Small DB class

```
<?php

/**
 * Generic database connection class
 *
 * Required: PHP 5.6
 * Based on: https://codeshack.io/super-fast-php-mysql-database-class/
 *
 * Example usage:
 *  include_once("db.php");
 *  $this->db = new DB($dbhost, $dbuser, $dbpass, $dbname);
 *  $this->db->query("INSERT INTO users (name, birthdate) VALUES (?, ?)", array("John Doe",
"1970-01-01"));
 *  $id = $this->db->lastInsertId();
 *  $user1 = $this->db->query("SELECT name FROM users WHERE id = ?", $id)->getRow();
 *  $all_users = $this->db->query("SELECT name FROM users")->getResult();
 */
class DB {
	protected $connection;
	protected $query;
	protected $show_errors = true;
	protected $query_closed = true;
	protected $query_count= 0;

	public $error = false;

	public function __construct($dbhost = "localhost", $dbuser = "root", $dbpass = "", $dbname =
"", $charset = "utf8") {
		$this->connection = new mysqli($dbhost, $dbuser, $dbpass, $dbname);

		// Check connection
		if ($this->connection->connect_error) {
			$this->error("Database connection failed: " . $this->connection->connect_error);
		}
```

```
    $this->connection->set_charset($charset);
  }

  public function __destruct() {
    $this->close();
  }

  public function affectedRows()
  {
    return $this->connection->affected_rows;
  }

  public function close()
  {
    $this->connection->close();
  }

  private function error($error)
  {
    $this->error = $error;

    if ($this->show_errors) {
      exit($error);
    }
  }

  public function getResult() {
    $return = array();
    $mysqli_result = $this->query->get_result();
        if (!$mysqli_result) {
            return false;
        }

    while($row = $mysqli_result->fetch_object()) {
      $return[] = $row;
    }
    return $return;
  }

  public function getRow()
  {
```

```
    return $this->query->get_result()->fetch_object();
  }

  private function getType($var)
  {
    if (is_string($var)) { return 's'; }
    if (is_float($var)) { return 'd'; }
    if (is_int($var)) { return 'i'; }
    return 'b';
  }

  public function lastInsertId()
  {
    return $this->connection->insert_id;
  }

  public function numRows()
  {
    $this->query->store_result();
    return $this->query->num_rows;
  }

  public function query($query, ...$args)
  {
    if (!$this->query_closed) {
      $this->query->close();
    }

    if ($this->query = $this->connection->prepare($query)) {
      // Check for arguments
      if (sizeof($args) > 0) {
        // $args can be variable length array or regular array.
        // Check which one it is.
        if (is_array($args[0])) {
          // An array was passed. Move it one level up in the array, so it's the same array as a
variable length array
          $args = $args[0];
        }

        $types = "";    // Types of the prepared statements as a string
        $prepared_statement_values = array();
```

```
                foreach ($args as $key => &$arg) {
                    $types .= $this->getType($args[$key]);
                    $prepared_statement_values[] = &$arg;
                }

                // Add the types to the front of the array
                array_unshift($prepared_statement_values, $types);

                // Call the bind_param-function the create the prepared statements
                call_user_func_array(array($this->query, 'bind_param'), $prepared_statement_values);
            }

            // Execute the query
            $this->query->execute();

            // Was there an error?
            if ($this->query->errno) {
                $this->error("Unable to process MySQL query: " . $this->query->error);
            }

            // Close the query and add to the query count
            $this->query_closed = false;
            $this->query_count++;
        } else {
            $this->error("Unable to prepare MySQL statement: " . $this->connection->error);
        }

        return $this;
    }
}
```

Created 2023-01-31 15:10:33 UTC by Jelle
Updated 2024-10-18 08:51:07 UTC by Jelle