C/PHP Database API

Contents

C Side – ArduinoComm.c 3

Syntax 3

PHP Side – curl\_insert.php 4

PHP Side – database.php 5

Database 5

SRQUERY 5

insert 5

cleartable 5

## C Side – ArduinoComm.c

Certain functions in ArduinoComm.c require database insertion capabilities. To accomplish this, an external process is created to push a string via POST to a dedicated webpage.

### Syntax

The Raspberry Pi has the capability to use <unistd.h>, a header containing unix standard function definitions. This provides the *exec* function, which lets the Raspberry Pi start an external child process. For the purposes of the API, we start cURL, a command line tool, as an external process. Suppose an sql insertion we wish to execute is given as a string *query*; then we can send this string to POST on a PHP webpage via the following command line command:

curl -s -X POST --data-urlencode “string=*query*” *webpage*

In C, this is accomplished via *exec*:

exec(“curl.exe”, “curl.exe”, “-s”, “-X”, “POST”, “--data-urlencode”, “\”string=”+*query*+”\””, *webpage*)

## PHP Side – curl\_insert.php

A simple website that is not accessible to users is all that is needed to handle cURL POST requests from the command line or a C interface. With our framework defined in the previous section, this website need only check for data in the specific POST entry, and then submit it as a query to the database.

if (isset($\_POST['string'])) {

echo($\_POST['string']);

$db->genericQuery($\_POST['string'], 1);

}

unset($\_POST['string']);

Note the function genericQuery. This is defined in the PHP Client Side API as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Return Type | Name | Arguments | Description |
| void | genericQuery | $sql – a properly defined MySQL string to execute query with no return  $num – the number of attempts to execute the query before quitting | A function that attempts to query a MySQL string several times until success. The function is used for queries that return void, like insertions and deletions |

On every cURL execution, an instance of this website is created, so access to the website is safe and the database is queried uniquely each time.

## PHP Side – database.php

## Database

The constructor for a Database object.

Identifiers: public function

Example:

$connection = new Database(); //Declaring a new database object.

## SRQUERY

Function that returns the response of a query to the server that returns a single value.

Identifiers: public function

Parameters: A MySQL query as a string

Returns: A numerical value representing the result of the query

Example:

$sql = “SELECT COUNT(\*) FROM `table`”; //Gets the number of all entries in a table

$result = $connect->SRQuery($sql); //Calls function on Database object and stores value

## insert

Function that inserts a specified date a specified number of times as an entry in the traffic database.

Identifiers: public function

Parameters: A “Y-m-d hh:mm:ss” formatted date or “default” for the current time

The number of times to insert the date

Example:

$connect->insert(“2018-04-10 12:45:01”,1);

$connect->insert(“default”,2);

## cleartable

Function that resets all traffic data.

Identifiers: public function

Example:

$connect->clearTable();