Aufgabenblatt 4

Webtechnologien
Wirtschaftsinformatik
WS 2020/21

Lernziele

Es wird folgendes gelernt:

- Objekte/Klassen
- Wrapper- und Factory-Klassen
- MySQL
- MySQL-Zugriff per PHP (PDO)
- Erstellung und erstmaliges Befüllen einer Datenbank

Zu den genannten Zielen und Methoden gibt es eine Liste von Links auf weitere Informationen.

Klassen in PHP und MySQL – 3 Punkte

In diesem Aufgabenblatt lernen Sie den Zugriff auf eine MySQL-Datenbank mittels Klassen in PHP. Dabei werden verschiedene Klassen so benutzt, dass eine möglichst große Abstraktion, und damit eine universelle Benutzbarkeit und gute Wartbarkeit erreicht wird. In dem Teil der Folien über PHP werden die benötigten Wrapper- und Factory-Klassen beschrieben und erläutert. Sehen Sie sich dazu den Foliensatz bzw. das Video zur Vorlesung an.

In der Datenbank wird eine Tabelle mit ein paar Attributen angelegt. Am besten sind dies Produkte zusammen mit Preisen, z.B. Brötchen, Butter, Weißbrot, Emmentaler Käse, Bonbons, Schokolade mit den Preisen: 0.20, 1.50, 3.00, 2.50, 1.00 und 1.80 EUR (das ist nur ein Vorschlag, Sie können die Preise ruhig etwas niedriger wählen).

Die Aufgabe besteht nun darin, diese Tabelle zu füllen und auszugeben. Dazu wird nicht HTML, sondern die Konsole benutzt (echo-Funktion in PHP), d.h. die Lösung läuft nicht über den Web-Server, sondern direkt innerhalb der netbeans- bzw. eclipse-IDE – ohne Generierung von HTML durch PHP.

Und nun die einzelnen Schritte:

1. Es wird die Datenbank angelegt, z.B. mit mysqladmin, mysql workbench oder phpMyAdmin. Dann wird die Tabelle samt Attributen definiert. Hierzu empfiehlt es sich die einzelnen Felder samt Typen vorher auf Papier zu definieren. Es sollte noch ein eigener Primary Key benutzt werden, typischerweise ein Integer-Wert.

2. In Excel bzw. Calc wird nun der erste Tabelleninhalt definiert; hier in diesem Beispiel sind es nur Strings, Float- bzw. für den Schlüssel Integer-Werte. Dann werden die Tabellen als CSV – bitte das tatsächlich benutzte Format beachten – als Dateien exportiert und mit Hilfe des Programms mysqlimport bzw. LOAD DATA im mysql-Interpreter in die Datenbank gebracht. Selbstverständlich ist es auch möglich dies alles zu Fuß in phpMyAdmin einzutippen, was aber recht aufwendig ist und lange dauert.

3. [Abgabe] Jetzt geht es an den PHP-Teil. Als erstes definieren Sie ein Interface mit den Methoden open(), insert($record), query($name, $string), delete($name,$string) und close().

4. Sie definieren die Klasse DB, die den Zugriff auf MySQL realisiert und das Interface implementiert. Diese Klasse realisieren Sie mit Hilfe von PDO – mysqli oder mysql sind nicht in Ordnung. Dabei fehlt Ihnen der DSN-String mit dem Datenbanktyp, der Host-Adresse und dem Namen der Datenbank. Diese Informationen übergeben Sie als Parameter dem Konstruktor der Klasse DB.

   Nun zu den Methoden der Klasse DB:

   insert($record): $record ist ein PHP-Hash, dessen Elementnamen die Spaltennamen der Tabelle und dessen Werte eben die Werte einer Zeile in der Tabelle sind. Es wird eine neue Zeile (record) in die Tabelle eingefügt. Diese Idee ist im Foliensatz über PDO beschrieben.

   query($name, $string): führt ein select ... where $name='$string' aus, wobei $name der Name einer Spalte ist, z.B. Name des Produkts, und liefert die erste Tabellenzeile als Hash (wie insert() diesen Hash als Parameter erwartet), die der Bedingung genügt. Wenn die Bedingung mehrere Zeilen erfüllen, so wird immer nur die erste Zeile geliefert.

   delete($name, $string): macht dasselbe wie query(), nur wird die entsprechende Tabellenzeile gelöscht. Wenn die Bedingung mehrere Zeilen erfüllen, so wird immer nur die erste Zeile gelöscht.

   open() und close() öffnen und schließen die DB-Verbindung.

5. Nun wird das Ganze mit der Klasse PHPDB wiederholt, die auch das Interface realisiert und dieselben Daten per Konstruktor übergeben bekommt. Nur dass nun keine Datenbank-Tabelle sondern ein PHP-Hash die

Datenbank im RAM simuliert.

open() und close() retten das Hash in eine Datei bzw. stellen den Hash wieder her. Beim Retten in close() wird der Hash serialisiert und über File I/O in eine festgelegte Datei geschrieben; beim Wiederherstellen in open() wird von dieser Datei gelesen und der Wert so deserialisiert, dass der alte Wert des PHP-Hashs wieder vorhanden ist.

Einen Sonderfall bildet der erstmalige Beginn, denn dann ist keine Datei mit alten Werten vorhanden. In open() wird dies geprüft und ein leerer Hash in einer Variablen angelegt. Erst beim close() wird der aktuelle Hash in der Variable mittels Serialisierung in eine Datei geschrieben. Das zweite open() findet dann die Datei und liest sie ein wie oben beschrieben.

6. Nun wird eine Factory-Klasse geschrieben, die die beiden obigen Wrapper-Klassen instanziiert und über das Interface einem Benutzer zur Verfügung stellt. Ein Beispiel ist im Foliensatz beschrieben.

   Die PDO-Version verhüllt die Benutzung von MySQL, während die PHP-Version die Benutzung einer Datei zusammen mit einem Hash versteckt.

7. [Abgabe] Nun zum "Hauptprogamm": Dieses fügt 2-3 neue Zeilen in die Tabelle ein und listet auf der Konsole Teile des Inhalts auf. Das erfolgt ohne HTML. Mit der Funktion printf() können Sie den Output etwas besser gestalten. Dabei sollen beide Wrapper-Klassen sowie alle Funktionen zum Einsatz kommen. Auch sollen mehrmals die Daten eröffnet und geschlossen werden, so dass dadurch die Korrektheit der Realisierung erkennbar wird.

Kleiner Hinweis für die Klausur: Für den Fall, dass in der Klausur nach Wrapper- oder Factory-Klassen gefragt wird, sei gesagt, dass beide Art vollkommen unabhängig von einander sind, d.h. Wrapper-Klassen ohne Factory-Klassen sind genauso möglich, wie auch umgekehrt. Der Sinn einer Wrapper-Klasse besteht im Herstellen einer einheitlichen Schnittstelle – hier definiert durch ein Interface -; der Sinn einer Factory-Klasse besteht darin, ein Geflecht von Objekten aufzubauen und dadurch den *Prozess* des Aufbaus zu verdecken.

Links

- https://de.wikipedia.org/wiki/Adapter_(Entwurfsmuster)
- https://de.wikipedia.org/wiki/Fabrikmethode
- https://www.tutorialspoint.com/design_pattern/factory_pattern.htm
- https://www.w3schools.com/php/php_mysql_intro.asp
- https://www.php-einfach.de/mysql-tutorial/uebersicht-sql-befehle/

- https://www.php-einfach.de/mysql-tutorial/daten-ausgeben/
- http://php.net/manual/de/book.pdo.php
- http://php.net/manual/de/function.fread.php
- http://php.net/manual/de/language.oop5.serialization.php
- http://www.mysqltutorial.org/import-csv-file-mysql-table/

Abnahme und Bewertung

Das Ergebnis des Schrittes 7 wird per Email abgegeben. Dabei muss die gesamte "Site" als zip-Datei also das netbeans-Projekt sowie die Datenbank in einer entladenen Form abgegeben werden. Bei Abgaben per Email muss die Lösung auf den Rechnern der Dozenten im ersten Anlauf laufen.

Die Definition des Interface sowie dessen Benutzung (Schritt 2) gehört zur Abgabe.

Zur Abnahme gehört auch ein Snapshot-Dump der Demonstration von:

- Alle Funktionen mit der MySQL-Datenbank
- Alle Funktionen mit der RAM-Simulation

Am besten ist es, wenn das "Hauptprogramm" diese Routinen hintereinander aufruft.

Es sind max. 3 Punkte zu erreichen; fehlt etwas oder ist es nicht in Ordnung, so werden Punkte abgezogen bzw. die Lösung nicht anerkannt. Wichtig ist hierbei, dass mit PDO, einem Interface und mehreren Klassen implementiert wurde.

Es müssen die in den vorherigen Aufgabenblättern vorgestellten Regeln über CSS, Layout und HTML4/5 eingehalten werden.