

Neue Klassen mit Datenbankzugriff

Wir wollen unserem Projekt nun weitere Klassen hinzufügen, von denen einige Datenbankzugriff benötigen. Die jeweiligen Operationen auf der Datenbank sollen aber in den jeweiligen Klassen gekapselt sein.

Um dieses Ziel zu erreichen, *reduzieren* wir zunächst die Funktionalität unserer `mysqlDb`-Klasse darauf, eine ¹⁾ Datenbankverbindung herzustellen und ein entsprechendes PDO-Objekt, das diese Verbindung repräsentiert, zurückzuliefern, welches dann innerhalb der anderen Klassen für den Datenbankzugriff verwendet werden kann.

Die neue DB Klasse

Passen die Datenbank Klasse in der Datei `mysqlDb.class.php` folgendermaßen an:

```
class mysqlDb {

    static public $connection = null;
    private $dbobject;

    /**
     * Constructor - connect to db, set handle attribute
     *
     * @param string      $dbusername  DB username
     * @param string      $dbpassword  DB password
     * @param string      $dbname      Database to connect to
     * @param string      $host        Database host to connect to
     * (optional)
     *
     */

    private function __construct($dbusername, $dbpassword, $dbname, $host )
    {

        try {
            self::$connection = new PDO("mysql:host=$host;dbname=$dbname",
            "$dbusername", "$dbpassword");
            self::$connection->setAttribute(PDO::ATTR_ERRMODE,
            PDO::ERRMODE_EXCEPTION);

        } catch ( PDOException $e ) {
            echo 'Verbindung zur Datenbank fehlgeschlagen: ' .
            $e->getMessage();
            exit();
        }

    }

}
```

```
public static function getConnection($dbusername, $dbpassword, $dbname,
$host ) {
    // Wenn das Objekt noch keine PDO-Instanz hat
    // wird eine erzeugt
    if(self::$connection === null) {
        $dbobject = new mysqli($dbusername, $dbpassword, $dbname,
$host);
    }

    //Und das PDO-Objekt zurückgeben
    return self::$connection;
}

// Das Klonen dieser Instanz verhindern.
private final function __clone () {}
}
```

Anmerkungen:

- Die Schreibweise mit dem Doppel-Doppelpunkt ermöglicht den Zugriff auf statische Methoden und Felder, ohne ein Objekt der Klasse zu besitzen. Das benutzen wir, um in weiteren Klassen ein Datenbank-Handle zu erhalten.
- Die Klasse selbst ist so konstruiert, dass sie nur eine Datenbankverbindung aufbaut. Wenn noch keine Datenbankverbindung existiert, wird ein neues mysqli-Objekt samt Datenbankhandle erzeugt, andernfalls lediglich das Handle zurückgegeben.
- Außerdem verhindern wir, dass die Instanz geklont werden kann.

Benutzer

Nun wenden wir uns der Benutzer-Klasse zu. Diese hat zunächst vor allem die Aufgabe, zu überprüfen, ob der angemeldete Dokuwiki Benutzer bereits einen Eintrag als Blog-Benutzer in der Datenbank hat. Wenn nicht muss der Benutzer angelegt werden.

Erstelle im Unterverzeichnis class deines Plugins eine Datei `bloguser.class.php` und definiere darin die Klasse `bloguser`:

```
<?php

class bloguser {

    protected $db;           // Das DB-Handle

    //Konstruktor
    public function __construct ($dbusername, $dbpassword, $dbname,
$host="localhost" )
    {
```

```
// PDO Connection erzeugen/holen und als
// Objektattribut "speichern". Damit werden DB Zugriffe möglich.
$this->db = mysqlldb::getConnection($dbusername, $dbpassword, $dbname,
$host);
}

public function checkBlogUser ($dwusername) {
    print "Überprüfe, ob der Dokuwiki-Benutzer " .
$_SERVER['REMOTE_USER'] . " schon in der Datenbank angelegt ist.";
}
}
?>
```

Das ganze kann man dann etwa so in der syntax.php einbinden:

```
[...]
public function render($mode, Doku_Renderer $renderer, $data)
{
    // Wir rendern nur HTML, sonst nix!
    if ($mode !== 'xhtml') {
        return false;
    }

    // Breche die Verarbeitung ab, wenn der Benutzer nicht angemeldet
ist.
    if ( ! isset($_SERVER['REMOTE_USER']) ) {
        $renderer->doc .= "Um das Blog zu benutzen, müssen Sie sich
anmelden";
        return true
    }
    // get settings
    $dbpasswd = $this->getConf('dbpasswd');
    $dbname = $this->getConf('dbname');
    $dbhost = $this->getConf('dbhost');
    $dbtable = $this->getConf('dbtable');
    // make db connection
    $bloguser = new bloguser($this->getConf('dbusername'), $dbpasswd,
$dbname, $dbhost);
    // Prüfe, ob es den angemeldeten Benutzer schon in der DB gibt, lege
ihn an wenn
    // nicht. Liefer die bloguserID des Nutzers zur weiteren Verwendung
zurück
    $bloguser->checkBlogUser($_SERVER['REMOTE_USER']);
}
```

[...]



(A1)

← Schritt 1 Schritt 3 →

1)

und nur eine

From:
<https://info-bw.de/> -

Permanent link:
https://info-bw.de/faecher:informatik:oberstufe:datenbanken:projekt:dokuwiki_plugin:microblogging:step02:start?rev=1624291251

Last update: **21.06.2021 16:00**

