.

Be sure to use your own domain name...and add **/testing/connecttest01.php**


Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://www.schi...nnecttest01.php
PHP: db2_close - Manual
Zend Server
[redacted] https://www.[redacted].com/testing/connecttest01.php
Google
your script
your domain
you're a star!

```
Connection OK
Connection was successfully closed.
```

(quick note: apache is going to serve this / run this from **htdocs**. So you only need the **/testing/connecttest01.php** after your domain name since it starts from **htdocs**)

Hopefully you see something like that that in the browser window. Does yours look the same? If so, outstanding! You're a star!

If not, then it's time to go to the ZendServer logs, flip to the end and see what they have to say about what you did wrong. My own experience is that the logs always make perfect sense AFTER you figure out the problem....but I can never tell what they're saying if I haven't figured it out yet. They do, however, do one really handy feature...they tell you the line number that you're crashing on...and that IS helpful.

Also → secret tip #1....if the line they're pointing at is perfect, then it's often the previous line that's in error or missing a semicolon or something.

Secret tip#2 → PHP is case sensitive and very, very sensitive to all those semicolons and braces, so check those first.

Moving on...

Now we're going to take the next logical step, which is to bust out the connection details in anticipation of making things a little more secure. So we're going to be creating two scripts...one that contains the secret stuff and one that just uses the secret stuff to make a connection. We're going to set it up with PHP's equivalent of a service program. So crank up your editor and create a new file in the **testing** subfolder called **ConnectFunctions.php** and key in (or copy/paste from your last file):

```
<?php

function dbsetup(){
      if (!defined('DATABASE'))
            define('DATABASE', "yourdb");
      if (!defined('USER'))
            define('USER', "qpgmr");
      if (!defined('PASSWORD'))
            define('PASSWORD',"qpgmrpassword");
};

function Connect(){
      dbsetup();
      return  db2_connect(DATABASE, USER, PASSWORD);
};

function Close($connection){
      dbsetup();
      return db2_close($connection);
}
?>
```

Notice that we have three *functions*...if you look them over, this should all make some kind of sense...Connect() and Close() both utilize the user/password *function* that we've called dbsetup(), etc. and they open a connection to your database and close it.   So this will serve as our “service program”

...now we write our 2nd file, a little test script utilizing these functions and we're going to call this one **connecttest02.php** and it goes in the **testing** folder, as well.

```
<?php

      // this says to include our "service program" file...

      include_once "ConnectFunctions.php";

      // this calls the connect function in the service program file

      $conn = Connect();

      if (!$conn) {
            print "<br />\n no connection made";
      }
      else {
            echo "<br />\n Connection OK ";
      }

      //and this calls the close function in the service program file

      $closed = Close($conn);

      if ($closed) {
            print "<br />\n Connection was successfully closed.";
      }
      else {
            print "<br />\n nope...not closed yet.";
      }

?>
```

Look familiar? Sure it does...all we've really done is split off the connection information into a "service program" we can use over and over...we allow our **connecttest02** script to "see" those functions using the **include** statement...and then call those functions as needed.

So go ahead and save both these files....hit your browser with **yourdomain/testing/connecttest02.php**

Everything cool? Great. If not, you know where to go....the logs.

Now I expect you're saying to yourself "why bother"? Because, yes, we've split off a reusable routine, but cut and paste isn't that tough. Well, brother, cut and paste and reusable procedures are where it's at! And, if that's not enough, hang tight...we're going to go protect those passwords and hide them from malicious evil-doers out in the world and it'll all begin to make more sense.

Remember: ZendServer is going to serve everything / anything in **htdocs** and below (disclaimer...this is all with a standard apache/Zend type setup...you can always do things differently and get it so screwed up that nothing works right. Some people have a gift for that...)

We want to get our user / password stuff someplace someone can't just read it off or pull it down. Here's what you do: under **/www/zendsvr** make a new folder called "**secure**". Notice that it's now at the same "level" as **htdocs**....and move the **ConnectFunctions.php** file into the new folder.

So now, in your IFS, you should have a **/www/zendsvr/secure/ConnectFunctions.php** and a **/www/zendsvr/htdocs/testing/connecttest02.php**.

With me so far? Outstanding.

Now...one little tweak to **connecttest02.php**. We have to tell it the new place to find the "service programs". So change this:

```
include_once "ConnectFunctions.php";
```

To this:

```
include_once "../../secure/ConnectFunctions.php";
```

What you've done here is to tell your script that the service program things are now in **/www/zendsvr/secure/** ...right where they should be. Your script can see them but the bad guys can't.

So reload your browser with **domainname/testing/connecttest02.php** and see if it rolls.

Very cool, eh? This ends the how to set up a secure connection between your scripts and your IBMi (or AS400 or iServer or whatever).

But → obligatory warning...the next door neighbor's kid is going to turn up her nose at this...sure, it's comfortable for you....you're used to procedural programming, functional programming...you're used to subroutines and external programs and service programs.

<u>Coming up:</u>

The next how-to is very short....but we're going to take our **ConnectFunctions.php** script and turn it into a **class**. Which is an object oriented approach...and **connecttest03.php** will use the class we define to do this exact same thing.

..you'll be surprised just how easy it is...and the neighbor's kid is going to actually nod to you, as a worthy peer, when you pull your car in the drive.

Credit where credit is due:

Obviously, the folks doing all this PHP work and the Zend folks...they've made it very, very easy for us to start exploiting this fun language.

Bryan Jamison, the “young kid” in my own shop that continues to humor me and assist with “PHP for Dinosaurs” (his term, not mine)... ...everyone needs a smirking 20something in his shop.

Bob Dusek, my all time favorite percussion player that just happens to be a pick-a-language expert who also humors me and returns every wtf email with lengthy examples and explanations (everyone needs a friend like this!)