MySQL Integration

Data can be retrieved from various sources. In almost all cases data will come from an online database but you can imagine a lot of alternative sources like flat files, XML, scripts,.. You‘ll find here some example on how to interact with the pData class.

Retrieving data from MySQL

MySQL is a powerfull open source database engine. This is the perfect companion for the pChart library but of course you can use any other engine! A lot of databases connectors are shipped with PHP, the syntax to use will depends of the one you‘ll choose. The following example shows how to retrieve data from a MySQL database. In this example we‘ll use a table called **measures** containing history data for **temperature** and **humidity**.

We want here to extract the measured values and the associated timestamp to build an history chart. First step will be to create your pData object and connect to your database :

1./\* Include the pData class \*/

2.include("class/pData.class.php");

3.

4./\* Create the pData object \*/

5.$MyData = new pData();

6.

7./\* Connect to the MySQL database \*/

8.$db = mysql\_connect("localhost", "dbuser", "dbpwd");

9.mysql\_select\_db("pchart",$db);

Now we want to extract each column and bind it to one data serie. This can be done in many ways, the simplest (and not optimised) would be :

01./\* Build the query that will returns the data to graph \*/

02.$Requete = "SELECT \* FROM `measures`";

03.$Result  = mysql\_query($Requete,$db);

04.while($row = mysql\_fetch\_array($Result))

05. {

06.  /\* Get the data from the query result \*/

07.  $timestamp   = $row["timestamp"];

08.  $temperature = $row["temperature"];

09.  $humidity    = $row["humidity"];

10.

11.  /\* Save the data in the pData array \*/

12.  $myData->addPoints($timestamp,"Timestamp");

13.  $myData->addPoints($temperature,"Temperature");

14.  $myData->addPoints($humidity,"Humidity");

15. }

But you must prefer to reduce the number of calls to the pData class in order to greatly improve your script speed. As the **addPoints** method can handle arrays, it would be a nice way to push all the data in a array and pass it to the pData class in only one call :

01./\* Build the query that will returns the data to graph \*/

02.$Requete = "SELECT \* FROM `measures`";

03.$Result  = mysql\_query($Requete,$db);

04.$timestamp=""; $temperature=""; $humidity="";

05.while($row = mysql\_fetch\_array($Result))

06. {

07.  /\* Push the results of the query in an array \*/

08.  $timestamp[]   = $row["timestamp"];

09.  $temperature[] = $row["temperature"];

10.  $humidity[]    = $row["humidity"];

11. }

12.

13./\* Save the data in the pData array \*/

14.$myData->addPoints($timestamp,"Timestamp");

15.$myData->addPoints($temperature,"Temperature");

16.$myData->addPoints($humidity,"Humidity");

Now we want to use the data of the timestamp column as the abscissa labels. We also want to display it in a readable way :

1./\* Put the timestamp column on the abscissa axis \*/

2.$myData->setAbscissa("Timestamp");

As we want to chart here to values of different units, we must create a second axis and associate it the humidity :

1./\* Associate the "Humidity" data serie to the second axis \*/

2.$myData->setSerieOnAxis("Humidity", 1);

3.

4./\* Name this axis "Time" \*/

5.$myData->setXAxisName("Time");

6.

7./\* Specify that this axis will display time values \*/

8.$myData->setXAxisDisplay(AXIS\_FORMAT\_TIME,"H:i");

Now we can make some makeup by specifying the units and axis names :

1./\* First Y axis will be dedicated to the temperatures \*/

2.$myData->setAxisName(0,"Temperature");

3.$myData->setAxisUnit(0,"°C");

4.

5./\* Second Y axis will be dedicated to humidity \*/

6.$myData->setAxisName(1,"Humidity");

7.$myData->setAxisUnit(0,"%");