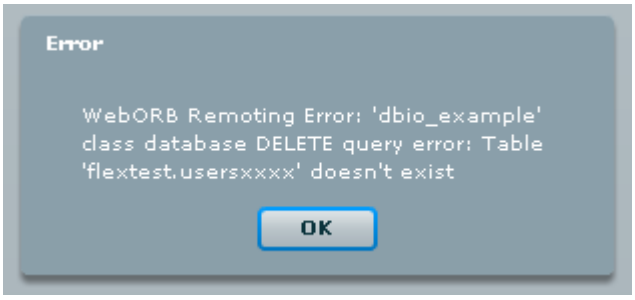
www.seaquest.com

This **simple_weborb_php** example WebORB for PHP application shows you how to message between Flex 2 clients and PHP Web server performing MySQL database I/O. The example shows how PHP generates exceptions and how Flex 2 displays them. The **simple_weborb_php** example can demonstrate this by clicking either or both exception buttons—shown below, with exception error display following.



Also see the weborb blog at the URL http://blog.themidnightcoders.com for further comments about the **simple_weborb_php** example application.

Note: This install document provides the necessary information to deploy the "simple_weborb_php" example hosted at Google Code **adobe-php-sdk**: http://code.google.com/p/adobe-php-sdk
We assume the reader of this document is familiar with using Adobe Flex 2 development tools and has experience in administration and deployment of AMP (Apache, MySQL, PHP) system elements. For example, we do not provide the "how to" details about installing MySQL databases and tables.

Follow the these steps to install the **simple_weborb_php** example application.

1. First install WebORB for PHP 1.1 or newer (1.0 will not work) into your Web server root. Expand the distribution archive into the root of your web server.

   See the URL— http://themidnightcoders.com/weborb/php/gettingstarted.htm —for more WebORB for PHP install layout details.

   Once the WebORB for PHP solution is installed, you should see the WebORB for PHP directory structure as shown at the top of the following page.

2. Run the "Deploy 'simple_weborb_php' Example.jse" JavaScript script.

   Note: Before running the JavaScript script, using your favorite text editor, first open the script and follow the data path configuration discussions listed in the JavaScript file.

   After the WebORB for PHP **simple_weborb_php** example is installed, by running the above JavaScript script, you should see the directory structure as shown on the next page.

3. Using your preferred MySQL client, create the required "flextest" example database by running "flextest.sql" SQL schema file located in your Web site "Services" folder. See directory structure, as shown on the next page, for location of the "flextest.sql" SQL schema file.

   Note: If you have previously installed the AMFPHP example titled "simple_amfphp" from at the Google Code **adobe-php-sdk** example repository, then you already have the MySQL "flextest" database installed. It is not necessary to perform this step #3.

4. Add the following XML element entry to the file "remoting-config.xml":

```
<destination id="SimpleWebOrbPHPDestination">
  <properties>
    <source>adobe_php_sdk.simpleWebOrbPHP.dbio_example</source>
  </properties>
</destination>
```

   To the file "remoting-config.xml" located at:
   web_server_root/Weborb/WEB-INF/flex/remoting-config.xml

   Note: You can copy and paste the above XML element from the "Manual edit to remoting-config.xml" file located in the same folder as your "remoting-config.xml" file.

"simple_weborb_php" installs are shown in **bold red** in the following WebORB for PHP install path hierarchy diagram

```
+-web server root
|
+--/Services — Contains deployed 'remotable' PHP classes
|     |
|     +--/adobe_php_sdk
|          |
|          +--/simpleWebOrbPHP
|               |
|               +-dbio_example.php —
|               |
|               +-flextest.sql      — SQL to create "flextest" example database.
|               |
|               +-test-dbio_example_object.php — PHP test stub for direct testing
|                                                 of class methods in:
|                                                 'dbio_example.php'
+--/Examples — Contains examples shipped with WebORB
|
+--/Weborb    — Contains configuration, log and WebORB for PHP source code.
      |
      |
      +--weborb-config.xml — Contains a reference to the /Services folder, as well
      |                      as other important WebORB configuration data.
      |
      |
      +--weborb-log.txt — The WebORB log file.
      |
      +--/WEB-INF
           |
           +--/flex
                |
                +--remoting-config.xml — Lists all deployed Flex RPC services.
                |
                +--Manual edit to 'remoting-config'.xml — Required manual edit
                |                                         addition to "remoting-config.xml" file.
                |                                         See step #4, preceding for details.
                |
                +--services-config.xml — Configures Flex RPC endpoint.
```

5. Create a Flex 2 project following the instructions at:
   http://themidnightcoders.com/weborb/php/gettingstarted.htm
   from the section: **GETTING STARTED - CREATING A FLEX APPLICATION**

   Name the Eclipse project "simple_weborb_php".

6. Replace the new project generated "simple_weborb_php.mxml" file with the one in this Zip file package.

7. That's it. From your **Eclipse IDE**, **Compile**, **Run**, and enjoy "simple_weborb_php".

Handling PHP Generated Exceptions in Flex 2

The "simple_weborb_php" example handles and displays MySQL database I/O exceptions rather nicely. MySQL database I/O exceptions are displayed in an Alert dialog. What's neat is that the example is displaying the exact exception message generated by the "mysqli" database access PHP extension. While the "mysqli" generated error messages are much too wordy and technical for general user population consumption, this "simple_weborb_php" example does show what can be provided in the way of detailed I/O system information from PHP can be displayed by Flex 2.

You can rather easily see learn about and view the database I/O exceptions in the Flex 2 Alert dialog, as follows.

From your Eclipse IDE, run the "simple_weborb_php" example. Leave the example Web page open.

Next, using your favorite text editor, or the Eclipse IDE, open the "dbio_example.php" file at:
web_server_root/Services/adobe_php_sdk/simpleWebOrbPHP

One line at a time, change the following "dbio_example.php" entries and save the file:

```
From:  Define('DATABASE_SERVER', 'localhost');
  To:  Define('DATABASE_SERVER', 'localhostxxx');

From:  Define('DATABASE_USERNAME', 'flext');
  To:  Define('DATABASE_USERNAME', 'flextxxx');

From:  Define('DATABASE_PASSWORD', 'p@ssword');
  To:  Define('DATABASE_PASSWORD', 'p@sswordxxx');

From:  Define('DATABASE_NAME', 'flextest');
  To:  Define('DATABASE_NAME', 'flextestxxx');
```

Then, with each, one at a time line change, per above, refresh the "simple_weborb_php" example Web page. You will see an Flex 2 Alert dialog displaying the PHP internally database I/O error.

The error messages, while similar in context, are all different. Restore the current change and edit another line, per above, and refresh the "simple_weborb_php" example Web page.

What is the "test-dbio_example_object.php" file for?

Initial debugging of Flex 2 to/from WebORB for PHP (PHP AMF Gateway) messaging can be a real pain to debug. If your messaging is not working, is it the transmitter or receiver? There's no need to be debugging messaging problems on both client and server simultaneously. One way to reduce the PHP messaging debugging is to create a PHP test stub to locally evaluate proper operation of your from WebORB for PHP messaging object. The enclosed file "test-dbio_example_object.php" is a reference example of a test stub to run against the "dbio_example.php" PHP 5 object to locally test proper PHP messaging operation.

I strongly recommend always stand-along testing of your WebORB for PHP objects methods before attempting messaging via Flex 2. Use the enclosed test stub to develop test messaging for your WebORB for PHP objects.