PHP und MySQL

Im Gegensatz zu clientseitigen Scriptsprachen wie CSS und JavaScript, die auf dem lokalen PC ausgeführt werden, werden PHP und MySQL ***serverseitig*** *ausgeführt und der User bekommt die fertig abgearbeiteten Ergebnisse gesendet. PHP und MySQL laufen also nicht auf den lokalen Rechnern der User, sondern auf dem Webserver, dem Computer des Anbieters, bzw. dessen Providers.*

Das ist einerseits gut, weil PHP vom User nicht installiert werden muss und somit eine große Kompatibilität gegeben ist: Wenn ein PHP-Script funktioniert, dann funktioniert es in jeder Umgebung. JavaScripts dagegen müssen auf dem lokalen PC ausgeführt werden und hier kommt es vor, dass User JavaScript deaktiviert haben (JavaScript kann den lokalen PC schädigen) oder veraltete Browser benutzen. Außerdem wird der eigentliche PHP-Code nicht zum User übertragen, sondern nur dessen Output – damit ist er relativ leicht vom Programmierer/ Urheber zu schützen. Zudem werden nur die Daten übertragen, für die der User sich gerade interessiert: Er muss zum Beispiel nicht den kompletten Katalog der auf ebay angebotenen Waren herunterladen, sondern bekommt nur diejenigen Waren angezeigt, nach denen er gesucht hat – und das fast in Echtzeit.

Andererseits aber hat das auch Nachteile: Für den Entwickler bedeutet dies, dass er seine Site über einen Webserver aufrufen muss – im Idealfall muss man also einen Webserver installieren (Empfehlung: den kostenlosen, portablen XAMPP verwenden).

Außerdem weiß ein PHP-Script bei der Ausführung nichts über den lokalen PC des Users (dessen Bildschirmauflösung, dessen Browser und Browserversion …) und kann entsprechend auf diese Gegebenheiten nicht reagieren. Es spricht aber nichts dagegen, PHP mit JavaScript und CSS zu kombinieren.

PHP

Eine PHP-Datei ist eine ***Textdatei*** *und kann mit jedem Texteditor geöffnet und editiert werden.*

Der PHP-Quellcode wird nicht kompiliert (wie zum Beispiel der Quellcode eines C+±Programms, der vom Compiler in ein lauffähiges Programm, zum Beispiel in eine exe-Datei, umgewandelt wird), sondern sobald eine PHP-Seite über den Webserver aufgerufen wird, life an einen Interpreter geschickt, der die ***HTML-Ausgabe*** *erzeugt, die dann über den Webserver an den Browser des Users geschickt wird. Man kann eine PHP-Datei also ganz einfach ändern, speichern – und schon kann die aktuelle Version vom User aufgerufen werden.*

Eine PHP-Datei ist einer HTML-Datei sehr ähnlich, weil sie am Ende als HTML-Datei an den Browser des Users geschickt wird.

Praxis

Man kann eine PHP-Datei zwar als File in den Browser laden, dann aber wird der PHP-Code nicht interpretiert, sondern wird vom Browser als bloßer Text angezeigt. Wenn man eine PHP-Datei richtig sehen will, muss man sie über einen Webserver aufrufen. Das ist immer dann der Fall, wenn man eine Homepage über das Internet abruft. Bei der Erstellung einer Site aber ist das sehr aufwändig, wenn man die Seiten, die man gerade erzeugt hat, erst via FTP auf den Webspace eines Internet-Providers hochladen muss, bevor man sie ausprobieren kann. Deshalb werden PHP-Entwickler einen lokalen Webserver installieren (zum Beispiel den kostenlosen Apache über das Installationstool XAMPP, das gleichzeitig auch PHP und MySQL installiert). Wenn ein Webserver auf dem lokalen Computer läuft, dann ist er gewöhnlich über **localhost** *und die IP-Adresse* 127.0.0.1 *zu erreichen (einfach als URL in die Adresszeile des Browsers eingeben). Alle Daten, die in dem htdocs-Verzeichnis des Apache-Webservers abgelegt sind, werden von diesem über diese Adressen zur Verfügung gestellt. Die URL* http://localhost/meinPHPScript.php *würde also die Seite* meinPHPScript.php *im Verzeichnis htdocs aufrufen. Entsprechend würde die URL* http://localhost/VZTest/meinPHPScript.php *jene Seite im Verzeichnis htdocs/VZTest aufrufen.*

Damit der PHP-Interpreter weiß, dass in einer PHP-Datei PHP-Code ist, muss diese Datei die Endung .php besitzen (kann in der httpd.conf des Apache geändert werden).

Innerhalb einer PHP-Datei müssen die Bereiche, die vom PHP-Interpreter geparst werden, durch PHP-Tags als solche markiert werden. Am Anfang eines PHP-Bereichs steht: <?php*, an dessen Ende:* ?>

Dazwischen wird jede Zeile vom PHP-Interpreter geparst. Hier ist also ausschließlich PHP-Code erlaubt. Außerhalb dieses Tags dagegen ist kein PHP-Code erlaubt, sondern nur HTML, CSS, JavaScript …

Kleine PHP-Einfügungen kann man auch in dieser Form eingeben: <?php echo "<p>Hello World</p>"?>

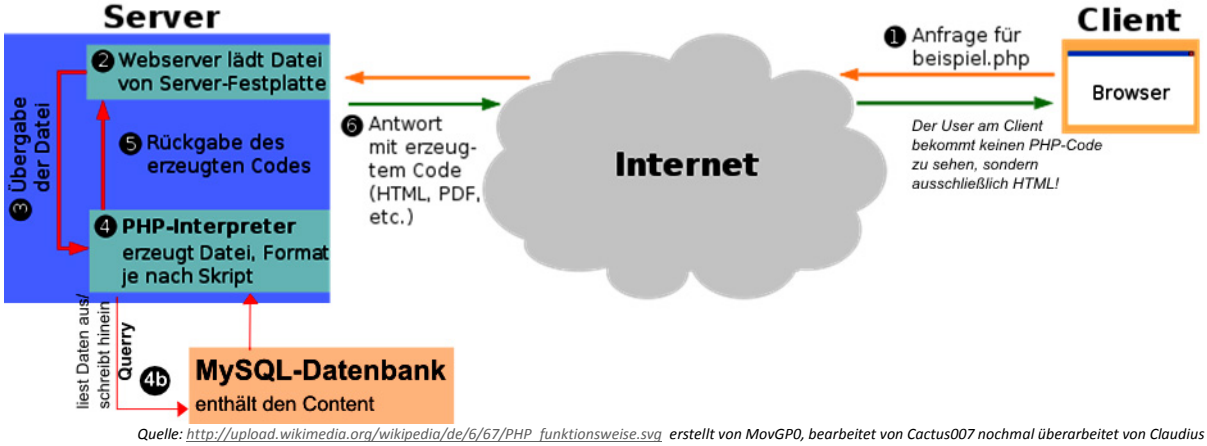
echo *ist dabei ein PHP-Befehl, der die Ausgabe der in Anführungsstrichen folgenden Anweisung bewirkt. Im obigen Beispiel wird also die Ausgabe von HTML-Code erzeugt:* Hello World

Dieser Code wird dann vom Webserver an den Browser des Users geschickt und der kann dann Hello World *lesen.*

© Claudius Mühlhäusler 2012

Wichtige Quellen: http://php.net/manual/de
http://www.selfphp.de/

Funktionsschema von PHP



Quelle: http://upload.wikimedia.org/wikipedia/de/6/67/PHP_funktionsweise.svg erstellt von MovGP0, bearbeitet von Cactus007 nochmal überarbeitet von Claudius Mühlhäusler

MySQL

*MySQL ist ein **relationales Open-Source-Datenbankverwaltungssystem**, das über XAMPP kostenlos und portabel zugänglich ist. Da PHP eine Vielzahl erprobter Zugriffsmöglichkeiten auf MySQL bietet, werden bei den meisten dynamischen Webauftritten diese beiden Programme zusammen eingesetzt.*

MySQL kann, wie PHP, auf einem Webserver installiert werden. Anschließend kann es mittels PHP über eine Internetschnittstelle bedient werden.

*Viele **dynamische Webseiten** funktionieren heutzutage so, dass die Inhalte (=Content) in einer Datenbank abgelegt werden. User können diese Inhalte dann nach ihren eigenen Interessen abrufen: zum Beispiel nach Büchern eines Autors, Musik eines Interpreten oder Nachrichten eines Zeitraumes suchen. Sie können sich anmelden und bekommen dann einen Benutzernamen mit zugehörigem Passwort und können dann eine Site personalisieren und auf ihre Bedürfnisse einstellen.*

*Die wichtigsten Merkmale des **Web 2.0** werden über Content-Management-Systeme aus PHP und MySQL gemanagt: Fremde Inhalte können kommentiert, editiert und, weitergegeben werden, eigene Inhalte können neu erzeugt und publiziert werden …*

Praxis

Das wichtigste Tool zur Pflege non MySQL-Datenbanken ist PHPMyAdmin – auch dieses ist bei XAMPP schon vorinstalliert.

Mit seiner Hilfe kann man mit wenigen Mausklicks eine Datenbank und miteinander verbundene Tabellen mit autoincrementellen IDs erzeugen.

Wichtige Quellen: http://dev.mysql.com/doc/refman/5.1/de/index.html *(5.1!)*

Projekt

Witzedatenbank: Idee von PHP/ MySQL-Guru Kevin Yank, der mehrere gute Bücher darüber verfasst hat.--> Bei Bedarf bitte unbedingt eine neue Auflage verwenden, weil sich besonders PHP in den letzten Jahren – vor allem wegen Sicherheitsproblemen – stark verändert hat.

Am besten lernt man PHP und MySQL über ein eigenes Projekt: learning by doing*!*

Wie wäre es zum Beispiel mit einem eigenen Gästebuch …

- http://www.youtube.com/watch?v=-ISCY7J1_LY *(In Youtube.com gibt es viele PHP/ MySQL-Tutorials!)*

Schritt 1: Erstellen einer Witze-Datenbank über PHPMyAdmin

Installieren Sie XAMPP*. Starten Sie* xampp_start.exe *im XAMPP-Root-Verzeichnis.*

In der Konfigurationsdatei des Webservers Apache (xampp/apache/conf/httpd.conf) ist bei XAMPP automatisch das Verzeichnis xampp/htdocs *als* DocumentRoot *des Webservers festgelegt. Das bedeutet, dass der Webserver die in* htdocs *abgelegten Dateien anzeigt, wenn man eine Seite vom Webserver (localhost) abruft. Der Aufruf von* localhost

© Claudius Mühlhäusler 2012

sendet also die Datei index.html an den aufrufenden PC, sofern sie sich im Verzeichnis htdocs *befindet. Der Aufruf von* localhost/testsite/testsite.html *lädt entsprechend die* Datei d:/xampp/apache /htdocs/testsite/testsite.html *(falls* D *der Laufwerksbuchstabe Ihrer XAMPP-Installation ist).*

Bitte legen Sie zunächst mit phpMyAdmin eine Datenbank witze *wie folgt an:*

1. *Starten Sie einen Browser (Firefox, Chrome, Internetexplorer …) und geben Sie als Adresse* localhost *an. Der Webserver antwortet und nach einer Weiterleitung sehen Sie die XAMPP-Willkommensseite.*
2. *Darauf befindet sich im Navigationsframe ein Link zu* Tools/ phpMyAdmin → *Klicken Sie darauf!*
3. *Klicken Sie auf der Startseite von phpMyAdmin auf den Button* Datenbanken*, geben Sie in das Textfeld* witze *ein und klicken danach auf* Anlegen. → *Nun ist die Datenbank* witze *erzeugt worden.*
4. *Klicken Sie nun auf* witze *(entweder im linken Frame oder in der Datenbankenliste) und erzeugen Sie eine Tabelle* autoren *mit drei Spalten.OK*
   Die erste Spalte heiße ID. Wählen Sie bei Index PRIMARY *(dann ist dieses Feld ein eindeutiges Primärschlüsselfeld) und machen Sie einen ein Häkchen bei* A_I *(=Autoincrement), damit MySQL automatisch einen eindeutigen Wert bei einem neuen Datensatz einfügt.*
   Die zweite Spalte heiße autornn *und sei vom Typ VARCHAR mit einer Länge von 40. Dieses Feld wird den Autornamen aufnehmen und dabei maximal 40 Buchstaben zulassen.*
   Die dritte Spalte heiße autorvn *und sei vom Typ VARCHAR mit einer Länge von 40. Dieses Feld wird den Autorvornamen aufnehmen und dabei maximal 40 Buchstaben zulassen.*
   Lassen Sie alle anderen Felder unverändert.

   autoren ID /autornn/ autorvn
5. *Klicken Sie links auf den Button* Erzeuge Tabelle *mit dem grünen +.*
   Geben Sie als Tabellenname witze *ein*
   Die erste Spalte heiße ID. Wählen Sie bei Index PRIMARY *(dann ist dieses Feld ein eindeutiges Primärschlüsselfeld) und machen Sie einen ein Häkchen bei* A_I *(=Autoincrement), damit MySQL automatisch einen eindeutigen Wert bei einem neuen Datensatz einfügt. – kennen wir ja schon.*
   Die zweite Spalte heiße datum *und sei vom Typ VARCHAR mit einer Länge von 10. Dieses Feld wird das Eintragedatum des Witzes in der 10-stelligen Form:* TT.MM.YYYY *aufnehmen. Es soll nicht vom Typ Date sein, da diese Felder besonders zu behandeln sind. Wir werden später nicht die Datumsfunktionen von MySQL, sondern jene von PHP verwenden.*
   Die dritte Spalte heiße autor *und sei vom Typ INT und nimmt als Fremdschlüssel den Primärschlüssel des ID-Feldes aus der Autortabelle auf – dazu gleich mehr.*
   Die vierte Spalte heiße witz *und sei vom Typ* text*. Dort soll später der eigentliche Witz eingetragen werden.*
   Lassen Sie alle anderen Felder unverändert. Klicken Sie rechts unten auf Speichern.

   witze ID /datum/ autor/ witz
6. *Klicken Sie nun in der zweiten Zeile bei der Tabelle* witze *auf* Struktur *und dann auf* Bezeihungsübersicht*. Wählen Sie in dem Kombinationsfeld* autor *bei* Interne Beziehung: witze.autoren.id *Damit stellen Sie ein, dass in der Tabelle* witze *in der Spalte* autor witze.autoren.id *eingetragen wird, also der Wert* id *aus der Tabelle* autoren *aus der Datenbank* witze*, was je mit einem Punkt verbunden ist. In der Tabelle Witze kann also nur noch die ID eines Autoren stehen, der in der Autor-Tabelle* autor *abgelegt ist. Speichern.*
7. *Um die Lesbarkeit für später zu erhöhen, sollten Sie nun noch zur Struktur der Autortabelle* autor *wechseln und dort in der Beziehungsübersicht bei* Anzuzeigende Spalte autornn *auswählen. Dann bekommen Sie später nicht die ID als Fremdschlüssel* angezeigt*, sondern den Nachnamen – trotzdem steht aber die ID in dem Autorfeld der Witzetabelle* witze! Speichern.
8. *Tragen Sie zunächst zwei Autoren und anschließend zwei Witze in die Datenbank* witze *ein:*
   Wählen Sie die Tabelle autoren *und klicken Sie anschließend auf den Button oben* Einfügen*. Tragen Sie nun die Werte in der Spalte* Wert *(nicht bei* Funktion*) ein. Lassen Sie die A_I-Felder leer. Klicken Sie abschließend auf OK. Sie erhalten eine Erfolgsmeldung (und können den an MySQL geschickten SQL-Code sehen).*
9. *Tragen Sie dann analog zwei Witze in die Witzetabelle* witze *ein. Witze → Einfügen →Werte einfügen*
10. *Nun lesen wir die Daten aus der witze-Datenbank über PHP aus.*
    Erstellen Sie mit Notepad++ eine HTML-Textdatei mit der Endung php *und speichern Sie diese in dem Ordner D:/xampp/htdocs (falls das Laufwerk Ihrer XAMPP-Installation D: ist).* witzelesen.php *können Sie dann unter der folgenden URL mit Ihrem Browser aufrufen:* localhost/witzelesen.php
    Prüfen Sie bei jedem signifikanten Schritt das Ergebnis Ihrer Eingaben (php-Datei speichern, Ansicht im Browser aktualisieren) und warten Sie damit nicht bis zum Schluss!).
11. *Erstellen Sie zunächst den HTML-Rahmen Ihrer Datei, die am Ende ganz ohne PHP-Anweisungen an Ihren Browser geschickt werden wird, der HTML verarbeiten kann:*

```
<html><head><title>Witze lesen</title></head>
<body>
</body>
</html>
```

12. *Um PHP ausführen zu können, müssen Sie den PHP-Code in einem PHP-Tag platzieren:*
    <?php PHP-ANEWEISUNG; ?>
    Überprüfen Sie, ob PHP funktioniert, indem Sie mit dem Echo-Befehl von PHP ein ‚Hallo Welt' ausgeben lassen – platzieren Sie den folgenden Text zwischen die body-Tags.

```
<?php
echo "Hallo Welt";
?>
```

    Speichern. Wenn Sie nun die Datei in Ihren Browser aufrufen, erscheint: Hallo Welt.
    Wenn Sie diese Ausgabe formatieren wollen, können Sie das mit HTML direkt machen oder aber indirekt über PHP realisieren:
    Mit HTML:

```
<h1><?php echo "Hallo Welt"; ?></h1>
```

    Mit PHP 1:

```
<?php echo "<h1>Hallo Welt</h1>"; ?>
```

    Mit PHP 2:

```
<?php echo "<h1>"; echo "Hallo Welt"; echo "</h1>"; ?>
```

13. *Verbindung mit MySQL erstellen:*
    Wenn Sie einen Datenbankzugriff umsetzen wollen, müssen Sie sich zunächst mit MySQL verbinden. DAs erledigt der die folgende PHP-Zeile:

```
$verbindung=mysql_connect(localhost,root);
```

    Der PHP-Befehl mysql_connect *erwartet drei mit einem Komma getrennte Parameter: den Namen des Hosts, den Benutzernamen eines Benutzers und das Passwort dieses Benutzers – bei dem MySQL von XAMPP heißt der Benutzer* root *und da er hier tatsächlich kein Passwort hat, müssen wir auch keines übergeben.*
    (Sie können jederzeit in phpMyAdmin *einen Benutzer erstellen und diesem alle Rechte für die Datenbank witze und ein Passwort zuordnen. Wenn Sie diesen User dann später über PHP in Ihre Datenbank einloggen, dann können Sie Benutzernamen und Passwort im Klartext in das PHP-Script eintragen, da der User über den Webserver den PHP-Code nicht sehen bekommt!)*
    Öffnen Sie nun eine Verbindung zu der Datenbank witze*:*

```
mysql_select_db("witze");
```

    Jetzt können Sie überprüfen, ob eine Verbindung zur Datenbank besteht, indem Sie zum Beispiel die Anzahl der dort eingetragenen Witze abfragen: Tragen Sie dazu zwischen die php-Tags ein:

```
$sqlBefehl="SELECT * FROM witze";
$Witze = mysql_query($sqlBefehl);
echo (mysql_num_rows($Witze));
```

    Der erste Befehl füllt die Variable $sqlBefehl *mit dem Text einer SQL-Query: Mit* SELECT * FROM witze *werden alle Datensätze (=*) *aus der Tabelle* witze *ausgewählt.*
    Der zweite Befehl schickt diesen SQL-Befehl an MySQL und weist den Rückgabewert (=alle Datensätze der Tabelle witze*) der Variablen* $Witze *zu: diese ist nun ein Array, das alle Witze enthält: Jeder Datensatz ist darin eine Zeile.*
    Der dritte Befehl gibt das Ergebnis des Zählbefehls aus, der die Anzahl der Datensätze, bzw. Zeilen in dem Witzearray $Witze *ermittelt. Wenn Sie vorhin zwei Witze eingegeben hatten, müsste der Aufruf von* http://localhost/witzelesen.php *also* 2 *ergeben.*
    Bauen Sie unbedingt noch eine Fehlerbehandlung ein, die sich im Fall eines Fehlers meldet und Sie bei der Fehlerbehebung sinnvoll unterstützt:

```
<?php
mysql_connect('localhost','root') or die(mysql_error());
mysql_select_db('witze') or die(mysql_error());
$sqlBefehl="SELECT * FROM witze";
$Witze = mysql_query($sqlBefehl);
echo (mysql_num_rows($Witze));
 ?>
```

    Ändern Sie nun die Parameter, speichern Sie Ihre Datei und schauen Sie sich die unterschiedlichen Fehlermeldungen an.
14. *Geben Sie die Witze aus der Datenbank aus:*
    Den SQL-Befehl können Sie unverändert übernehmen, denn er macht alle Datensätze der witze-Tabelle aus der Witze-Datenbank für PHP verfügbar.
    Mit dem PHP-Befehl mysql_fetch_row() *können Sie die Datensätze nun nicht nur zählen, sondern Zeile für Zeile auslesen. Immer wenn dieser Befehl aufgerufen wird, gibt er nicht nur einen Datensatz aus dem Witzearray aus, sondern er setzt außerdem einen Zeiger automatisch auf den je nächsten Datensatz. So wird eine Ausgabe der Daten über eine Schleife sehr erleichtert.*

© Claudius Mühlhäusler 2012

Eine Kopfgesteuerte While-Schleife wird in PHP solange ausgeführt, wie die Bedingung (der Ausdruck in der dem Schlüsselwort while *folgenden Klammer) kein* falsch *zurückgibt. Ist die Bedingung wahr, wird der von geschweiften Klammern umfasste Anweisungsblock wiederholt ausgeführt.* mysql_fetch_row($Witze) *ist so lange nicht* false*, wie der Zeiger auf einen neuen Datensatz stößt. Sobald der Zeiger auf das Ende des Witzearrays stößt, gibt er* false *zurück und bewirkt also, dass die While-Schleife abbricht und das nun die Anweisungen nach der geschweiften Klammer ausgeführt werden.*

mysql_fetch_row() *soll nur einmal pro Durchlauf der Schleife aufgerufen werden, denn andernfalls würde der interne Zeiger auf andere Datensätze verschoben. Um also pro Schleifendurchgang mehrfach auf die Daten eines Datensatzes zugreifen zu können, speichert man die Ausgabe dieses einen Funktionsaufrufs in einer Variablen, die genau einen Datensatz, also genau eine Zeile enthält:*

```
while ($row = mysql_fetch_row($Witze)) { }
```

Dieser Befehlsaufruf erzeugt für jeden Schleifendurchlauf eine Variable $row*, die jeweils die Daten eines Datensatzes enthält. Weil die Bedingung in der Klammer einmal ganz am Anfang pro Schleifendurchlauf aufgerufen wird, rückt der Zeiger der fetch_row-Anweisung pro Durchlauf einen Datensatz weiter. Die Daten im* $row*-Array kann man so oft, wie man will abrufen, ohne dass sich eine Zeigerproblematik ergibt. Man kann eine Spalte in einem Datensatz über deren Index, der immer mit 0 beginnt, adressieren: Will man den Inhalt der ersten Spalte des vorleigenden Datensatzes auslesen, dann tut man das mit:* $row[0]

Wir können nun alle Witze aus der Datenbank auslesen:

```
while ($row=mysql_fetch_row($Witze)) { echo $row[0];}
```

Die Ausgabe ist noch sehr unübersichtlich, enthält aber alle Daten aus der Datenbank. Analog können wir auch die Autoren auslesen:

```
<html> <head> <title>Witze lesen</title> </head> <body>
<?php
mysql_connect('localhost','root') or die(mysql_error());
mysql_select_db('witze') or die(mysql_error());
$sqlBefehl1="SELECT * FROM witze";
$Witze = mysql_query($sqlBefehl1);
while ($row=mysql_fetch_row($Witze))
  {
        echo $row[0];
        echo $row[1];
        echo $row[2];
        echo $row[3];
  }
$sqlBefehl2="SELECT * FROM autoren";
$Autoren = mysql_query($sqlBefehl2);
while ($row=mysql_fetch_row($Autoren))
  {
        echo $row[0];
        echo $row[1];
        echo $row[2];
  }
 ?>
</body> </html>
```

15. *Gestaltung der Ausgabe der Datenbankinhalte:*

Eine Tabelle eignet sich immer gut, um Daten aus einer Datenbank auszugeben.

Der table*-Tag umschliießt die komplette Tabelle, muss also vor der Schleife geöffnet und nach der Schleife geschlossen werden. Jeder neue Datensatz soll in eine neue Zeile, also muss jeder Schleifendurchlauf mit einem* td*-Tag beginnen, der bei jedem Durchlaufsende wieder geschlossen wird. Die einzelnen Spalten werden mit einem td-Tag begonnen.*

Bitte beachten Sie: wenn eine PHP-Variable innerhalb eines Ausdrucks in Gänsefüßchen (Anführungsstriche, doppeltes Hochkomma) vorkommt, dann löst PHP sie auf und ersetzt sie durch deren Inhalte. In einfachem Hochkomma gibt PHP eine Variable als Text aus, z. B. als $row[0].

```
echo "<table>";
while ($row=mysql_fetch_row($Witze))
  {
        echo "<tr>";
        echo "<td>$row[0]</td>";
        echo "<td>$row[1]</td>";
        echo "<td>$row[2]</td>";
```

© Claudius Mühlhäusler 2012

```
            echo "<td>$row[3]</td>";
            echo "</tr>";
   }
echo "</table>";
```

Wenn Sie die HTML-Tags durch Attribute verfeinern wollen, müssen Sie beachten, dass Gänsefüßchen innerhalb des echo *Befehls nicht möglich sind, weil sie das führende Gänsefüßchen schließen würden. Deshalb ist es nötig, dass man entweder kombiniert, also den Ausdruck mit dem Kombinationsoperator . zusammensetzt, oder man innerhalb der echo-Gänsefüßchen diejenigen Gänsefüßchen der Attribute durch einfache Hochkommata ersetzt oder sie mit \ maskiert.*

```
mysql_connect('localhost','root') or die(mysql_error());
mysql_select_db('witze') or die(mysql_error());
$sqlBefehl1="SELECT * FROM witze";
$Witze = mysql_query($sqlBefehl1);
echo "<table border='2' width='80%' align='center'>";
while ($row=mysql_fetch_row($Witze))
   {
            echo "<tr bgColor=\"#ffffff\" onMouseOver=\"this.bgColor='#fdcd00'\"
                                                                onMouseOut=\"this.bgColor='#ffffff'\">";
            echo "<td>$row[0]</td>";
            echo "<td>$row[1]</td>";
            echo "<td>$row[2]</td>";
            echo "<td>$row[3]</td>";
            echo "</tr>";
   }
echo "</table>";
```

Der Gestaltung der Tabelle über indirektes HTML oder CSS sind keine Grenzengesetzt.

16. *Unschön ist noch, dass die Ausgabe zwei unterschiedliche Tabellen ausgibt. Es wäre schöner, wenn nur die Witzedatenbank ausgegeben würde, aber mit dem Namen und Vornamen des Witzautoren.*
    Um das zu erreichen kann man zum Beispiel eine SQL-Abfrage an MySQL schicken, die nicht alle Daten einer Tabelle auswählt, sondern nach der Autor-ID filtert: SELECT * FROM autoren WHERE ID = $row[2]
    Dieser SQL-Befehl würde nur diejenigen Datensätze ausgeben, bei welchen die Spalte ID in autoren dem Wert $row[2] *entspricht. Weil* $row[2] *die eindeutige Autor-ID des aktuellen Datensatzes enthält, würde der SQL-Befehl also immer nur einen einzigen Datensatz enthalten, und zwar denjenigen des Autors des Witzes.*
    Insgesamt ergibt sich also der folgende Code:

```
<html><head><title>Witze lesen</title></head>
<body> <h1>Witze zum Totlachen</h1>
<?php
mysql_connect('localhost','root') or die(mysql_error());
mysql_select_db('witze') or die(mysql_error());
$sqlBefehl1="SELECT * FROM witze";
$Witze = mysql_query($sqlBefehl1);
echo "<table border='2' width='80%' align='center'>";
while ($row=mysql_fetch_row($Witze))
   {echo "<tr bgColor=\"#ffffff\" onMouseOver=\"this.bgColor='#fdcd00'\" onMouseOut=\"this.bgColor='#ffffff'\">";
            echo "<td>$row[0]</td>";
            echo "<td>$row[1]</td>";
            //Autor auslesen
            $sqlBefehl2="SELECT * FROM autoren WHERE ID = $row[2]";
            $arAutoren = mysql_query($sqlBefehl2);
            echo "<td>" .mysql_fetch_row($arAutoren)[1] ."</td>";
            echo "<td><span style=\"color:blue\">$row[3]</span></td></tr>";}
echo "</table>";
 ?>
</body>
</html>
```

Nun fehlt nur noch eine Möglichkeit, Daten in die Witzedatenbank einzutragen!

© Claudius Mühlhäusler 2012