

Ein Handbuch

PHP – Dynamische Websites und Clientlösungen

Einführung in die Scriptsprache PHP

von

Urs Gehrig

urs@php.net

Bern, im April 2003

Dieser Inhalt ist unter einem Creative Commons Namensnennung-NichtKommerziell-Weitergabe unter gleichen Bedingungen Lizenzvertrag lizenziert. Um die Lizenz anzusehen, gehen Sie bitte zu <http://creativecommons.org/licenses/by-nc-sa/2.5/> oder schicken Sie einen Brief an Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

Inhalt

<i>Inhalt</i>	<i>I</i>
<i>Abkürzungen</i>	<i>III</i>
<i>Allgemeine Literatur</i>	<i>IV</i>
1 <i>Einleitung</i>	5
1.1 Übersicht.....	5
1.2 Entstehung	5
1.3 Was ist PHP?	6
2 <i>Installation und Grundlagen</i>	8
2.1 Entwicklungen mit PHP	8
2.2 Erste Gehversuche	9
3 <i>Sprachreferenz</i>	11
3.1 Variablen, Typen, Funktionen, Arrays und Kontrollstrukturen	11
3.2 Funktionen	12
3.3 Zeichenkettenfunktionen	16
3.4 CLI – Command Line Interface.....	18
4 <i>Formulare und Parameterübergabe, E-Mail</i>	19
4.1 Formulare	19
4.2 Dateiupload.....	20
4.3 E-Mail.....	21
5 <i>Umgang mit Dateien</i>	22
6 <i>Cookies und Sessions</i>	24
6.1 Cookies	24
6.2 Sessions	25
7 <i>PHP und MySQL</i>	27
7.1 Structured Query Language (SQL).....	27
7.2 Installation und Konfiguration von MySQL	27
7.3 MySQL via Kommandozeile.....	27
7.4 MySQL via PHP	29
8 <i>Bildmanipulation</i>	30
9 <i>PHP und XML</i>	31
9.1 XSLT	32
9.2 DOM Repräsentation von XML.....	33
9.3 XML konventionell erstellt	34
10 <i>PHP und PDF</i>	35

11	<i>Funktionen und Klassen</i>	36
11.1	Objektorientiert Programmieren.....	36
12	<i>Abstraktion</i>	37
13	<i>PEAR</i>	38
14	<i>Grafisches Tool-Kit PHP-GTK</i>	40
15	<i>PHP und ODBC</i>	41
16	<i>XML RPC</i>	42
16.1	Server.....	42
16.2	Client	42
17	<i>Cygwin</i>	44
17.1	Was ist Cygwin?.....	44
17.2	Installation	44
17.3	Häufig verwendete Pakete.....	44
18	<i>Apache Webserver</i>	46
18.1	PHP mit Erweiterungen (modules, extensions).....	47
18.2	Apache unter Windows	47
19	<i>PHP5 Vorschau</i>	49
19.1	Objekte	49
19.2	Funktionen.....	49
19.3	Variablen	50
19.4	Weiter Konzepte	51
20	<i>Linux</i>	52
21	<i>Opensource Community</i>	52
21.1	Community	52
21.2	Tools	53
22	<i>Zusammenfassung</i>	54
23	<i>Stichwortverzeichnis</i>	55
24	<i>Anhang</i>	56
24.1	Syntax-highlighting von Sourcecode.....	56
24.2	Dateifunktionen	57
24.3	MySQL Funktionen in PHP	59
24.4	Links	60

Dieser Inhalt ist unter einem Creative Commons Namensnennung-NichtKommerziell-Weitergabe unter gleichen Bedingungen Lizenzvertrag lizenziert. Um die Lizenz anzusehen, gehen Sie bitte zu <http://creativecommons.org/licenses/by-nc-sa/2.5/> oder schicken Sie einen Brief an Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

Abkürzungen

C

CSV comma separated values
CLI Command line interface

F

FAQ Frequently Asked Questions (Fragen und Antworten)

G

GD GD Graphics Library.

O

ODBC Open DataBase Connectivity¹.

R

RPC Remote Procedure Call

S

SQL Structured Query Language.

U

URI Uniform Resource Identifier².
URL Uniform Resource Locator.

X, Y, Z

XML Extensible Markup Language³.

¹ Open DataBase Connectivity, <http://www.microsoft.com/data/>.

² Web Naming and Addressing Overview, <http://www.w3.org/Addressing/>.

³ XML, <http://www.w3.org/XML>.

Allgemeine Literatur

- FOGEL/BAR 2002** KARL FOGEL; MOSHE BAR, *Open Source-Projekte mi CVS*, mitp-Verlag, Bonn, 2002.
- KRIEKE 1998** RAINER KRIENKE, *UNIX für Einsteiger - Eine praxisorientierte Einführung*, Karl Hanser Verlag, München, 1998.
- POTT/WIELAGE 2000** OLIVER POTT; GUNTER WIELAGE, *XML – Praxis und Referenz*, Markt&Technik Verlag, München, 2000.
- SAMAR/STOCKER 2002** RICHARD SAMAR; CHRISTIAN STOCKER, *PHP de Luxe - Fortgeschrittene PHP Programmierung*, mitp-Verlag, Bonn, 2002.
- YAGER/REESE/KING 1999** RANDY J. YAGER; GEORGE REESE; TIM KING, *MySQL & mSQL – Databases for Moderate-Sized Organizations & Web Sites*, O'Reilly & Associates, Inc., Cambridge, 1999.

1 Einleitung

PHP (PHP Hypertext Preprocessor) ist eine Server-seitige Scriptsprache, die sich hervorragend für die Generierung dynamischer Websites eignet. Die zahlreichen Erweiterungen ermöglichen Datenbankanbindungen, dynamische Bildgenerierung, Filehandling etc. auf komfortable Art und Weise.

1.1 Übersicht

Dieses Kapitel gibt eine Einführung in die Scriptsprache PHP⁴ und erleichtert den Einstieg in die Programmierung von dynamischen Websites. PHP (rekursives Akronym für „PHP: Personal Homepage Programming“) setzt umfangreiche Programmierkenntnisse in anderen Sprachen nicht voraus. Im den folgenden Kapiteln werden über die Grundlagen und Installation hinaus Themen wie das Filehandling, Sessionverwaltung, die Anbindung von Datenbanken und anderem mehr behandelt. Die Syntax von PHP orientiert sich stark an C, sowie C++, Perl und Java und lässt sich sehr einfach erlernen:

```
01 <html>
02 <body>
03     <?php
04         /* Datei: kapitel_01_01.php */
05         print("Hallo! Ich bin ein PHP Script." );
06     ?>
07 </body>
08 </html>
```

Ausgabe:

```
Hallo! Ich bin ein PHP Script.
```

1.2 Entstehung

Gegen Ende des Jahres 1994 hat Rasmus Lerdorf⁵ die Scriptsprache PHP/FI erstmals eingesetzt, um die Zugriffe auf sein online gestelltes Curriculum Vitae zu protokollieren. PHP entstand aus den beiden Paketen PHP („Personal Home Page Tools“) und FI („Form Interpreter“), wobei letzteres ein CGI Tool zur Anbindung einer SQL Datenbank geschrieben worden war. Seit damals hat sich PHP weit verbreitet und gewann mit den nachfolgend überarbeiteten Versionen an Format, sowohl bezüglich Leistungsfähigkeit, wie auch Funktionalität. Wesentlich dazu beigetragen haben Andi Gutmans und Zeev Suraski⁶, deren welche den Parser von Grund auf neu programmiert und PHP damit zu deutlicher Leistungssteigerung verholfen ha-

⁴ PHP, <http://www.php.net>.

⁵ Rasmus Lerdorf, <http://www.lerdorf.com>.

⁶ Andi Gutmans und Zeev Suraski, <http://www.zend.com/management.php>.

ben. Laut Statistiken von Netcraft⁷ und SecuritySpace⁸ kommt PHP als Scriptsprache zur Zeit (12/2001) weltweit bei rund 6,5 Mio. Domains zum Einsatz.

1.3 Was ist PHP?

Was aber genau ist nun PHP? Die drei Haupteinsatzgebiete von PHP sind:

- Server-seitige Scriptsprache; Dazu bedarf es eines Webservers, sowie PHP als CGI- oder Servermodul installiert und einem Webbrowser.
- Command-line Anwendungen; Das CGI-Modul („Common Gateway Interface“) wird direkt im Command-line Modus verwendet, ohne dass dazu ein Webserver installiert werden muss.
- Grafische Window-basierte Client-seitige Anwendungen; Mit PHP-GTK⁹ lassen sich – ebenfalls unabhängig von einem Webserver – Window-basierte Anwendungen programmieren.

Dadurch, dass PHP ein Produkt der Open Source Community ist und unter einer der GNU General Public License (GPL)¹⁰ ähnlichen Lizenz¹¹ frei verfügbar ist, hat PHP den Zugang zu den verschiedensten Betriebssystemen von Unix über Linux, FreeBSD, Microsoft Windows bis hin zu Mac OS X gefunden. Interessant wird PHP auch dadurch, dass praktisch die meisten Webserver - Apache, Microsoft Internet Information Server, Personal Web Server, Netscape and iPlanet Servers, Oreilly Website Pro Server, Caudium, Xitami, OmniHTTPd etc. - unterstützt werden. Der naheliegende Gedanke, PHP könne nur für die Verarbeitung von HTML eingesetzt werden, kann bedenkenlos beiseite gelegt werden; Flash Filme, XHTML, XML und PDF sind nur einige der weiteren Möglichkeiten.

Das wohl beeindruckendste Argument für den Einsatz von PHP ist die Vielzahl der unterstützten Datenbanken.

Adabas D	Ingres	Oracle (OCI7 and OCI8)
dBase	InterBase	Ovrimos
Empress	FrontBase	PostgreSQL
FilePro (read-only)	mSQL	Solid
Hyperwave	Direct MS-SQL	Sybase
IBM DB2	MySQL	Velocis
Informix	ODBC	Unix dbm

⁷ Netcraft, <http://www.netcraft.com/survey/>.

⁸ SecuritySpace, http://www.securityspace.com/s_survey/.

⁹ PHP-GTK, <http://gtk.php.net>.

¹⁰ GNU General Public License, <http://www.gnu.org/copyleft/gpl.html>.

¹¹ PHP Lizenz, <http://www.php.net/license/>.

DBX, eine neue Erweiterung für PHP, erlaubt den transparenten Einsatz einer jeder von ihr unterstützten Datenbank. ODBC („Open Database Connectivity“)¹² unterstützt zudem eine Reihe weiterer Datenbanken.

Mit LDAP, IMAP, SNMP, NNTP, POP3, HTTP, XML-RPC und anderen unterstützt PHP eine grosse Palette an Protokollen, die die Kommunikation mit anderen Diensten erlauben. Erste Resultate der Integration von Microsofts .NET Technologie¹³ unter PHP sind in Form einer Erweiterung verfügbar. Damit dürften sich für Entwickler kaum erahnte Möglichkeiten der Interoperabilität eröffnen!

¹² Microsoft ODBC, <http://www.microsoft.com/data/odbc/>

¹³ PHP .NET, <http://www.php4win.com>.

2 Installation und Grundlagen

2.1 Entwicklungen mit PHP

An PHP arbeiten weltweit um die 600 Programmierer aktiv mit. Nebst dem Core-Team um Rasmus Lerdorf zeichnet sich das PHP-QAT („Quality Assurance Team“)¹⁴ für die Ausgabe der offiziellen Releases¹⁵ verantwortlich. Releaseversionen gelten als stabil und für die Produktion geeignet. Jederzeit sind aber auch aktuellste Versionen über einen CVS Server¹⁶ abrufbar. Releaseversionen sind – ausser für Microsoft Windows – ausschliesslich als Sourcecode verfügbar. Bei der Vielfalt der Betriebssysteme und unterschiedlichen Konfigurationen würde der Aufwand für Distribution von Binaries ins Unermessliche steigen. Einzig und allein für Microsoft Windows steht eine solche Binärdistribution zur Verfügung¹⁷.

Um vernünftig Webapplikationen entwickeln zu können, empfiehlt es sich, eine Installation des Webserver Apache vorzunehmen. Wie PHP, ist auch Apache als Open Source Entwicklung frei verfügbar¹⁸. Unter Microsoft Windows kann dieser Webserver über eine benutzerfreundliche Installationsroutine konfiguriert werden. Empfehlenswert ist folgende Installation:

```
C:\Programme\Apache Group\Apache\      ;Programmdateien
C:\www\                                  ;Server-Root
```

Für die Installation von PHP 4 unter Microsoft Windows kann auf die Arbeit von Daniel Beulshausen und Andreas Otto von php4win.com zurückgegriffen werden. Sie bieten eine PHP Binärdistribution speziell für Microsoft Windows Betriebssysteme an. PHP sollte einfachheitshalber direkt in das Stammverzeichnis installiert werden:

```
C:\php\                                  ;PHP Installation
```

Die Installation kopiert eine zusätzliche Datei in:

```
C:\WINNT\php.ini                          ;PHP Konfigurationsdatei
```

Als sogenanntes Apache-Modul ist PHP im Gegensatz zur CGI-Variante performanter. Einen weiteren Vorteil kann darin gesehen werden, dass von servernahe Funktionen, so bei-

¹⁴ PHP QA, <http://qa.php.net>.

¹⁵ PHP Releases, <http://www.php.net/downloads.php>.

¹⁶ PHP CVS Anleitung, <http://www.php.net/anoncv.php>.

¹⁷ PHP Binary, <http://snaps.php.net/win32/>.

¹⁸ Apache Webserver, <http://httpd.apache.org>.

spielsweise PHP in Verbindung mit anderen Apache-Modulen, profitiert werden kann. Dabei ist sicherlich die Portierbarkeit der PHP Scripts zu beachten, sofern die Entwicklungsumgebung von derjenigen des Internet Service Providers unterscheidet.

Um alternativ PHP als CGI unter Apache zu verwenden ist die httpd.conf Datei wie folgt zu ergänzen:

```
# Verweis auf die Binärdatei von PHP
ScriptAlias /php/ "c:/php/php.exe"
# Dateien mit der Erweiterung .php sollen vom Parser erfasst werden
AddType application/x-httpd-php .php
Action application/x-httpd-php "/php/php.exe"

DocumentRoot "C:/www"
```

Die httpd.conf Datei ist für den Einsatz des PHP Apache-Moduls mit folgenden Zeilen zu ergänzen:

```
# Laden des PHP Apache-Moduls
LoadModule php4_module c:/php/sapi/php4apache.dll
# Dateien mit der Erweiterung .php sollen vom Parser erfasst werden
AddType application/x-httpd-php .php

DocumentRoot "C:/www"
```

Damit das PHP Apache-Modul gestartet wird, muss der Webserver unter Windows angehalten und danach wieder hochgefahren werden:

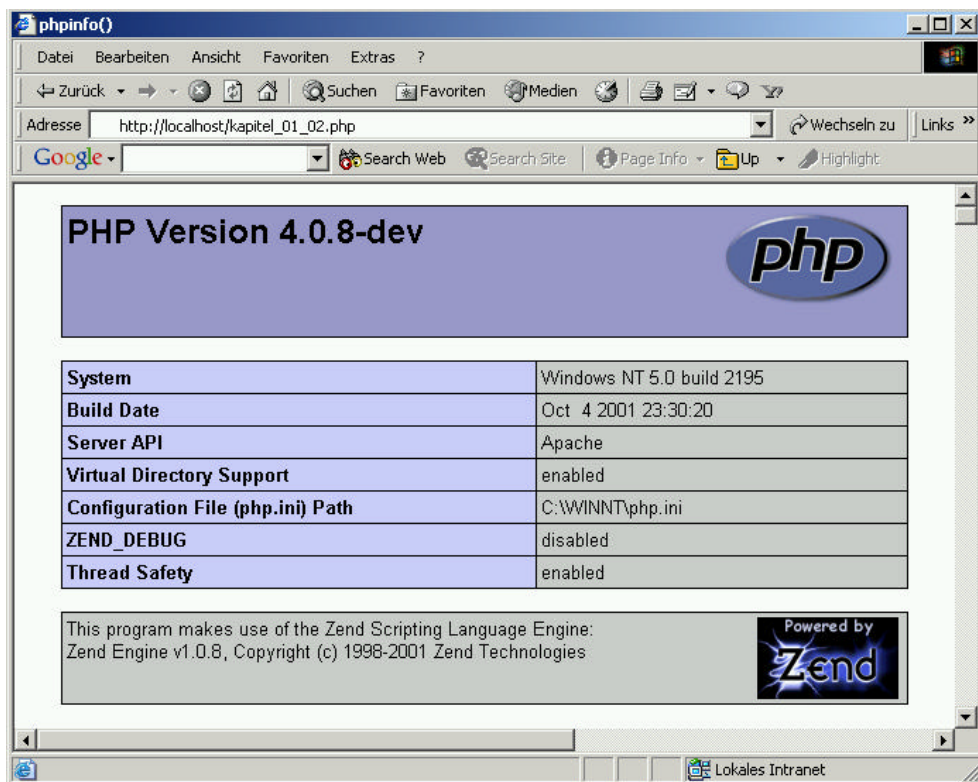
```
C:\>net stop apache
C:\>net start apache
```

2.2 Erste Gehversuche

Der Webserver sollte nun im Browser unter <http://localhost> den Startbildschirm der Apache Standardinstallation anzeigen. Um die Installation von PHP testen zu können, wird das folgende Script in den Ordner C:\www kopiert.

```
<?php
// Datei: kapitel_01_02.php
phpinfo();
?>
```

Unter http://localhost/kap_01/kap_01_02.php wird nach erfolgreicher Installation ein Einblick in die Interna von PHP ermöglicht:



Sollte dieser Schritt nicht gelingen, so empfiehlt es sich das Online-Ausgabe des PHP Handbuchs zur Hilfe zu nehmen. Dankbare Hilfeleistung bieten auch die Archive der Mailinglisten, sowie verschiedenste Resourcendatenbanken. Für den deutschsprachigen Raum seien insbesondere [PHP-Center.de](http://www.php-center.de)¹⁹ und [Dynamic-Webpages.de](http://www.dynamic-webpages.de)²⁰, sowie die Schweizer PHP Usergroup unter [phpug.ch](http://www.phpug.ch) erwähnt²¹.

¹⁹ PHP Center, <http://www.php-center.de>.

²⁰ Dynamic Webpages, <http://www.dynamic-webpages.de>.

²¹ PHP Usergroup Schweiz, <http://www.phpug.ch>.

3 Sprachreferenz

3.1 Variablen, Typen, Funktionen, Arrays und Kontrollstrukturen

Aus den vorangehenden beiden Beispielen wird ersichtlich, wie ein PHP Script aufgebaut wird. Eine PHP Datei wird grundsätzlich von oben nach unten vom Parser bearbeitet. PHP Quellcode wird durch entsprechende Anweisungen ersichtlich gemacht.

```
01 <?php
02     echo "PHP Anweisungen beginnen mit &lt;?";
03 ?>
```

Innerhalb dies Anweisungen müssen die Ausdrücke „<?“ und „?>“ vermieden werden. Kommentare werden, wie wir gesehen haben, in der Form „/* */“ oder mit doppeltem Schrägstrich „/“ gekennzeichnet werden.

3.1.1 Variablen

Im Gegensatz zu anderen Programmiersprachen müssen Variablen nicht deklariert werden. Eine Variable wird erzeugt, indem mit dem Operator „=“ ein Wert zugewiesen und die Zeile mit „;“ abgeschlossen wird. Eine Variable wird durch ein vorangestelltes „\$“ Zeichen erkannt.

PHP unterstützt folgende Datentypen für Variablen:

<i>integer, int</i>	Ganzzahl
<i>double, real</i>	Fliesskommazahl
<i>string</i>	Zeichenkette von 0 bis 32'768 Zeichen
<i>array</i>	Array
<i>object</i>	Objekt

```
01 <?php
02     // Datei: kap_03_04.php, Variablen
03     $titel = "PHP - Dynamische Websites und Clientl&ouml;sungen";
04     $zahl = 2001;
05     $real = 6.0;
06 ?>
```

Variablen können je nach Bedarf auch in andre Typen konvertiert werden, so wird aus einem String folgendermassen ein Integer:

```
01 <?php
02 $str = "5";
03 echo (is_string((int)$str) ? "Ein String\n " : "Kein String\n " );
04 echo (is_integer((int)$str) ? "Ein Int\n " : "Kein Int\n" );
05 ?>
```

In PHP besteht auch die Möglichkeit von variablen Variablen, d.h. dass der Name der Variable bspw. erst in einer for Schleife zugewiesen wird.

```
01 <?php
02 // $Id: kap_03_06.php,v 1.0 2003-05-13 11:06:50+02 urs Exp urs $
03 for($i=0; $i<5; $i++) {
04     echo ${"var_$i"} = $i;
05     echo "\n";
06     ${$i}[$i] = "Wert: $i";
07 }
08 print_r($GLOBALS['4'] ); // Ausgabe: Array ( [4] => Wert: 4 )
09 print_r($GLOBALS['var_3'] ); // Ausgabe: 3
10 ?>
```

3.2 Funktionen

Funtionen erlauben es, bestimmte Operationen, die immer wieder verwendet werden zusammenzufassen. Dadurch kann der Code verständlicher und klarer geschrieben werden.

```
01 <?php
02 // Datei: kap_03_05.php, Variablen im Kontext
03 // Ausgabe: Eine Ausgabe: und noch was...
04
05 function testFunktion($addon="") {
07     return "Eine Ausgabe: $text $addon";
08 }
09 echo blabla("und noch was...");
10 ?>
```

Variablen sind in dem Kontext gültig, in welchem sie definiert worden sind. Verdeutlichen lässt sich das am einfachsten an einer selbst definierten Funktion.

```
01 <?php
02 // Datei: kap_03_05.php, Variablen im Kontext
03 $text = "Globale Variable";
04
05 function blabla($addon="") {
06     global $text;
07     return "Eine Ausgabe: $text $addon";
08 }
09 echo blabla("und noch was...");
10 ?>
```

3.2.1 Konstanten

Konstanten können bis auf die Ausnahmen der vordefinierten Systemkonstanten²² in PHP frei gewählt und definiert werden. Gültige Zeichen sind Buchstaben aus a-z, A-Z und die ASCII-Zeichen von 127 bis 255 (0x7f-0xff).

```
01 <?php
02 define("CONSTANT", "Hallo Welt.");
03 echo CONSTANT; // Ausgabe: "Hallo Welt."
04 echo Constant; // Ausgabe: "Constant" und eine Notice.
05 ?>
```

3.2.2 Operatoren

Operatoren sind für die Bearbeitung von Zahlen und Zeichen unerlässlich. Nachfolgend eine kurze Übersicht:

- +, -, *, /, % u.a.
- Ein „.“ als Concatenator bei Strings (in PHP/FI "+")
- +=, %=, |=, *=, ^= usw., in Ahnlehnung an C
- = (Zuweisung), == (Wertgleichheit), === (Wert- und Typgleichheit, seit PHP4)
- @ zur Unterdrückung von Fehlermeldungen.

Eine umfangreichere Liste ist auf der Website von PHP zu finden²³.

3.2.3 Kontrollstrukturen

Kontrollstrukturen in PHP erinnern an diejenigen der Programmiersprache C. Bedingungsabfragen werden häufig für die Validierung von Formularen eingesetzt.

```
01 <?php
02 // Datei: kap_01_07.php, Kontrollstrukturen
03 $test = 3;
04 if($test == 3) {
05     echo "Wert: drei";
06 } else {
07     echo "Wert: $test";
08 }
09 ?>
```

Nicht zu weit hergeholt sind auch die wichtigen Schleifenstrukturen „do-while“, „while“ und „for“.

²² Vordefinierte Konstanten, <http://ch.php.net/manual/de/language.constants.predefined.php>.

²³ PHP Operatoren, <http://ch.php.net/manual/en/language.operators.php>.

Die Nähe des PHP-Quellcodes zu demjenigen von C macht die Strukturen auf Anhieb verständlich.

```
01 <?php
02 // Datei: kap_01_08.php, Schleifen
03 do {
04     $c = blabla();
05 } while ($c);
06
07 while ($d) {
08     echo "ok<BR />\n";
09     $d = blabla();
10 }
11
12 for ($i = 0; $i < 10; $i++) {
13     echo "i=$i<BR />\n";
14 }
15 ?>
```

Ein häufig verwendetes Konstrukt ist die Switch-Kontrollstruktur, die sich wie die anderen auch an C anlehnt:

```
01 <?php
02 // $Id: kap_03_05.php,v 1.0 2003-05-07 22:30:05+02 urs Exp urs $
03 // switch Struktur
04 $val = 2;
05 switch($val)
06 {
07 case 1:
08     print "val ist 1";
09     break;
10 case 2:
11     print "val ist 2";
12     break;
13 default:
14     print "val ist weder 1 noch 2";
15     break;
16 }
17 ?>
```

3.2.4 Arrays

Häufig drängt es sich auf, mit Variablen geordnet umzugehen. Eine Möglichkeit bieten dabei Arrays, die als mehrdimensionale Variablensammlung verstanden werden können. Nachfolgendes Beispiel zeigt, wie einfach das in PHP gehen kann:

```
01 <?php
02 // Datei: kap_03_06.php, Arrays
03 $array1 = array(1, 3, 4, 7, 13);
04 $array2 = array("eins" => 1, "zwei" => 2, "drei" => 3);
05 $array3 = array(8 => "acht", "neun", "zehn");
06
07 printf("7:%d,1:%d,'neun':%s\n",$array1[3],$array2['eins'],$array3[9]);
08 ?>
```

Wie die Variablen selbst, brauchen auch Arrays nicht deklariert zu werden; sie deklarieren sich sozusagen bei der ersten Zuweisung von Werten mit dem entsprechenden Datentypen selbst.

```
01 <?php
02 /**
03  * Einige Array Funktionen:
04  * in_array($needle, $haystack, $strict)
05  * count($val)
06  * is_array($val)
07  */
08
09 for($i = 0; $i < count($myarray); $i++) {
10     echo $myarray[$i];
11 }
12
13 foreach($array_val as [$key_val =>] $value) {
14     echo $value;
15 }
16 ?>
```

Die soeben besprochenen Grundlagen von PHP decken bei weitem nicht den ganzen Umfang ab. Ein Blick in die Funktionsreferenz auf der Website von PHP²⁴ wird weiterhelfen.

²⁴ Funktionsreferenz, <http://www.php.net/funcref> bzw. Quick Reference unter <http://www.php.net/quickref>.

3.3 Zeichenkettenfunktionen

Mit Zeichenkettenfunktionen werden Strings (Zeichenketten) auf verschiedene Arten verändert. Wenn diese Standardfunktionen nicht ausreichen, kann sich für speziellen Anwendungen in den Abschnitten zu regulären Ausdrücken informieren²⁵.

Im Folgenden werden ein paar Standardfunktionen besprochen:

- `int strlen (string str)`

```
01 <?php
02 $str = "Wie lang wird dieser String wohl sein?";
03 echo strlen($str); // Ausgabe: 38
04 ?>
```

- `mixed str_replace (mixed search, mixed replace, mixed subject [, int &count])`

```
01 <?php
02 $str = "Wie lang wird dieser String wohl sein?";
03 echo str_replace("lang", "kurz", $str );
04 ?>
```

- `int strpos (string haystack, string needle [, int offset])`

```
01 <?php
02 $str = "Wie lang wird dieser String wohl sein?";
03 $pos = strpos ($str, "lang");
04 if ($pos === false) { // Achtung: 3 Gleichheits-Zeichen
05     echo "nicht gefunden...";
06 } else {
07     echo $pos; // Ausgabe: 4
08 }
09 ?>
```

- `string join (string glue, array pieces)`

```
01 <?php
02 $arr = array("Rose", "Tulpe", "Margarite");
03 echo join("; ", $arr); // Ausgabe: Rose; Tulpe; Margarite
04 ?>
```

- `array explode (string separator, string string [, int limit])`

```
01 <?php
02 $str = "Rose,Tulpe,Margarite";
03 $arr = explode(",", $str);
04 print_r($arr ); // Ausgabe: Array([0]=>Rose[1]=>Tulpe[2]=>Margrite)
05 ?>
```

²⁵ Reguläre Ausdrücke, <http://www.php.net/regex>.

Das "Here Document" Format ermöglicht grössere HTML Abschnitte auf einfache Weise auszugeben:

```
01 <?php
02 $head = <<<ENDH
03 <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
04 <html>
05   <head>
06     <title>Feedback Formular</title>
07   </head>
08
09   <body bgcolor="white">
10     <h1 align=center>Feedback Formular</h1>
11   ENDH;
12   echo $head;
13   ?>
```

Folgendes Beispiel zeigt ein einfaches Gästebuch, das jedoch so, ohne auf Sicherheit geprüft zu sein, nicht benutzt werden sollte.

```
01 <?php
02 // $Id$
03 // Einfaches Gästebuch
04 ?>
05 <html>
06   <head><title>My Guestbook</title></head>
07 <body>
08 <h1>Welcome to my Guestbook</h1>
09 <h2>Please write me a little note below</h2>
10 <form action="<?php echo basename($_SERVER['PHP_SELF']) ?>" method="POST">
11   <textarea name="note" wrap="virtual"></textarea><br />
12   <input type="submit" value="Send it">
13 </form>
14 <?php
15 if(isset($_POST['note'])) {
16   $fp = fopen("./notes.txt", "a+");
17   fputs($fp, nl2br($_POST['note']). "<hr /><br />" );
18   fclose($fp);
19 }
20 ?>
21 <h2>The entries so far:</h2>
22 <?php
23   @readfile("./notes.txt")
24 ?>
25 </body>
26 </html>
```

Die Einträge werden direkt wieder aus der Datei notex.txt ausgelesen und angezeigt.

3.4 CLI – Command Line Interface

PHP ist als Scriptsprache für Websites bekannt geworden. Zusehends werden die Möglichkeiten der flexiblen Sprache mit vielseitigem Funktionsumfang auch als solche für die Arbeit im shell bzw. Kommandofenster ersichtlich. PHP Anweisungen können also auch direkt in einem Fenster ausgeführt werden, ohne zuvor eine Datei beschreiben zu müssen.

```
C:>\php\cli\php -h
Usage: php [options] [-f] <file> [args...]
       php [options] -r <code> [args...]
       php [options] [-- args...]

-a          Run interactively
-c <path>|<file> Look for php.ini file in this directory
-n          No php.ini file will be used
-d foo[=bar] Define INI entry foo with value 'bar'
-e          Generate extended information for debugger/profiler
-f <file>   Parse <file>.
-h          This help
-i          PHP information
-l          Syntax check only (lint)
-m          Show compiled in modules
-r <code>   Run PHP <code> without using script tags <?..?>
-s          Display colour syntax highlighted source.
-v          Version number
-w          Display source with stripped comments and whitespace.
-z <file>   Load Zend extension <file>.

args...    Arguments passed to script. Use -- args when first argument
           starts with - or script is read from stdin
```

Nachfolgende Zeilen zeigen, was mit CLI gemeint sein kann:

```
C:>\php\cli\php
<?php
$path = "/this/goes/to/the/path/";
echo trim($path, "/");
?>
^Z
this/goes/to/the/path
```

Unter Windows wird ^Z mittels F6 erzeugt, unter Linux ist *Ctrl+D* einzugeben. Mit der Anweisung *-r* kann PHP Code auch ohne die Script tags ausgeführt werden.

```
C:>\php\cli\php -r "print_r(array_map(\"trim\", array(\" 3 \", \" abc\")));\"
Array
(
    [0] => 3
    [1] => abc
)
```

So können auch Remotedateien geladen werden (bspw. bei der Installation von PEAR):

```
C:>\php\cli\php -r "include(\"http://go-pear.org\");"
```

4 Formulare und Parameterübergabe, E-Mail

4.1 Formulare

Formulare werden vor allem da eingesetzt, wo der Benutzer zur Eingabe von Daten aufgefordert werden soll; einige Beispiele sind Kontaktformulare, Gästebücher und Foren.

```
01 <?php
02 // $Id: kap_04_02.php,v 1.0 2003-05-02 18:13:25+02 urs Exp urs $
03
04 if(stristr($_SERVER['REQUEST_METHOD'], "GET" ))
05 {
06 print <<<FOOBAR
07 <html>
08     <body>
09     <form name="bezeichnung" method="post">
10         Name:<br />
11         <input type="text" name="name" size="25" />
12         <br />
13         Kommentar:<br />
14         <textarea cols="25" rows="5" name="kommentar"></textarea>
15         <br />
16         <input type="submit" value="send" name="Senden" />
17     </form>
18 </body>
19 </html>
20 FOOBAR;
21 } else {
22     printf("<p>Sie heissen: %s</p>\n", $_POST['name'] );
23     printf("<p>Ihr Kommentar: %s</p>\n", $_POST['kommentar'] );
24     print_r($_POST);
25 }
26 ?>
```

Die Formulare sind prinzipiell mit HTML zu erstellen. Für die Auswertung hingegen wird PHP zum Einsatz kommen.

Wenn das „action“-Attribut eines Formulars ein PHP-Script ist, dann stehen die Variablen aus dem Formular und aus den Cookies automatisch als Elemente in einem von drei Arrays in PHP zur Verfügung: Je nach der Art der Übergabe stehen sie in \$_GET, \$_POST oder \$_COOKIE bereit. Wenn das „method“-Attribut nicht angegeben wird, können die Daten auch über das Array \$_REQUEST abgefragt werden.

4.2 Dateiupload

Dateien gilt es oft auch vom Client auf den Server hochzuladen. PHP stellt dazu spezielle Funktionen und Systemvariablen zur Verfügung. Hier ein grundsätzliches Beispiel:

```
01 <?php
02 if(stristr($_SERVER['REQUEST_METHOD'], "GET"))
03 {
04     print <<<EOF
05 <html>
06     <form enctype="multipart/form-data" action=
07     ".{$_SERVER['PHP_SELF']}" method="post">
08         <input type="hidden" name="MAX_FILE_SIZE" value="30000" />
09         Datei uploaden:
10         <input name="userfile" type="file" />
11         <input type="submit" value="Upload" />
12     </form>
13 </html>
14 EOF;
15 }
16 else
17 {
18     echo "<pre>";
19     $uploadaddir = realpath(getcwd());
20     $destination = $uploadaddir . "/" . uniqid("test") . $_FILES['userfile']['name'];
21     $_FILES['userfile']['destination'] = $destination;
22     if (move_uploaded_file($_FILES['userfile']['tmp_name'], $destination) )
23     {
24         print "Datei ist gültig und wurde erfolgreich hochgeladen:\n";
25         print_r($_FILES);
26         printf ("<a href=\"%s\">%s</a>", $_SERVER['PHP_SELF'], "reload");
27     }
28     else
29     {
30         print "Der Dateityp ist nicht zulässig:\n";
31         print_r($_FILES);
32     }
33     echo "</pre>";
34 }
35 ?>
```

Hier die Ausgabe der Protokollierung:

```
01 Datei ist gültig und wurde erfolgreich hochgeladen:
02 Array
03 (
04     [userfile] => Array
05     (
06         [name] => php.png
07         [type] => image/png
08         [tmp_name] => c:\tmp\php180.tmp
09         [error] => 0
10         [size] => 3356
11         [destination] => U:\User\BiCT\test3ebf8d2543377php.png
12     )
13 )
14 reload
```

4.3 E-Mail

E-Mail, also das Verschicken elektronischer Nachrichten, ist heute kaum mehr aus dem täglichen Leben wegzudenken. PHP greift hier auf bestehende Systemressourcen zurück. D.h. es bedarf einer Softwareinstallation, die über das Betriebssystem oder via eines Servers das Verschicken von E-Mails ermöglicht.

```
01 <?php
02     /**
03     * Simple SMTP mail setup for Windows
11     *
12     * http://cvs.php.net/cvs.php/php4/ext/standard/mail.c
13     */
14     // $Id: mail.php,v 1.0 2002-10-16 10:24:07+02 urs Exp urs $
15
16     $sender_email = "me@example.com";
17     $sender_name  = "example.com";
18     $sender       = sprintf("%s <%s>", $sender_name, $sender_email );
19
20     $recipient_cc = "";
21     $recipient_bcc = "you.1@example.com";
22     $recipient     = "you.2@example.com";
23
24     ini_set("SMTP", "smtp.your-isp.ch" );
25     ini_set("sendmail_from", $sender_email );
26
27     $headers = "MIME-Version: 1.0\r\n" ;
28     $headers .= "Reply-To: $sender\r\n";
29     $headers .= "X-Sender: $sender\r\n";
30     $headers .= "X-Mailer: ". phpversion() ."\r\n";
31     $headers .= "X-Priority: 3\r\n";
32     $headers .= "Return-Path: $sender_email\r\n";
33     $headers .= "Content-Type: text/plain; \n\tcharset=iso-8859-1\r\n";
34     $headers .= "Content-Transfer-Encoding: 8bit\r\n";
35     $headers .= "Cc:$recipient_cc\r\n";
36     $headers .= "Bcc:$recipient_bcc\r\n";
37
38     mail($recipient, "a äöüèèà subject", "a äöüèèà message", $headers);
39
40     ini_restore("SMTP");
41 ?>
```

Im wesentlichen für den Versand zuständig ist die Funktion *mail*²⁶.

²⁶ PHP mail-Funktion, <http://ch.php.net/mail>.

5 Umgang mit Dateien

Lesen, Speichern und Schreiben von Dateien sind fast kaum aus der Funktionspalette wegzudenken.

```
01 <?php
02 // $Id$
03 $fp = fopen("source.html", "w+");
04 fwrite($fp, "<html>Eine HTML Datei</html>");
05 fclose($fp);
06 ?>
```

Nicht immer steht eine Datenbank zur Verfügung bzw. es ist nicht immer sinnvoll, auch eine solche zu benutzen. Häufig können Daten in Dateien abgelegt und oder zwischengespeichert werden.

Der Parameter *mode* spezifiziert den gewünschten Zugriffstyp auf den Datei-Stream und kann folgende Werte haben²⁷:

<i>mode</i>	<i>Beschreibung</i>
'r'	Öffnet die Datei nur zum Lesen und positioniert den Dateizeiger auf den Anfang der Datei.
'r+'	Öffnet die Datei zum Lesen und Schreiben und setzt den Dateizeiger auf den Anfang der Datei.
'w'	Öffnet die Datei nur zum Schreiben und setzt den Dateizeiger auf den Anfang der Datei sowie die Länge der Datei auf 0 Byte. Wenn die Datei nicht existiert wird versucht sie anzulegen.
'w+'	Öffnet die Datei zum Lesen und Schreiben und setzt den Dateizeiger auf den Anfang der Datei sowie die Länge der Datei auf 0 Byte. Wenn die Datei nicht existiert, wird versucht sie anzulegen.
'a'	Öffnet die Datei nur zum Schreiben. Positioniert den Dateizeiger auf das Ende der Datei. Wenn die Datei nicht existiert, wird versucht sie anzulegen.
'a+'	Öffnet die Datei zum Lesen und Schreiben. Positioniert den Dateizeiger auf das Ende der Datei. Wenn die Datei nicht existiert, wird versucht sie anzulegen.

Bilder bzw. binäre Dateien werden zudem – insbesondere unter Windows – mit dem Zusatz *b* behandelt, bspw. .

```
01 <?php
02 // $Id$
03 $fp = fopen("bild.png", "wb+");
04 fwrite($fp, $daten);
05 fclose($fp);
06 ?>
```

²⁷ Dateifunktion *fopen*, <http://ch.php.net/fopen>.

Daten, die mit einem Trennzeichen in einer Datei abgelegt sind, können bspw. mittels der Funktion `fgetcsv` zurückgewonnen und einem Array zugeordnet werden.

```
01 <?php
02 // $Id: kap_05_03.php,v 1.0 2003-05-07 23:17:46+02 urs Exp urs $
03 $handle = fopen ("test.csv", "r");
04 while($data = fgetcsv($handle, 1000, ",")) {
05     $num=count($data);
06     print "$num Felder in Zeile $row:<br />\n";
07     print_r($data );
08 }
09 fclose ($handle);
10 ?>
```

Dateien im CSV Format entstehen bspw. mit Hilfe einer Exportfunktion in Tabellenverwaltungsprogrammen oder Datenbanken.

6 Cookies und Sessions

6.1 Cookies

Ein Cookie bietet eine Hilfsfunktion zu Speicherung von Daten auf dem Rechner des Clients.

```
01 <?php
02 // $Id: kap_06_03.php,v 1.0 2003-05-07 21:56:12+02 urs Exp urs $
03
04 function setimage($set=0) {
05     $im = @imagecreate (100, 25) or die ("Nicht unterstützt...");
06     if ($set) {
07         $background_color = imagecolorallocate ($im, 0, 0, 0);
08     } else {
09         $background_color = imagecolorallocate ($im, 128, 128, 0);
10     }
11     imagepng($im, "../cookie.png");
12 }
13
14 $arr = array("Ich mag Guezi.", "Sie mag Guezi.", "Alle mögen Guezi." );
15
16 if(stristr($_GET['act'], "remove" )) {
17     setcookie("mampf", "", time()-3600); // Cookie für lh setzen
18     header("Cache-Control: private");
19     header(sprintf("Location: %s%s",
20         $_SERVER['HTTP_HOST'],
21         basename($_SERVER['PHP_SELF']) ) );
22 } else {
23     setcookie("mampf", serialize($arr), time()+3600 );
24 }
25
26 if (isset($_COOKIE['mampf'])) { // Neuladen und wieder ausgeben
27     setimage(1);
28     echo "<br />\n<img src=\"cookie.png\" alt=\"\" />";
29     $ret = unserialize(stripslashes($_COOKIE['mampf']));
30     foreach ($ret as $name => $value) {
31         echo "<br />\n$name => $value";
32     }
33     printf("<br />\n<a href=\"%s?act=remove\" title=\"\">Loeschen</a>",
34         basename($_SERVER['PHP_SELF']) );
35 } else {
36     setimage(0);
37     echo "<br />\n<img src=\"cookie.png\" alt=\"\" />";
38     echo "<br />\nDa ist kein solches Cookie da.";
39     printf("<br />\n<a href=\"%s\" title=\"\">Setzen</a>",
40         basename($_SERVER['PHP_SELF']) );
41 }
42 ?>
```

Der Anschaulichkeit halber wird ein Cookie in obigem Beispiel in der gleichen Datei sowohl gesetzt, wie auch gelöscht. Dazwischen muss die Datei aber neu geladen werden.

Es können nur diejenigen Cookiedaten gelesen werden, die zu der Domain gehören, die sie auch gesetzt hat. Somit ist es prinzipiell nicht möglich, evtl. vertrauliche Daten vom

Rechner zu lesen. PHP stellt eine einzige Funktion zur Verfügung, mit welcher Cookies gesetzt und gelöscht werden können²⁸.

6.2 Sessions

PHP 4 hatte erstmals Sessions implementiert, nachdem immer häufiger Shop Anwendungen mit PHP programmiert worden waren; denn ein Problem bestand darin, dass die Daten eines Kunden A, der die Ware a kaufen wollte, beim Wechsel der Seiten seine Daten entweder verlor, oder diese dem Kunden B zugeordnet worden waren.

Kurzum, das Sessionhandling basiert auf der Idee, dass beim Wählen einer Website, dem Benutzer eine eindeutige Identifikationsnummer – eben eine Session ID – zugeordnet wird. Diese wird einerseits auf dem Sever abgespeichert und andererseits beim Client von Seite X zu Seite Y weitergegeben, oder, wenn Cookies aktiviert sind, in einem solchen zwischengespeichert. Damit ist die Zuordnung von Kunde A zu Produkt a perfekt.

Hier ein von Rasmus Lerdorf²⁹ zusammengestellter Sesseionhandler, bei welchem die Sessions in einer Datenbank gespeichert werden.

```
01 <?php
02 /**
03 <Directory "/var/html/test">
04     php_value session.save_handler user
05     php_value session.save_path mydb
06     php_value session.name sessions
07 </Directory>
08
09 The MySQL schema looks like this:
10
11 CREATE TABLE sessions (
12     id char(32) NOT NULL,
13     data text,
14     ts timestamp,
15     PRIMARY KEY (id)
16 )
17
18 <?php
19 // Our PHP files under /var/html/test then simply need to
20 // look something like this:
21 require 'handler.php';
22 session_start();
23 session_register('var');
24 $var = "Hello World";
25 ?>
26 */
27 function open($db, $name)
28 {
29     global $stable;
30     mysql_connect('localhost');
```

²⁸ Cookie Referenz, <http://ch.php.net/setcookie>.

²⁹ Rasmus Lerdorf, Tips and Tricks, PHPCon2002, <http://www.lerdorf.com/tips.pdf>.

```

31     mysql_select_db($db);
32     $stable = $name;
33     return true;
34 }
35 function close()
36 {
37     mysql_close();
38     return true;
39 }
40 function read($id)
41 {
42     global $stable;
43     $result = mysql_query("SELECT data FROM $stable WHERE id='$id'");
44     if($result && mysql_num_rows($result))
45     {
46         return mysql_result($result,0);
47     }
48     else
49     {
50         error_log("read: ".mysql_error()."\n",3,"/tmp/errors.log");
51         return "";
52     }
53 }
54 function write($id, $data)
55 {
56     global $stable;
57     $data = addslashes($data);
58     mysql_query("REPLACE INTO $stable (id,data) VALUES('$id','$data')");
59     or error_log("write: ".mysql_error()."\n",3,"/tmp/errors.log");
60     return true;
61 }
62 function destroy($id)
63 {
64     global $stable;
65     mysql_query("DELETE from $stable WHERE id='$id'");
66 }
67 function gc($max_time)
68 {
69     global $stable;
70     mysql_query(
71         "DELETE FROM $stable WHERE UNIX_TIMESTAMP(ts)<UNIX_TIMESTAMP()-$max_time")or
72     error_log("gc: ".mysql_error()."\n",3,"/tmp/errors.log");
73     return true;
74 }
75 session_set_save_handler('open','close','read','write','destroy','gc');
76 ?>

```

7 PHP und MySQL

7.1 Structured Query Language (SQL)

SQL ist ein Sprachkonzept, das es erlaubt, mittels verhältnismässig verständlichen Kommandos bspw. Datenbanken zu erstellen, auf Elemente in Tabellen einer Datenbank zuzugreifen und Werte aufzudatieren.

Grundlegende Kommandos sind:

- CREATE
- INSERT
- UPDATE
- DELETE
- DROP

SQL funktioniert mit Datenbanksystemen wie MySQL, SQLite, MS Access, DB2, Informix, MS SQL Server, Oracle, Sybase, etc. Im Internet finden sich verschiedentliche Einführungen in SQL; einige sind im Anhang erwähnt.

7.2 Installation und Konfiguration von MySQL

Installation unter Windows geschieht mit Hilfe der herunterladbaren Installationspakets. Unter Linux werden entweder die Source Dateien kompiliert und installiert, oder aber auch ein vorhandenes Installationspaket direkt geladen.

Es wird zwischen der eigentlichen Serversoftware und dem Client unterschieden, welcher das Benutzerinterface zur Administration der Daten und des Servers bereitstellt.

7.3 MySQL via Kommandozeile

```
maui:/# mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15 to server version: 4.0.12-log
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql>
```

Wenn diese Kommandozeile erreicht ist, kann man direkt mittels SQL Kommandos auf Datenbanken zugreifen:

```
mysql> CREATE DATABASE test1;
Query OK, 1 row affected (0.05 sec)

mysql> use test1
Database changed

mysql> CREATE TABLE table1 (a tinyint not null);
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO test1.table1 VALUES ('45');
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM table1;
+-----+
| a     |
+-----+
| 127   |
+-----+
1 row in set (0.00 sec)
```

Um die Datenbank zu sichern, wird das Kommando mysqldump verwendet.

```
C:\>mysqldump test1

-- MySQL dump 9.07
-- Host: localhost      Database: test1
-----
-- Server version      4.0.7-gamma-max-nt
-- Table structure for table 'table1'

CREATE TABLE table1 (
  a tinyint(4) NOT NULL default '0'
) TYPE=MyISAM;

-- Dumping data for table 'table1'

INSERT INTO table1 VALUES (127);
INSERT INTO table1 VALUES (45);

C:\>
```

Weitere Informationen sind auf der Website von MySQL zu finden³⁰.

³⁰ MySQL Referenzhandbuch, <http://www.mysql.com/doc/de/>.

7.4 MySQL via PHP

Die zuvor erstellte Tabelle *table1* in der Datenbank *test1* soll mittels der MySQL Funktionen, die PHP zur Verfügung stellt, bearbeitet werden.

```
01 <?php
02 // $Id$
03 $link = mysql_connect ("localhost", "mysql_user", "mysql_password")
04     or die ("keine Verbindung möglich: " . mysql_error());
05 print ("Verbindung erfolgreich");
06 mysql_close ($link);
07 ?>
```

Eine Funktionsreferenz von MySQL in PHP findet sich im Anhang. Später wird die Klassensammlung PEAR vorgestellt, die unter anderem Datenbank Abstraktionen für verschiedene Datenbanksysteme, so z.B. auch für MySQL anbietet.

```
01 <?php
02 require_once 'DB.php';
03 $db = DB::connect('mysql://user:passwort@host/mydb');
04 $stmt = $db->prepare('SELECT * FROM comments;');
05 $result = $db->execute($stmt);
06 while($row = $db->fetchrow($result)) {
07     while($row as $feld => $wert ) {
08         echo "$feld: $wert<br />\n";
09     }
10 }
11 $db->disconnect();
12 ?>
```

8 Bildmanipulation

Die Grafikfunktionen von PHP basieren einerseits auf der GD Library von Thomas Boutell³¹, die bislang als Extension zugeladen werden muss. Es besteht auch die Möglichkeit auf Imagemagick und NetPBM, sowie andere zurückzugreifen.

```
01 <?php
02 // $Id$
03 // php -f kap_08_01.php > out.png
04 header ("Content-type: image/png");
05 $im = @ImageCreate (60, 50)
06     or die ("Kann keinen neuen GD-Bild-Stream erzeugen");
07 $background_color = ImageColorAllocate ($im, 255, 255, 255);
08 $text_color = ImageColorAllocate ($im, 233, 14, 91);
09 ImageString ($im, 1, 5, 5, "Ein Test-String", $text_color);
10 ImagePNG ($im);
11 ?>
```

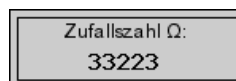
Resultat des obigen Scripts ist in etwa folgendes Bild:



Ein Test-String

```
01 <?php
02 // $Id: kap_08_02.php,v 1.0 2003-05-02 18:29:39+02 urs Exp urs $
03 // 
04
05 function get_random()
06 {
07     srand(time());
08     $max = getrandmax();
09     return rand(1,$max) + rand(1,$max) ;
10 }
11 $type = array("png", "jpeg");
12 $img = imagecreate(150,50);
13 $white = imagecolorallocate($img,255,255,255);
14 $black = imagecolorallocate($img,0,0,0);
15 $grey = imagecolorallocate($img,204,204,204);
16 imagefill($img,0,0,$grey);
17 imagerectangle($img,5,5,144,44,$black);
18 imagerectangle($img,0,0,149,49,$black);
19
20 $number = get_random();
21 @imagettftext($img,9,0,35,18,$black,"C:/WINNT/Fonts/Arial.ttf","Zufallszahl &#937;:");
22 imagestring($img,9,50,25,$number,$black);
23
24 header("Content-type: image/" . $type[0]);
25 eval( 'image' . $type[0] . '($img);' );
26 imagedestroy($img);
27 ?>
```

Resultat des obigen Scripts ist in etwa folgendes Bild:



Zufallszahl Q:
33223

³¹ GD von Boutell, <http://www.boutell.com/gd/>.

9 PHP und XML

Daten können in XML strukturiert abgelegt werden. Einführendes Beispiel stellt eine Funktion dar, die die Werte eines Arrays in XML umwandelt.

PHP stellt standardmässig den EXPAT³² - ursprünglich von James Clark³³ entwickelt - Parser zur Verfügung, mit welchem XML Dokumente abgearbeitet werden. DOM XML ist ein extension, die auf libxml{2} basiert und EXPAT in gewissen Punkten überlegen ist.

Ein XML Dokument kann mit EXPAT folgendermassen geparkt werden:

```
01 <?php
02 // $Id: kap_09_00.php,v 1.0 2003-05-10 17:54:20+02 urs Exp urs $
03
04 $data = file_get_contents('kap_09_00.xml');
05
06 $xh = xml_parser_create();
07 xml_parser_set_option($xh, XML_OPTION_CASE_FOLDING, true );
08 xml_parser_set_option($xh, XML_OPTION_SKIP_WHITE, true );
09 xml_parse_into_struct($xh, $data, $wishlist, $tags );
10 xml_parser_free($xh);
11
12 // Access All Description Elements
13 $descs = $tags['DESCRIPTION'];
14 foreach ($descs as $descid)
15 {
16     echo $wishlist[$descid]['value'];
17     echo "<br />\n";
18 }
19 ?>
```

XML Quelldatei (*kap_09_00.xml*):

```
01 <?xml version="1.0"?>
02 <wishlist uri="http://www.amazon.com/exec/obidos/">
03     <item id="3826607996">
04         <title>PHP de Luxe - fortgeschrittene PHP-Programmierung</title>
05         <author>Richard Samar, Christian Stocker</author>
06         <description>PHP deutschsprachig ausgeleuchtet.</description>
07     </item>
08     <item id="0672323257">
09         <title>PHP Developer's Cookbook</title>
10         <author>Sterling Hughes</author>
11         <description>The best darn book out there.</description>
12     </item>
13 </wishlist>
```

³² PHP und Expat, <http://ch.php.net/manual/de/ref.xml.php>.

³³ James Clark's XML Parser Toolkit, <http://www.jclark.com/xml/expat.html>.

9.1 XSLT

XSLT ist ein Acronym für die Transformation von XML Dateien via XSL Stylesheets in ein nahezu beliebiges anderes Format (XML, XSL, PDF, XHTML etc.)

```
01 <?xml version="1.0" encoding="iso-8859-1"?>
02 <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
03   <xsl:output method="html" version="1.0" indent="yes" />
04
05   <xsl:template match="/wishlist">
06     <xsl:call-template name="items" />
07   </xsl:template>
08
09   <xsl:template name="items">
10     <xsl:for-each select="item">
11       <xsl:value-of select="description" /><br />
12     </xsl:for-each>
13   </xsl:template>
14 </xsl:stylesheet>
```

Die obige Datei stellt das Stylesheet (XSL) dar, mittels welchem das Element description aus der Datei *kap_09_00.xml* in HTML transformiert werden soll. Dazu verwendet man einen Prozessor. Gängige Prozessoren sind *Sablotron*, *MSXML*, *libXSLT*, *SAX*, *XT*, *Xalan* etc.

Das Script zeigt die XSLT mit der PHP extension Sablotron (ext/xslt).

```
01 <?php
02 // $Id: kap_09_05_xslt.php,v 1.0 2003-05-13 00:53:09+02 urs Exp urs $
03 $xml = "./kap_09_00.xml";
04 $xsl = "./kap_09_00.xsl";
05
06 $xh = xslt_create();
07
08 if(!$result = @xslt_process( $xh,$xml,$xsl,NULL,array(),array()))
09 {
10   printf( "Sablotron Error (%s): <br /><b>%s</b>",
11          xslt_errno($xh ), xslt_error($xh ) );
12 }
13 xslt_free($xh );
14 echo $result;
15 ?>
```

9.2 DOM Repräsentation von XML

Ein XML Dokument kann an sich als ein sogenanntes Document Object Model (DOM) gesehen werden. Eine gängige Sicht ist dabei die des Baums oder des *tree*.

```
01 <?php
02 /**
03  * $Id: kap_09_06_domxml.php,v 1.0 2003-05-13 01:15:59+02 urs Exp urs $
04  * Ausgabe: <?xml version="1.0"?>
05  *           <html><body><h1>Hello PHP world.</h1></body></html>
06  */
07
08 // Neues DomAttribute object erstellen
09 $dom = domxml_new_doc('1.0');
10
11 // Elemente erzeugen
12 $htmlElement = $dom->create_element('html');
13 $bodyElement = $dom->create_element('body');
14 $headingElement = $dom->create_element('h1');
15 $headingTextElement = $dom->create_text_node("Hello PHP world.");
16
17 // XML document konstruieren
18 $headingElement->add_child($headingTextElement);
19 $bodyElement->add_child($headingElement);
20 $htmlElement->add_child($bodyElement);
21
22 // Das root Element zum Dokument hinzufügen
23 $dom->append_child($htmlElement);
24
25 // Ausgabe
26 echo $dom->dump_mem();
27
28 ?>
```

Die Sicht als *tree* des Dokuments kap_09_00.xml kann wie folgt erstellt werden.

```
01 <?php
02 // $Id: kap_09_07_xpath.php,v 1.0 2003-05-13 01:44:04+02 urs Exp urs $
03 // Ausgabe: The best darn book out there.
04 $doc = xmldocfile("kap_09_00.xml");
05 $context = xpath_new_context($doc);
06
07 $nodes = xpath_eval($context, "//description/text()");
08 print_r($nodes->nodeset[1]->content );
09 ?>
```

Das Objekt *\$nodes* repräsentiert ein Array folgender Form:

```
01 XPathObject Object
02 (
03     [type] => 1
04     [nodeset] => Array
05         (
06             [0] => domtext Object
07                 (
08                     [type] => 3
09                     [name] => #text
10                     [content] => PHP deutschsprachig ausgeleuchtet.
11                     [0] => 6
```

```

12         [1] => 14390528
13     )
14
15     [1] => domtext Object
16     (
17         [type] => 3
18         [name] => #text
19         [content] => The best darn book out there.
20         [0] => 7
21         [1] => 14410464
22     )
23 )
24 )

```

Hilfreiche Tutorials zu XSLT finden sich unter Zvon.org³⁴.

9.3 XML konventionell erstellt

Ab und zu kann es auch nötig sein, eine Arraystruktur in XML umzuwandeln:

```

01 <?php
02 // $Id: datei_03_01.php,v 1.0 2003-05-01 23:29:27+02 urs Exp urs $
03 // # astyle --style=kr --mode=c < datei_03_01.php > out.php
04
05 function arr2xml($arr, $indent="") {
06     foreach($arr as $key => $val ) {
07         if ($key == "__attr")
08             continue;
09         // auf __attr prüfen
10         if (is_object($val->__attr)) {
11             foreach ($var->__attr as $key2 => $val2 ) {
12                 $attr .= " $key2=\"$val2\"";
13             }
14         } else {
15             $attr = "";
16         }
17         if (is_array($val) || is_object($val)) {
18             print("$indent<$key$attr>\n");
19             arr2xml($val, $indent . " ");
20             print("$indent</$key>\n");
21         } else {
22             print("$indent<$key$attr>$val</$key>\n");
23         }
24     }
25 }
26 ?>

```

Obige Funktion setzt ein Objekt oder Array in einen XML string um.

³⁴ Zvon XSLT Referenz, <http://www.zvon.org/xxl/XSLTreference/Output/index.html>.

10 PHP und PDF

Nebst der Extension ext/fpdf ist es auch möglich, mit Hilfe der Klasse *fpdf*³⁵ mit PHP einfache PDF Dateien dynamisch zu generieren.

```
01 <?php
02     // $Id$
03     // http://fpdf.org
04     define('FPDF_FONTPATH', 'font/');
05     require_once('fpdf.php');
06
07     $pdf= &new FPDF();
08     $pdf->Open();
09     $pdf->AddPage();
10     $pdf->SetFont('Arial', 'B', 16);
11     $pdf->Cell(40, 10, 'Grosse weite Welt!');
12     $pdf->Output();
13 ?>
```

³⁵ FPDF, <http://www.fpdf.org>.

11 Funktionen und Klassen

11.1 Objektorientiert Programmieren

Strukturiertes und endlich objektorientiertes (OO) Programmieren ist letztendlich auch eine Frage der Wartbarkeit des Codes. Im weiteren lassen sich abstrahierte Lösungen etwas einfacher auf mehrere Programmierer aufteilen. In diesem Zusammenhang wird auch etwa die Wiederverwendbarkeit von Codes erwähnt.

Ungeachtet der unterschiedlichen Motivationsgründe für OO sollen hier die grundlegenden Konzepte des objektorientierten Programmierens in PHP 4 aufgezeigt werden.

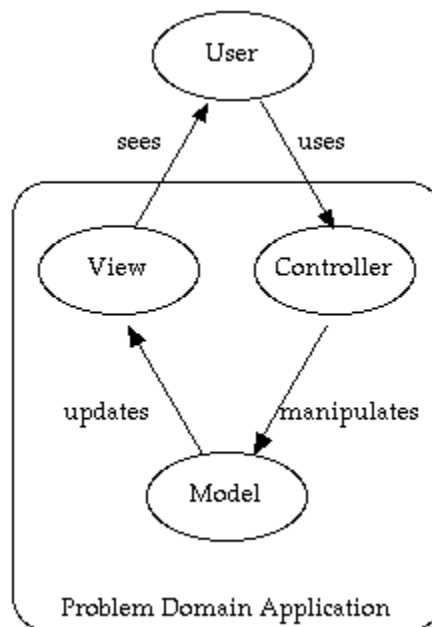
```
01 <?php
02 // $Id$
03 // http://www.php.net/manual/de/keyword.parent.php
04 class A
05 {
06     function example()
07     {
08         echo "Beispiel A::example() mit Grundfunktionen.<br>\n";
09     }
10 }
11
12 class B extends A
13 {
14     function example()
15     {
16         echo "Beispiel B::example() mit Zusatzfunktionen.<br>\n";
17         parent::example();
18     }
19 }
20
21 $b = new B;
22
23 // Zuerst wird B::example() aufgerufen, danach A::example().
24 $b->example();
25 ?>
```

Die Klasse A hat eine einzige Methode oder Funktion, nämlich *example*. Die Klasse B greift auf diese Methode mittels Vererbung der Klasse A zu und fügt zudem eine Änderung hinzu (*echo* „Beispiel...“;).

12 Abstraktion

Mit steigendem Umfang eines Projekts ist man gut beraten, sich schon früh die Frage nach der Aufteilung der Problembereiche der Anwendung zu machen. Eine solche Methode nennt sich MCV, bzw. Model-Controller-View³⁶.

- **Model:** Die Kernapplikation. Sie repräsentiert Zustände und Daten der Anwendung. Bei signifikanten Änderungen im Modell, werden alle ihre Views aufdatiert.
- **Controller:** Stellt das Userinterface dar, welches dem Benutzer zur Manipulation der Anwendung zur Verfügung gestellt wird.
- **View:** Das Benutzerinterface, welches Informationen des Modells dem Benutzer zur Verfügung stellt. Jedes Objekt, das Informationen vom Model benötigt, muss eine registrierte View des Modells sein.



Dieses Modell lässt sich auch auf Lösungen anwenden, die mit PHP realisiert werden. Harry Fuecks hat dazu einen Artikel geschrieben³⁷.

³⁶ MCV, <http://www.cs.indiana.edu/~cbaray/projects/mvc.html>.

³⁷ PHP MVC, <http://www.phppatterns.com/index.php/article/articleview/11/1/8/>.

13 PEAR

PEAR ist eine Sammlung von PHP Klassen und Funktionen und ist ein Pendant zur der Perl Bibliothek CPAN.

```
01 Microsoft Windows 2000 [Version 5.00.2195]
02 (C) Copyright 1985-2000 Microsoft Corp.
03
04 C:\>pear list
05 INSTALLED PACKAGES:
06 =====
07 +-----+-----+-----+
08 | PACKAGE          | VERSION | STATE |
09 | Archive_Tar      | 1.0     | stable|
10 | Auth             | 1.2.0   | stable|
11 | Auth_SASL        | 1.0.0   | stable|
12 | Cache            | 1.5.3   | stable|
13 | Console_Getopt   | 1.0     | stable|
14 | DB               | 1.4b1   | beta  |
15 | HTML_Form        | 1.0.1   | stable|
16 | HTML_Template_IT | 1.1     | stable|
17 | HTTP_Request     | 1.1.1   | stable|
18 | Image_Transform  | 0.2     | beta  |
19 | Mail             | 1.0.2   | stable|
20 | Mail_Mime        | 1.2.1   | stable|
21 | Net_POP3         | 1.2     | stable|
22 | Net_SMTP         | 1.2.2   | stable|
23 | Net_Socket       | 1.0.1   | stable|
24 | Net_URL          | 1.0.10  | stable|
25 | PEAR             | 1.0.1   | stable|
26 | SOAP            | 0.7.1   | beta  |
27 | XML_Parser       | 1.0.1   | stable|
28 | XML_RPC          | 1.0.4   | stable|
29 | XML_XSLT_Wrapper | 0.2     | alpha |
30 | XML_image2svg    | 0.1     | stable|
31 | XML_sql2xml      | 0.3.1   | beta  |
32 +-----+-----+-----+
33
34 C:\>
```

Ist PEAR installiert, so können neue Klassen auf einfache Weise installiert werden. Eine Klasse zur Behandlung von HTML Templates stellt *HTML_Template_IT* dar. Sie ermöglicht es, die Trennung von Design und Coding vorzunehmen, bzw. wieder zusammenzubringen. Auf der einen Seite steht also das HTML Template, das statt Inhalte, bloss HTML mit Variablen enthält, die mittels eines PHP Scripts dann erst mit Werten gefüllt werden. Variablen werden in der HTML Datei mit geschweiften Klammern gekennzeichnet.

Im folgenden ein Beispiel zur Verdeutlichung:

```
01 <!-- main.tpl.html //-->
02 <html>
03 <table border="1">
04   <tr>
05     <td>{VORNAME}</td><td>{NAME}</td>
06   </tr>
07 </table>
08 </html>
```

Die oben beschriebene Datei soll eine HTML Tabelle ausgeben. Der Designer kann diese nach Belieben modifizieren, solange er die Variablen unverändert lässt.

```
01 <?php
02 require_once "HTML/Template/IT.php";
03
04 $data = array ( "0" => array("Hans", "Muster") );
05
06 $tpl = new HTML_Template_IT("./templates");
07 $tpl->loadTemplatefile("main.tpl.html", true, true);
08
09 $tpl->setVariable("VORNAME", $data[0][0]) ;
10 $tpl->setVariable("NAME", $data[0][1]) ;
11 $tpl->parse() ;
12 $tpl->show();
13 ?>
```

Die Template Datei *main.tpl.html* befindet sich im Unterordner *templates*.

14 Grafisches Tool-Kit PHP-GTK

PHP-GTK ist ein völlig anderer Ansatz um die Scriptsprache zu verwenden, denn basieren auf GTK – dem Gnome Toolkit – können auf der Clientseite grafische Benutzeroberflächen (GUI) programmiert werden. Es braucht dazu keinen Webserver.

```
01 #!/usr/local/bin/php -q
02 <?php
03 // $Id$
04 // C:\>php -f kap_14_01.php
05
06 function _quit()
07 {
08     Gtk::main_quit();
09 }
10
11 dl( (stristr(PHP_OS, "WIN") ? 'php_gtk.dll' : 'php_gtk.so') ) or
12     die( 'Konnte php_gtk extension nicht laden!\n' );
13
14 $window = &new GtkWindow;
15 $window->set_title("PHP funktioniert mit GTK.");
16 $window->set_default_size(gdk::screen_width()*0.50, gdk::screen_height()*0.50);
17
18 $window->connect( 'delete_event', '_quit' );
19 $window->show_all();
20
21 Gtk::main();
22 ?>
```

Das Beispiel erzeugt ein GTK Fenster mit dem Titel „*PHP funktioniert mit GTK*“.

15 PHP und ODBC

ODBC steht für Open DataBase Connectivity³⁸. Es handelt sich um echte native Datenbankschnittstellen in PHP, die den Funktionsnamen und den Funktionssyntax der ODBC-Funktionen nutzen.

```
01 <?php
02 // via ODBC
03 $db = odbc_connect("DSN=SQLite.1;UID=;PWD=", "", "" );
04 $result = odbc_exec($db, "SELECT * from tbl1;") or die(odbc_errormsg());
05 if (odbc_fetch_row($result))
06 {
07     for($i=1; $i<=odbc_num_fields($result); $i++)
08     {
09         $key = odbc_field_name($result, $i);
10         $val = odbc_result($result, $key);
11         $stack[$i-1] = array($key => $val);
12     }
13 }
14 print_r($stack);
15
16 // via PECL SQLite extension
17 $database = "./var/testdb";
18 $db = sqlite_open($database);
19 $r = sqlite_query("SELECT * from tbl1;", $db);
20 $stack = sqlite_fetch_array($r, SQLITE_NUM);
21 sqlite_close($GLOBALS['db']);
22 print_r($stack);
23 ?>
```

Obiges Beispiel zeigt, wie die Datenbank *testdb* auch via ODBC und alternativ mittels der PECL extension (ab Zeile 16) angesprochen werden kann.

³⁸ Open DataBase Connectivity, <http://www.microsoft.com/data/>.

16 XML RPC

XML RPC³⁹ stellt ein Framework dar, mittels dessen ein Client auf einem entfernten Rechner eine Funktion aufrufen kann – sog. *remote procedure calls*, welche dem Client einen Rückgabewert liefert. Dieses Konzept geht zurück auf Dave Winer⁴⁰, der dies schon 1985 mit Bill Gates besprochen haben soll; damals ging es um die Kommunikation zwischen Applikationen unter MS-DOS.

16.1 Server

Nebst anderen, war Edd Dumbill⁴¹ einer der ersten, der das Client-Server Konzept auch auf PHP appliziert hat. Seine beiden Klassen fanden letztlich Einzug in PEAR⁴². Nachfolgend ein Beispiel der Serverimplementation welches voraussetzt, dass PEAR installiert ist und die Klassen von XML RPC zur Verfügung stehen.

```
01 <?php
02 // $Id: kap_16_06_server.php,v 1.0 2003-05-07 18:28:09+02 urs Exp urs $
03
04 require_once "XML/RPC/Server.php";
05
06 function hello($params) {
07     $obj = new XML_RPC_Value("Hello World", "string");
08     return new XML_RPC_Response($obj);
09 }
10 $arr = array("examples.hello" => array("function" => "hello"));
11 $server = new XML_RPC_Server($arr);
12 ?>
```

Anstelle der mittels PHP codierten Klassen kann auch auf die PHP extension *xmlrpc* zurückgegriffen werden⁴³.

16.2 Client

Der Client ruft die im Server Script individuellen Funktionen auf und verarbeitet die Rückgabewerte. Die Daten werden in der Regel über den Port 80 geschickt.

³⁹ XML RPC Spezifikation, <http://www.xmlrpc.com/spec>.

⁴⁰ Dave Winer, <http://davenet.userland.com/1998/07/14/xmlRpcForNewbies>.

⁴¹ Edd Dumbill, <http://xmlrpc.usefulinc.com/php.html>.

⁴² XML_RPC, <http://pear.php.net/package-info.php?pacid=17>.

⁴³ PHP XML RPC extension, <http://www.php.net/manual/en/ref.xmlrpc.php>.

```

01 <?php
02 // $Id: kap_16_06_client.php,v 1.0 2003-05-07 18:37:07+02 urs Exp urs $
03 // Client
04
05 require_once "XML/RPC.php";
06
07 $rpcserver = "/htdocs/kap_16/kap_16_06_server.php";
08 $server = new XML_RPC_Client($rpcserver, 'localhost', 91);
09 $server->setDebug(1);
10 $message = new XML_RPC_Message("examples.hello" );
11 $result = $server->send($message);
12
13 // Process the response.
14 if (!$result) {
15     print "<p>Could not connect to HTTP server.</p>";
16 } elseif ($result->faultCode()) {
17     printf("<p>XML-RPC Error# %s: %s",
18         $result->faultCode(),
19         $result->faultString());
20 } else {
21     $struct = $result->value();
22     printf("<b>%s</b>", $struct->me['string']);
23 }
24 ?>

```

Der Rückgabewert bei erfolgreicher Ausführung müsste “Hello World” sein. Es können zudem gewissen Debuginformationen ausgegeben werden.

```

---GOT---
HTTP/1.1 200 OK
Date: Wed, 07 May 2003 16:27:24 GMT
Server: Apache/2.0.45 (Win32) PHP/4.3.2-RC
X-Powered-By: PHP/4.3.2-RC
Connection: close
Content-Type: text/xml
Content-length: 160

<?xml version="1.0"?>
<!-- DEBUG INFO: -->
<methodResponse>
<params>
<param>
<value><string>Hello World</string></value>
</param>
</params>
</methodResponse>
---END---

---EVALING---[42 chars]---
new XML_RPC_Value("Hello World", 'string');
---END---

Hello World

```

17 Cygwin

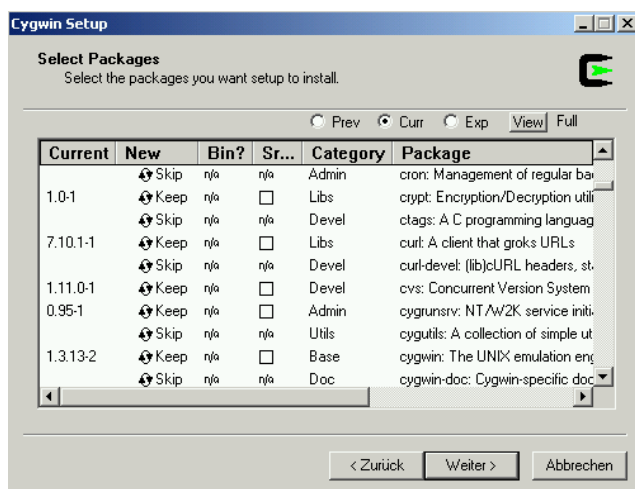
17.1 Was ist Cygwin?

Cygwin ist Unix-konforme Entwicklungs- und Applikationsumgebung unter Windows. Eine umfangreiche Anzahl von Paketen ist auf Windows portiert worden.

Die von Cygwin angebotene Software kann bis auf ein Basispaket vollkommen frei konfiguriert werden. Dazu bietet Cygwin eine visuelle Installationsroutine an. Starten Sie direkt aus diesem Link heraus das Setupprogramm: <http://www.cygwin.com/setup.exe>. Nachfolgend eine typische Dateistruktur eines Cygwin-Systems:

```
c:\cygwin\bin
  \etc
  \lib
  \source
  \tmp
  \usr
  \var
```

17.2 Installation



Cygwin wird empfehlenermassen in `c:\cygwin` installiert, das Standard-Dateiformat ist dabei "Unix". In einer ersten Phase sollte das Basispaket installiert und dann in einem weiteren Schritt, die nach den individuellen Interessen ausgewählten Programme installiert werden. Programme können jederzeit nach- oder deinstalliert werden. Zusätzlich können

dabei die Quelldateien heruntergeladen werden, sofern man die Pakete selbst kompilieren möchte. Sollten Sie auf den grafischen Komfort der Windowsinstallation von MySQL und Apache verzichten wollen, so können diese beiden Pakete auch unter Cygwin installiert werden.

17.3 Häufig verwendete Pakete

- `cvs`: Concurrent Version System; Wird zur Versionsverwaltung von Dateien verwendet. Unter <http://php.net/anoncvsvs.php> ist nachzulesen, wie man sich die aktuellste Entwicklungsversion des PHP Sourcecodes auschecken kann.

-
- cygrunsrv: NT Service; Datenbank- und Webserver bspw. laufen als sogenannten NT Service und müssen nicht bei jedem Neustart von Hand gestartet werden; Diesen Service kann man sich mit cygrunsrv auch für andere Anwendungen zu Nutze machen.
 - gcc; GNU C compiler oder GNU compiler collection ; gcc bietet eine Reihe von Compiler Tools für C.
 - weitere: libjpeg, libpng, libxslt2, libcrypt, ssh, lynx, wget etc.

18 Apache Webserver

Installation von Apache mit PHP als Modul unter Linux⁴⁴:

```
gzip -d httpd-2_0_NN.tar.gz
2. tar xvf httpd-2_0_NN.tar
3. gunzip php-NN.tar.gz
4. tar -xvf php-NN.tar
5. cd httpd-2_0_NN
6. ./configure --enable-so
7. make
8. make install
```

Apache 2.0.NN ist nun unter `/usr/local/apache2` verfügbar und kann mit `„/usr/local/apache2/bin/apachectl start“` gestartet und mit `„/usr/local/apache2/bin/apachectl stop“` angehalten werden. PHP wird nun als Apache Module kompiliert.

```
9. cd ../php4-NN
10. ./configure --with-apxs2=/usr/local/apache2/bin/apxs
11. make
12. make install
13. cp php.ini-dist /usr/local/lib/php.ini
```

Die `php.ini` Datei muss unter Umständen noch angepasst werden. Sollte sich die Datei nicht in obigem Pfad befinden, so kann der individuelle Pfad zu Kompilationszeit angegeben werden (Schritt 10):

```
--with-config-file-path=/pfad
```

14. Die Datei `httpd.conf` sollte folgende Zeilen enthalten:

```
LoadModule php4_module modules/libphp4.so
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

15. Danach muss Apache neu gestartet werden. z.B.:

```
/usr/local/apache2/bin/apachectl start
```

Ein umfangreiches Tutorial, wie man Apache unter Linux installiert hat Luc de Louw⁴⁵ zusammengestellt. Für Mac OSX bietet Marc Liyanage äusserst umfangreiche Informationen und Software, was Apache, PHP und MySQL, sowie andere Pakete anbelangt⁴⁶.

⁴⁴ Apache unter Linux, <http://ch.php.net/manual/de/install.apache2.php>.

⁴⁵ Apache HowTo, <http://www.delouw.ch/linux/apache.phtml>.

⁴⁶ Apache, PHP und MySQL unter Mac OSX, <http://www.entropy.ch/software/>.

18.1 PHP mit Erweiterungen (modules, extensions)

PHP kann mit weiteren Optionen kompiliert werden, um Module oder extensions nach den jeweiligen Bedürfnissen hinzuzufügen. Hilfreiche Informationen bieten insbesondere die Mailinglisten und FAQs.

```
./configure \  
--with-mysql \  
--with-xml \  
--with-bzip2 \  
--with-gettext \  
--with-imagick=/usr/share/ImageMagick
```

18.2 Apache unter Windows

Für die Installation von PHP und Apache greift man am besten auf die zur Verfügung gestellten Binärdistributionen von Apache 2⁴⁷ und PHP 4⁴⁸ zurück.

Es gibt verschiedene Möglichkeiten PHP als Modul von Apache zu installieren. Eine Variante stellt dabei dar, dass die php4ts.dll nach <apachedir>\modules\ kopiert wird. Anstatt die *.dll Dateien nach %SYSTEMROOT%\system32 zu koieren, oder in der php.ini Datei irgendwelche Konfigurationen vorzunehmen, kann folgendes befolgt werden:

```
# PHP as CGI  
ScriptAlias /php/ "c:/php/"  
Action application/x-httpd-php "/php/php.exe"  
  
AddType application/x-httpd-php .php .php3 .phtml  
AddType application/x-httpd-php-source .phps  
DirectoryIndex index.html index.php
```

Als Apache Modul:

```
# PHP as Apache module  
LoadModule php4_module "c:/php/sapi/php4apache2.dll"  
  
AddType application/x-httpd-php .php .php3 .phtml  
AddType application/x-httpd-php-source .phps  
DirectoryIndex index.html index.php
```

⁴⁷ Apache 2, <http://nagoya.apache.org/mirror/httpd/binaries/win32/>.

⁴⁸ PHP Binärdateien für Windows, <http://ch.php.net/downloads.php>.

18.2.1 Virtuelle Hosts

Bei mehreren virtuellen Servern nimmt httpd.conf immer den ersten Eintrag für den Aufruf von http://localhost

```
NameVirtualHost 127.0.0.1

<VirtualHost localhost.site2.ch>
  ServerAdmin admin@localhost.site2.ch
  DocumentRoot "G:/www/site2"
  ServerName localhost.site2.ch
  ErrorDocument 404 G:/www/site2/error.htm
  ErrorLog logs/site2-error_log
  CustomLog logs/site2-access_log common
</VirtualHost>
```

Konfiguration der Datei hosts.

```
127.0.0.1    localhost
127.0.0.1    localhost.site2.ch
```

Eine weitere Variante sind die Port-basierten VirtualHosts:

```
Listen 82
<Virtualhost _default_:82>
  ServerName localhost
  ServerAdmin me@localhost
  DocumentRoot "C:/www/site1/htdocs/"
  DirectoryIndex index.shtml index.html index.php
  ErrorLog logs/error.log
</Virtualhost>
```

Es gilt dabei zu beachten, dass gewisse Ports nicht frei belegbar sind, bzw. einer besonderen Aufgabe zugeteilt sind, so Port 442 für SSL Anwendungen. Die beiden gezeigten Methoden lassen sich unter Umständen auch parallel betreiben.

18.2.2 Das Modul mod_rewrite

Das Modul mod_rewrite basierend auf einer Entwicklung von Ralf Engelshall. Um bspw. suchmaschinenfreundliche URLs zu generieren, verwendet man im Ansatz folgende Zeilen, welche in die .htaccess Datei zu schreiben sind.

```
01 # http://zend.com/zend/trick/tricks-apr-2003-urls.php
02 # http://www.example.com/articles/apache/145/
03 RewriteEngine On
04 RewriteRule ^/(articles)/(.*)$ /$1.php [L,PT,E=PATH_INFO:/$2]
```

19 PHP5 Vorschau

PHP wird voraussichtlich im Verlaufe von 2003 in der nächsten Version 5 erscheinen. Einige Unterlagen sind dazu bereits verfügbar⁴⁹. Mit PHP5 geht auch die Neuauflage der Zend Engine in Version 2 einher; insbesondere im Bereich der objektorientierten Programmierung (OOP⁵⁰).

19.1 Objekte

```
01 <?php
02 /**
03  * PHP5: Funktion von __construct und __destruct.
04  */
05 class Test
06 {
07     function __construct()
08     {
09         echo 'Konstruktor __construct() wurde aufgerufen.<br />';
10     }
11     function __destruct()
12     {
13         echo 'Destruktor __destruct() wurde aufgerufen.';
14     }
15 }
16 $foo = new Test();
17 /**
18  * Ausgabe:
19  *     Konstruktor __construct() wurde aufgerufen.
20  *     Destruktor __destruct() wurde aufgerufen.
21  */
22 ?>
```

Bei PHP4 ist ein Konstruktor nicht zwingend erforderlich bzw. wird, falls benötigt, nach dem Klassennamen bezeichnet; hier bspw. `function Test() {;}`.

19.2 Funktionen

Mit PHP5 werden sogenannte abstrakte (*abstract*), private (*private*) und auch geschützte (*protected*) Funktionen eingeführt. Abstrakte Funktionen sind solche, die innerhalb einer Klasse in weiser Voraussicht reserviert werden können. Private Funktionen sind solche, die innerhalb einer Klasse sichtbar sein sollen und auch nicht vererbbar sind, geschützte Funktionen sodann solche, welche zwar vererbbar, aber nicht von ausserhalb ausführbar sind⁵¹.

⁴⁹ ZEND, Zend Engine 2, <http://www.zend.com/engine2/ZendEngine-2.0.pdf>; PHP VOLCANO, PHP5 and the future of PHP, <http://www.phpvolcano.com/articles/php5/>; ALEXANDER MERZ, PHP5, <http://www.tu-chemnitz.de/linux/tag/lt4/vortraege/material/php5.pdf>.

⁵⁰ PHP OOP, <http://www.php.net/oop> oder http://www.faqs.com/knowledge_base/view.phtml/aid/7569/fid/39.

⁵¹ PHPVolcano; Beschreibung von abstract, protectet und private Funktionen.

19.3 Variablen

Ähnlich dem Konzept bei den Funktionen gibt es auch verschiedene Variablentypen, die in Zusammenhang mit Klassen deklariert werden können. So sind *private* Variablen nur innerhalb der Klasse Foo sichtbar. *Protected* Variablen sind nur innerhalb der Klasse Foo, oder in von Foo vererbten Klassen sichtbar; jedoch nicht von ausserhalb adressierbar.

Variablen vom Typ `constant` definieren Konstanten im Zusammenhang mit einer Klasse.

```
01 <?php
02 class Foo
03 {
04     const HELLO = 'Hello there';
05 }
06 // Gibt aus: 'HELLO'
07 echo HELLO;
08 // Gibt aus: 'Hello'
09 echo Foo::HELLO;
10 // Ergibt einen fatal error
11 echo Foo::HLLLO;
12 ?>
```

19.4 Weiter Konzepte

Bezüglich der Kontrollstrukturen wie *if-else*, oder *switch*, wird neu die *try-catch* Struktur hinzukommen.

```
01 <?php
02 /**
03  * Any division by zero that occurs when executing the
04  * try block will result in execution of the catch block.
05  * Once either block completes, execution continues
06  * at the statement after the catch block.
07  */
08 // http://zend.com/tips/tips.php?id=162&single=1
09
10 class DivideByZeroException
11 {
12     private $message;
13     function __construct($message)
14     {
15         $this->message = $message;
16     }
17     function toString()
18     {
19         return 'DivideByZeroException : ' . $this->message;
20     }
21 }
22 try
23 {
24     $x = 5;
25     $y = 0;
26     echo $z = $x / $y; // $res = div($x, $y); if(!$res) throw...
27     throw new DivideByZeroException('This is wrong.');
```

Obiges Beispiel soll die *try-catch* Struktur verdeutlichen, denn Änderungen an der Zend Engine2 bleiben vorbehalten⁵².

⁵² Änderungen an der Zend Engine, http://www.php.net/ZEND_CHANGES.txt.

20 Linux

Linux ist schlicht der Inbegriff der Opensource Bewegung. Es gibt eine Vielzahl von sogenannten Distributionen⁵³. Einige wirklich interessante Möglichkeiten bietet Debian GNU/Linux⁵⁴.

Softwarepakete können dort durch ein überzeugendes Paketmanagmentssystem⁵⁵ verwaltet werden, das sich *Advanced Package Tool* nennt. Die am häufigsten verwendeten Kommandos sind dabei *apt-setup*, *apt-get upgrade*, *apt-get install*, *apt-cache search*, *apt-cache show*, *dpkg -l*, *dpkg* mit *grep*, *dpkg -L*, *dpkg -S* und andere.

21 Opensource Community

21.1 Community

Dass man das Betriebssystem Linux einfach „herunterladen“ kann, davon haben einige bereits gehört. Was aber unter dem Begriff Community und insbesondere deren im Opensource Bereich zu verstehen ist, soll hier kurz zu erläutern versucht werden.

Meist fängt es damit an, dass jemand einen Quellcode zu einem Problem unter einer dafür vorgesehenen Lizenz ins Internet stellt und sozusagen die Allgemeinheit dazu aufmuntert, an der Weiterentwicklung mit teilzunehmen. Dazu verwendet man verschiedene Kommunikations- und andere Hilfsmittel. E-Mail ist eines des Wichtigsten Hilfsmittel unter Softwareentwicklern. Sodann findet auch der *Internet Relay Chat* (IRC⁵⁶) breiten Anklang, da er Echtzeitkommunikation erlaubt.

Um Quelldateien, die unter der Mitarbeit verschiedener Personen bearbeitet werden einigermaßen übersichtlich vewalten zu können, werden sog. *Concurrent Versions Systems* (CVS⁵⁷) benutzt.

⁵³ Distrowatch gibt eine Übersicht über die verschiedenen Linux Distributionen, <http://www.distrowatch.com>.

⁵⁴ Debian GNU/Linux, <http://www.de.debian.org>.

⁵⁵ apt-get Howto, <http://newbiedoc.sourceforge.net/system/apt-get-intro.html>.

⁵⁶ IRC home, <http://www.irchelp.org>.

⁵⁷ CVS, <http://www.cvshome.org>.

21.2 Tools

- CVS
 - <http://www.wincvs.org>
- IRC
 - <http://irchelp.org/irchelp/altircfaq.html>
 - [#php.ch](irc.datacomm.ch:6667)
- Mailinglisten
 - <http://ch.php.net/mailling-lists.php>
 - <http://marc.theaimsgroup.com/?l=mysql-german>
- PHP FAQ auf deutsch
 - <http://www.dclp-faq.de>

22 Zusammenfassung

PHP ist eine einfach zu erlernende Scriptsprache mit einem unterdessen enormen Umfang an Funktionen; als Entwickler hat man dabei oft die Qual der Wahl. Ein grosser Vorteil von PHP liegt sicher darin, dass die Community auch im deutschsprachigen Raum bedeutend ist und man dadurch meist recht schnell zur Lösung eines individuellen Problems mit Hilfe von anderen Entwicklern kommt.

23 **Stichwortverzeichnis**

C

Community 49
Concurrent Versions Systems 47

D

Debian GNU/Linux 47

H

Here Document..... 15

I

Internet Relay Chat 47

L

Linux 47

M

Mac OSX 41

S

Scriptsprache 5

24 Anhang

24.1 Syntax-highlighting von Sourcecode.

```
01 <?php
02 // $Id: source.php,v 1.4 2003-04-29 23:36:45+02 urs Exp urs $
03 // http://ch.php.net/highlight-string
04
05 function highlight_code($str="")
06 {
07     $contents = highlight_string($str, true);
08     $contents = explode("<br />", $contents);
09     $trans = array(
10         '<code><font color="#000000">' => "",
11         '</font></code>' => ""
12     );
13     $ln = 1;
14
15     $cache .= "<html>\n<head>\n<style type=\"text/css\">\n";
16     $cache .= "body {font-family: courier new; font-size: 10pt; }\n";
17     $cache .= "</style>\n</head>\n<body>\n<table><tr><td>\n<code>\n";
18     foreach($contents as $line) {
19         $line = str_replace("\n", "", $line );
20         $line = strstr($line, $trans);
21         $cache .= sprintf("<font color=\"#000000\">%02s</font>&nbsp;&nbsp;&nbsp;:%s\n<br />", $ln, trim($line));
22         $ln++;
23     }
24     $cache .= "\n</code>\n</td></tr></table>\n</body>\n</html>\n";
25     return $cache;
26 }
27
28 if(stristr($_SERVER['REQUEST_METHOD'], "GET" ))
29 {
30     print <<<FOOBAR
31     <html>
32         <head>
33         <style type="text/css">
34             body {font-family: courier new; font-size: 10pt; }
35         </style>
36         </head>
37         <form method="post">
38             <textarea cols="60" rows="15" name="source"></textarea>
39             <br />
40             <input type="submit" value="convert" name="convert" />
41         </form>
42     </html>
43     FOOBAR;
44 } else {
45     $contents = stripslashes($_POST['source']);
46     $s = highlight_code($contents );
47     printf("%s<br /><a href=\"source.php\">restart</a>", $s );
48     $fp = fopen("source.html", "w+");
49     fwrite($fp, $s);
50     fclose($fp);
51 }
52 ?>
```

24.2 Dateifunktionen

basename	Extrahiert den Namen einer Datei aus einer vollständigen Pfadangabe
chgrp	Wechselt die Gruppenzugehörigkeit einer Datei
chmod	Ändert die Zugriffsrechte einer Datei
chown	Ändert den Eigentümer einer Datei
clearstatcache	Löscht den Status Cache
copy	Kopiert eine Datei
delete	Siehe unlink() oder unset()
dirname	Extrahiert den Verzeichnis-Namen aus einer vollständigen Pfadangabe
disk_free_space	Liefert den freien Speicherplatz in einem Verzeichnis
disk_total_space	Liefert die Gesamtgröße eines Verzeichnisses
diskfreespace	Ist ein Alias für disk_free_space()
fclose	Schließt einen offenen Dateizeiger
feof	Prüft, ob der Dateizeiger am Ende der Datei steht
fflush	Schreibt den Ausgabepuffer in eine Datei
fgetc	Liest das Zeichen, auf welches der Dateizeiger zeigt
fgetcsv	Liest eine Zeile von der Position des Dateizeigers und prüft diese auf Komma-Separierte-Werte (CSV)
fgets	Liest eine Zeile von der Position des Dateizeigers Liest eine Zeile von der Position des Dateizeigers und entfernt HTML Tags.
fgetss	Überprüft, ob eine Datei existiert
file_exists	Überprüft, ob eine Datei existiert
file_get_contents	Liest die gesamte Datei in einen String
file_put_contents	Write a string to a file
file	Liest eine Datei komplett in ein Array
fileatime	Liefert Datum und Uhrzeit des letzten Zugriffs auf eine Datei
filectime	Liefert Datum und Uhrzeit der letzten Änderung des Dateizeigers Inode
filegroup	Liefert die Gruppenzugehörigkeit einer Datei
fileinode	Liefert die Inode-Nummer einer Datei
filemtime	Liefert Datum und Uhrzeit der letzten Dateiänderung
fileowner	Liefert den Dateieigentümer
fileperms	Liefert die Zugriffsrechte einer Datei
filesize	Liefert die Größe einer Datei
filetype	Liefert den Typ einer Datei
flock	Portables Datei-Verriegelungs-Verfahren
fnmatch	Match filename against a pattern
fopen	Öffnet eine Datei oder URL
fpass thru	Gibt alle verbleibenden Daten eines Dateizeigers direkt aus.
fputs	Schreibt Daten an die Position des Dateizeigers
fread	Liest Binärdaten aus einer Datei Interpretiert den Input einer Datei entsprechend einem angegebenen Format
fscanf	Interpretiert den Input einer Datei entsprechend einem angegebenen Format
fseek	Positioniert den Dateizeiger
fstat	Liefert Informationen über eine Datei mit offenem Dateizeiger
ftell	Ermittelt die aktuelle Position des Dateizeigers
ftruncate	Kürzt eine Datei auf die angegebene Länge
fwrite	Schreibt Binärdaten in eine Datei
glob	Find pathnames matching a pattern
is_dir	Prüft, ob der gegebene Dateiname ein Verzeichnis ist
is_executable	Prüft, ob eine Datei ausführbar ist
is_file	Prüft, ob der Dateiname eine reguläre Datei ist
is_link	Prüft, ob der Dateiname ein symbolischer Link ist

is_readable	Prüft, ob eine Datei lesbar ist
is_uploaded_file	Prüft, ob die Datei mittels HTTP POST upgeloadet wurde
is_writable	Prüft, ob in eine Datei geschrieben werden kann
is_writeable	Prüft, ob in eine Datei geschrieben werden kann
link	Erzeugt einen absoluten Link
linkinfo	Liefert Informationen über einen Link
lstat	Liefert Informationen über eine Datei oder einen symbolischen Link.
mkdir	Erstellt ein Verzeichnis
move_uploaded_file	Verschiebt eine upgeloadete Datei an einen neuen Ort
parse_ini_file	Analysiert eine Konfigurationsdatei
pathinfo	Liefert Informationen über den Dateipfad
pclose	Schließt einen Prozess-Dateizeiger
popen	Öffnet einen Prozesszeiger
readfile	Gibt eine Datei aus
readlink	Liefert das Ziel eines symbolischen Links
realpath	Erzeugt einen kanonisch absoluten Pfadnamen
rename	Benennt eine Datei um
rewind	Setzt den Dateizeiger auf das erste Byte der Datei
rmdir	Löscht ein Verzeichnis
set_file_buffer	Alias für stream_set_write_buffer()
stat	Liefert diverse Informationen über eine Datei
symlink	Erzeugt einen symbolischen Link
tempnam	Erzeugt eine Datei mit eindeutigem Dateinamen
tmpfile	Legt eine temporäre Datei an
touch	Setzt die Zugriffs- und Modifizierungszeit einer Datei
umask	Ändert die aktuelle umask (Zugriffsrechte)
unlink	Löscht eine Datei

24.3 MySQL Funktionen in PHP

<code>mysql_affected_rows</code>	Liefert Anzahl betroffener Datensätze einer vorhergehenden MySQL Operation
<code>mysql_change_user</code>	Ändert den zur Zeit angemeldeten Benutzer der aktiven Verbindung
<code>mysql_client_encoding</code>	Returns the name of the character set
<code>mysql_close</code>	Schließt eine Verbindung zu MySQL
<code>mysql_connect</code>	Öffnet eine Verbindung zu einem MySQL-Server
<code>mysql_create_db</code>	Anlegen einer MySQL-Datenbank
<code>mysql_data_seek</code>	Bewegt den internen Ergebnis-Zeiger
<code>mysql_db_name</code>	Liefert Ergebnisdaten
<code>mysql_db_query</code>	Absetzen einer Anfrage an die Datenbank
<code>mysql_drop_db</code>	Löschen einer Datenbank
<code>mysql_errno</code>	Rückgabe einer Fehlermeldungsnummer einer zuvor ausgeführten MySQL Operation
<code>mysql_error</code>	Liefert den Fehlertext der zuvor ausgeführten MySQL Operation
<code>mysql_escape_string</code>	Maskiert einen String zur Benutzung in <code>mysql_query</code> .
<code>mysql_fetch_array</code>	Liefert einen Datensatz als assoziatives Array, als numerisches Array oder beides
<code>mysql_fetch_assoc</code>	Liefert einen Datensatz als assoziatives Array
<code>mysql_fetch_field</code>	Liefert ein Objekt mit Feldinformationen aus einem Anfrageergebnis
<code>mysql_fetch_lengths</code>	Liefert die Länge eines jeden Feldes in einem Ergebnis
<code>mysql_fetch_object</code>	Liefert eine Ergebniszeile als Objekt
<code>mysql_fetch_row</code>	Liefert einen Datensatz als indiziertes Array
<code>mysql_field_flags</code>	Liefert die Flags eines Feldes in einem Anfrageergebnis
<code>mysql_field_len</code>	Liefert die Länge des angegebenen Feldes
<code>mysql_field_name</code>	Liefert den Namen eines Feldes in einem Ergebnis
<code>mysql_field_seek</code>	Setzt den Ergebniszeiger auf ein bestimmtes Feldoffset
<code>mysql_field_table</code>	Liefert den Namen der Tabelle, die das genannte Feld enthält
<code>mysql_field_type</code>	Liefert den Typ eines Feldes in einem Ergebnis
<code>mysql_free_result</code>	Gibt belegten Speicher wieder frei
<code>mysql_get_client_info</code>	Liefert MySQL Clientinformationen
<code>mysql_get_host_info</code>	Liefert MySQL Host Informationen
<code>mysql_get_proto_info</code>	Liefert MySQL Protokollinformationen
<code>mysql_get_server_info</code>	Liefert MySQL Server Informationen
<code>mysql_info</code>	Liefert Informationen über die zuletzt ausgeführte Anfrage zurück
<code>mysql_insert_id</code>	Liefert die ID einer vorherigen INSERT-Operation
<code>mysql_list_dbs</code>	Auflistung der verfügbaren Datenbanken auf einem MySQL Server
<code>mysql_list_fields</code>	Listet MySQL Ergebnisfelder auf
<code>mysql_list_processes</code>	Zeigt die MySQL Prozesse an
<code>mysql_list_tables</code>	Listet Tabellen in einer MySQL Datenbank auf
<code>mysql_num_fields</code>	Liefert die Anzahl der Felder in einem Ergebnis
<code>mysql_num_rows</code>	Liefert die Anzahl der Datensätze im Ergebnis
<code>mysql_pconnect</code>	Öffnet eine persistente Verbindung zum MySQL Server
<code>mysql_ping</code>	Ping a server connection or reconnect if there is no connection
<code>mysql_query</code>	Sendet eine Anfrage an MySQL
<code>mysql_real_escape_string</code>	Escapes special characters in a string for use in a SQL statement, taking into account the current charset of the connection.
<code>mysql_result</code>	Liefert Ergebnis
<code>mysql_select_db</code>	Auswahl einer MySQL Datenbank
<code>mysql_stat</code>	Zeigt den momentanen Serverstatus an
<code>mysql_tablename</code>	Liefert den Namen einer Tabelle
<code>mysql_thread_id</code>	Zeigt die aktuelle Thread ID an
<code>mysql_unbuffered_query</code>	SQL Anfrage an MySQL, ohne Ergebniszeilen abzuholen und zu puffern.

24.4 Links

24.4.1 Portale

- Referenz verschiedener PHP Artikel, <http://www.troobloo.com/tech/php.toc.shtml>
- Plattform für PHP, <http://www.onlamp.com/php/>
- Zend Tutorials, <http://zend.com/zend/tut/>
- PHPpatterns - Tutorials, <http://www.phppatterns.com>
- DynamicWebpages - Tutorials, <http://dynamicwebpages.de>
- PHP Center, <http://phpcenter.de>
- PHP Referenz, <http://php-groupies.de/php/>

24.4.2 Konferenzen

- Int. PHP Conference, <http://www.php-conference.com>
- PHP Con, <http://www.php-con.com>
- PHP Gipfeltreffen, <http://www.php-gipfeltreffen.ch>

24.4.3 Magazine

- PHP Architect, <http://www.phparch.com>
- PHP Magazin, <http://www.php-mag.de>

24.4.4 Foren

- PHPUG, <http://phpug.ch>
- dclp-faq, <http://www.dclp-faq.de>

24.4.5 Tutorials

- PHP Einführung
 - <http://123.koehntopp.de/kris/artikel/php3-einfuehrung/>
 - <http://print-www.informatik.uni-hamburg.de/Dokumentation/php.phtml>
 - http://drweb.de/programmierung/php_kurs.shtml
- SQL Tutorial
 - <http://www.w3schools.com/sql/>