

### **Bibliografische Information der Deutschen Nationalbibliothek**

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie. Detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

ISBN 978-3-8266-5939-3

1. Auflage 2008

Alle Rechte, auch die der Übersetzung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder einem anderen Verfahren) ohne schriftliche Genehmigung des Verlages reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden. Der Verlag übernimmt keine Gewähr für die Funktion einzelner Programme oder von Teilen derselben. Insbesondere übernimmt er keinerlei Haftung für eventuelle aus dem Gebrauch resultierende Folgeschäden.

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Printed in Austria

© Copyright 2008 by REDLINE GMBH, Heidelberg

[www.mitp.de](http://www.mitp.de)

Lektorat: Ernst-Heinrich Profener

Satz: III-satz, Husby, [www.drei-satz.de](http://www.drei-satz.de)

# Inhaltsverzeichnis

	<b>Vorwort</b> .....	19
<b>Teil I</b>	<b>Grundlagen im Überblick</b> .....	23
<b>1</b>	<b>Hinweise zum Buch und zur CD-ROM</b> .....	25
1.1	Die Erläuterungen .....	25
1.2	Die Struktur .....	25
1.3	Begriffe .....	26
1.4	Quellen .....	27
1.4.1	Quellen zu PHP .....	27
1.4.2	W3C-Standards .....	27
1.4.3	Weitere Quellen .....	28
1.5	Schreibweisen .....	29
1.6	Abbildungen .....	30
1.7	Quelltexte und Dateien auf CD-ROM .....	31
1.8	Arbeitsgrundlagen .....	32
1.9	Die Beispiele ausführen .....	33
1.9.1	Mit XAMPP unter Linux arbeiten .....	33
1.9.2	Mit XAMPP unter Windows arbeiten .....	34
1.9.3	Einige sind Webprogramme .....	35
<b>2</b>	<b>Der PHP-Interpreter</b> .....	37
2.1	Varianten des PHP-Interpreters .....	37
2.2	Weitere hilfreiche Optionen .....	37
2.3	Die »richtige« Programmiersprache .....	39
2.3.1	Kriterien .....	39
2.3.2	Scriptsprachen für Anwendungen .....	40
2.3.3	Immer mehr Entscheidungen zur Laufzeit .....	41
2.3.4	Wenn sich die Unterlage ändert .....	41
2.3.5	Open Source .....	42
2.3.6	Moderierte Herstellung .....	42

2.3.7	Objektorientierte Programmierung .....	43
2.3.8	Und das Web?.....	44
<b>3</b>	<b>Die Programmiersprache PHP.....</b>	<b>45</b>
3.1	Der Programmrahmen .....	45
3.2	Abarbeitungs-Modi .....	45
3.3	Bezeichnungen und Anweisungen .....	46
3.4	Ausgabeanweisungen .....	47
3.5	Variablen und Konstanten .....	48
3.6	Typen und Werte .....	49
3.6.1	Basistypen .....	49
3.6.2	Literale.....	49
3.6.3	Einige Operatoren .....	50
3.7	Zeichenketten .....	51
3.7.1	Zeichenketten und Quotes .....	51
3.7.2	Steuerzeichen in Zeichenketten .....	51
3.7.3	Zeichenketten verbinden .....	52
3.8	Ausdrücke und Werte .....	53
3.9	Funktionen.....	54
3.9.1	Sichtbarkeit.....	54
3.10	Spezielles bei Funktionen.....	56
3.10.1	Vorbelegte Parameter.....	56
3.10.2	Beliebige Zahl von Argumenten .....	56
3.10.3	Funktionsaufruf über Variablen .....	56
<b>4</b>	<b>Arbeiten mit Arrays.....</b>	<b>57</b>
4.1	Eigenschaften.....	57
4.1.1	Schlüssel und Element.....	57
4.1.2	Array-Elemente ansprechen.....	60
4.1.3	Einige Array-Funktionen .....	62
4.2	Arrays in andere Strukturen konvertieren.....	63
4.2.1	Arrays und Variablen .....	63
4.2.2	Arrays und Zeichenketten .....	65
<b>5</b>	<b>Kopie und Referenz.....</b>	<b>67</b>
5.1	Werte kopieren.....	67
5.1.1	Kopieren einer Variablen.....	67
5.1.2	Call by Value .....	68
5.1.3	Return by Value .....	69

5.2	Referenzen statt Kopien .....	70
5.2.1	Referenzen von Variablen .....	70
5.2.2	Call by Reference .....	71
5.2.3	Return by Reference .....	72
5.3	Konsequenzen durch Referenzen .....	73
5.3.1	Variablen löschen .....	73
5.3.2	Array-Zuweisung: Kopie oder Referenz? .....	74

## **Teil II Start frei für die OOP .....** 75

<b>6</b>	<b>PHP objektorientiert - der Anfang .....</b>	<b>77</b>
6.1	Der Begriff »Objekt« .....	77
6.1.1	Herkunft .....	77
6.1.2	Objektresidenz: der Hauptspeicher .....	78
6.1.3	Datentyp: Größe eines Objekts .....	79
6.1.4	Die OOP lenkt den Fokus auf den Typ .....	80
6.1.5	Begriffe .....	81
6.2	Die erste Klasse .....	82
6.2.1	\$c ist \$a plus \$b .....	82
6.2.2	Ein Objekt der Klasse verwenden .....	82
6.2.3	Öffentlich zugänglich .....	83
6.2.4	Ausgabe des Ergebnisses .....	84
6.3	Mehrere Instanzen einer Klasse .....	84
6.3.1	Unabhängigkeit .....	84
6.3.2	Was ist eigentlich \$this? .....	85
6.3.3	API und Implementierung .....	86
6.4	Autoloading .....	87
6.4.1	Klassendefinitionen automatisch einbinden .....	87
6.4.2	Einheitliche Namensmuster sind Voraussetzung .....	87
<b>7</b>	<b>Konstruktor und Destruktor .....</b>	<b>89</b>
7.1	Der »Lebenslauf« eines Objekts .....	89
7.2	Ausgaben machen es deutlich .....	90
7.3	Ein Destruktor zerstört ein Objekt – wirklich? .....	92
7.4	Parameter im Konstruktor .....	93
7.5	Der Standardkonstruktor .....	94

<b>8</b>	<b>File-Klassen</b> .....	97
8.1	Privat, Geschützt, Öffentlich. ....	97
8.1.1	Eine Textdatei lesen .....	97
8.1.2	Klasse FileBase .....	98
8.1.3	API zur Klasse FileBase .....	100
8.2	Vererben .....	101
8.3	Private bleibt private .....	102
8.4	Erweitern durch Vererben. ....	102
8.4.1	Zusätzliche Methoden .....	102
8.4.2	Erweiterung zum Schreiben .....	103
8.5	Überschreiben nach Vererben .....	104
8.5.1	Der Konstruktor wird erweitert .....	104
8.5.2	Der Gültigkeitsbereichsoperator .....	105
8.5.3	Implizite Aufrufe .....	106
8.6	Überschreiben genauer betrachtet .....	108
<b>9</b>	<b>Objekt- und Klassenzugriffe</b> .....	111
9.1	static-Member .....	111
9.1.1	Ordnung ist nicht immer einfach .....	111
9.1.2	Als static deklarierte Variablen .....	111
9.1.3	Als static deklarierte Member .....	112
9.1.4	Objekte nummerieren mit static-Member .....	113
9.1.5	Objektzählung nach Ableitung .....	115
9.1.6	Jedem sein eigenes static .....	116
9.2	static-Methoden .....	118
9.3	Klassenzugriff auf reguläre Methoden. ....	119
<b>10</b>	<b>Bindungen und Ähnlichkeiten</b> .....	121
10.1	Objekt als Referenz und Kopie .....	121
10.1.1	Zur Erinnerung .....	121
10.1.2	Objekt im Test .....	122
10.1.3	Wichtige Hinweise für PHP4-Programmierer. ....	122
10.1.4	Das PHP4-Beispiel in PHP5 .....	123
10.1.5	Kopie eines Objekts bei PHP5 .....	123
10.1.6	Endgültige Klärung .....	124
10.2	Objekte verwenden .....	126
10.2.1	Objekte in Klassen .....	126
10.2.2	Der Operator instanceof .....	127
10.2.3	Späte dynamische Bindung .....	128

<b>II</b>	<b>Knoten und Bäume</b> . . . . .	131
II.1	Warum ausgerechnet Bäume? . . . . .	131
II.2	Darstellung von Bäumen . . . . .	132
II.3	Merkmale eines Knotens . . . . .	132
II.4	Klasse Knoten konstruieren . . . . .	133
II.4.1	Initialisierung . . . . .	134
II.4.2	Den übergeordneten Knoten anbinden . . . . .	135
II.4.3	Untergeordnete Knoten hinzufügen . . . . .	135
II.4.4	Einen untergeordneten Knoten selbst erzeugen . . . . .	136
II.4.5	Weiterer Service . . . . .	137
II.4.6	Ein Zwischentest . . . . .	138
II.5	Selbstverwaltung . . . . .	139
II.5.1	Vom Knoten zum Baum . . . . .	139
II.5.2	Traversierung . . . . .	139
II.5.3	Eine Baumdarstellung . . . . .	141
II.5.4	Knoten finden . . . . .	143
II.5.5	Der Weg nach oben . . . . .	144
<b>12</b>	<b>Verwaltung eines Verzeichnisbaums</b> . . . . .	147
12.1	Die Klasse DirReader . . . . .	147
12.1.1	Vorbereitung . . . . .	147
12.1.2	Einer für alle . . . . .	147
12.1.3	Klasse DirReader . . . . .	148
12.1.4	Baumdarstellung für Verzeichnisse . . . . .	149
12.1.5	Test: Wenig Programm mit großer Wirkung . . . . .	150
12.1.6	Verzeichniswechsel und weitere Features . . . . .	151
12.2	Verlorene Objekte . . . . .	152
12.2.1	Verweise ohne Bedenken? . . . . .	152
12.2.2	Eine Klasse A zum Testen . . . . .	153
12.2.3	Zuweisungs- und Versuchsteil . . . . .	153
12.2.4	Der erste Versuch . . . . .	154
12.2.5	Der zweite Versuch . . . . .	155
12.2.6	Dritter und letzter Versuch . . . . .	156
12.2.7	Schlussfolgerung . . . . .	158
12.3	Knotenbaum demontieren . . . . .	158
12.3.1	Methode rmnodes() . . . . .	158
12.3.2	Entfernen unterer Knoten testen . . . . .	159
12.3.3	Entfernen des kompletten Baums . . . . .	160

<b>13</b>	<b>Abstrakte Klassen.</b>	161
13.1	Eine Methode für den Durchlauf	161
13.1.1	Die Methode tree() aufbohren	161
13.1.2	Informationen für den Durchlauf	162
13.1.3	Der Quellcode von walker()	163
13.2	Abstrakte Methoden und Klassen.	165
13.3	Die Klasse TreeExporter	167
13.4	Baumdarstellung, die zweite.	168
13.4.1	Anwendungsklasse TextTree	168
13.4.2	Ergänzung mit Verzeichnisnamen	168
13.4.3	Nichts zu tun für postcall()	169
13.4.4	Ist der aktuelle Knoten der oberste?	169
13.4.5	Ebenenabhängiger Vorspann	170
13.4.6	Modusabhängiger Teil	170
13.4.7	Die Endinformation	171
13.4.8	Am Ziel: TextTree in der Anwendung	171
13.4.9	Diskussionen	172
13.5	Schnell erstellte Anwendungen	173
13.5.1	Baumdarstellung in einem HTML-Dokument	173
13.5.2	Verzeichnisstruktur zur Datenbank	177
13.5.3	Die Klasse SqlInserts	178
13.5.4	Fazit.	179
<b>14</b>	<b>Interfaces und SPL</b>	181
14.1	Interfaces	181
14.1.1	Mehrfachvererbung	181
14.1.2	Von Interfaces ableiten	182
14.1.3	Interface als Basis	183
14.2	Das Interface Countable	184
14.2.1	Wo ist die SPL?	184
14.2.2	Die Funktion count()	185
14.2.3	Interface Countable implementieren	186
14.2.4	Ohne Countable geht es nicht	187
14.2.5	Eine sinnvolle Implementierung von Countable	187
14.3	Container und Iteratoren	189
14.3.1	Abstrakte Datenstrukturen	189
14.3.2	Das Interface Iterator	190
14.3.3	Die Anwendung steuert den Durchlauf	191
14.3.4	Extern, intern und IteratorAggregate	191

14.3.5	Das Interface <code>ArrayAccess</code> . . . . .	192
14.3.6	Die Klasse <code>ArrayIterator</code> . . . . .	193
14.3.7	Die Klasse <code>ArrayObject</code> . . . . .	193
14.3.8	Die Klasse <code>RecursiveArrayIterator</code> . . . . .	194
14.3.9	Beispiele zur SPL . . . . .	194
<b>15</b>	<b>ArrayAccess im Einsatz.</b> . . . . .	<b>195</b>
15.1	Eine Demo-Implementierung . . . . .	195
15.2	Formularwerte mit <code>ArrayAccess</code> abrufen . . . . .	197
15.2.1	Der Plan: Formularwerte abrufen. . . . .	197
15.2.2	Die Prüfungsfunktionen . . . . .	197
15.2.3	Das Formular . . . . .	198
15.2.4	Die Zuordnung der Prüfmethode. . . . .	199
15.2.5	<code>Array</code> -Zugriff auf die geprüften Daten . . . . .	201
15.2.6	Die Anwendung im Webprogramm. . . . .	204
15.2.7	Fazit und Ausblick . . . . .	206
<b>16</b>	<b>Iterator im Einsatz.</b> . . . . .	<b>207</b>
16.1	Eine Demo-Implementierung . . . . .	207
16.2	Ein Iterator für <code>FormArray</code> -Objekte. . . . .	209
16.2.1	Die Klasse für einen externen Iterator . . . . .	209
16.2.2	Die Funktionsweise . . . . .	210
16.2.3	Formularwerte vom Iterator abholen . . . . .	211
16.2.4	Iterator per <code>IteratorAggregate</code> . . . . .	212
16.2.5	Verwenden von <code>FormArrayAgg</code> . . . . .	213
<b>17</b>	<b>Iterator-Klassen der SPL</b> . . . . .	<b>215</b>
17.1	Die Klasse <code>ArrayIterator</code> . . . . .	215
17.1.1	Vorbemerkungen . . . . .	215
17.1.2	Eine Demo-Anwendung . . . . .	215
17.1.3	<code>ArrayIterator</code> im Formularbeispiel . . . . .	216
17.1.4	<code>ArrayIterator</code> wird vererbt . . . . .	217
17.1.5	Methoden überschreiben, wenn nötig . . . . .	218
17.1.6	Die Anwendung im Formularprogramm. . . . .	219
17.2	Die Klasse <code>ArrayObject</code> . . . . .	219
17.2.1	Vorbemerkungen . . . . .	219
17.2.2	Eine Demo-Anwendung . . . . .	219
17.2.3	Die Anwendung im Formularprogramm. . . . .	220
17.3	Iterieren über mehrere Ebenen . . . . .	221



17.3.1	Der RecursiveArrayIterator .....	221
17.3.2	Der RecursiveIteratorIterator .....	224
17.3.3	Der RecursiveDirectoryIterator .....	226
17.3.4	Der ParentIterator .....	228
17.4	Zum Schluss .....	229
<b>18</b>	<b>DOM in PHP5 .....</b>	<b>231</b>
18.1	Vorbemerkungen .....	231
18.2	W3C-Dokumente zu DOM .....	232
18.3	Interface Definition Language (IDL) .....	233
18.4	Weitere Bezeichnungen und ihre Implementierung .....	234
18.5	Die Baumstruktur im DOM .....	236
18.5.1	XML-Dokument als DOM-Knotenbaum .....	236
18.5.2	Knotenobjekte nach IDL .....	237
18.6	Die DOM-Klassen in PHP5 .....	239
18.6.1	Kurze Übersicht .....	239
18.6.2	Das Beispieldokument wird zum DOMDocument-Objekt ..	239
18.6.3	DOM-Erweiterung und Unicode .....	240
18.7	Den DOM-Baum abbilden .....	240
18.7.1	Die Klasse DomReader .....	240
18.7.2	Textausgabe mit Klasse TextDom .....	242
18.7.3	Der DOM-Baum des Beispieldokuments .....	244
<b>19</b>	<b>DOM-Montage .....</b>	<b>247</b>
19.1	Eine DOM Struktur sichern .....	247
19.1.1	XML-Dokument, ganz oder teilweise sichern .....	247
19.1.2	Direkt in eine Datei speichern .....	248
19.1.3	Ausgabe als HTML .....	248
19.2	Eine DOM-Struktur erstellen .....	249
19.2.1	XML-Version und Zeichensatz .....	249
19.2.2	Andere Eigenschaften .....	250
19.3	Laden von XML-Dokumenten .....	250
19.3.1	Das Laden mit Optionen beeinflussen .....	251
19.4	Eine DOM-Struktur verändern .....	252
19.4.1	Neue Knoten hinzufügen .....	252
19.4.2	Knoten importieren .....	252
19.5	Knoten aus der DOM-Struktur entfernen .....	255
19.6	Inhalte von Knoten bearbeiten .....	256
19.6.1	Member für Inhalte .....	256

19.6.2	Werte direkt zuweisen . . . . .	256
19.6.3	Methoden zur Wertänderung . . . . .	259
19.6.4	Attributwerte ändern . . . . .	260
19.7	Knoten suchen und finden . . . . .	261
19.7.1	Methoden zum Finden . . . . .	261
19.7.2	Elementknoten anhand des Namens finden . . . . .	261
19.7.3	Attributknoten finden . . . . .	262
19.7.4	Elementknoten anhand eines ID-Werts finden . . . . .	263
20	<b>XPath-Objekte . . . . .</b>	265
20.1	Von DOM zu XPath . . . . .	265
20.1.1	Ein DOMXPath-Objekt bilden. . . . .	265
20.1.2	Knotensets mit Query erfragen. . . . .	266
20.1.3	Die XPath-Klasse kocht auch nur mit DOM . . . . .	267
20.2	Such- und Abfragesprache XPath . . . . .	268
20.2.1	Sehr vertraut: Pfade . . . . .	268
20.2.2	Eine Testumgebung. . . . .	269
20.2.3	Elementknoten finden . . . . .	271
20.2.4	Attribute finden . . . . .	272
20.2.5	Knotentests . . . . .	272
20.2.6	Schritt für Schritt, nicht nur nach unten . . . . .	273
20.2.7	Knoten auf Achse . . . . .	274
20.2.8	Prädikate . . . . .	276
21	<b>Datentransformation mit XSLT . . . . .</b>	279
21.1	Vorbemerkungen. . . . .	279
21.2	Eine XML-Sprache für Übungszwecke . . . . .	279
21.2.1	XML-Elemente für den Anfang. . . . .	279
21.2.2	Ein Titel muss sein. . . . .	280
21.2.3	Laden des Testdokuments in eine DOM-Struktur. . . . .	281
21.3	Transformieren mit XSLT . . . . .	281
21.3.1	Ein XSLT-Prozessor wird gebraucht. . . . .	281
21.4	XSLT-Templates . . . . .	284
21.4.1	Eine Transformation ausführen . . . . .	284
21.4.2	Ein Template für ein Knotenset . . . . .	284
21.4.3	Traversierung fortführen. . . . .	286
21.4.4	Immer im Kontext . . . . .	287
21.4.5	Implizite Templates . . . . .	287
21.4.6	Werte holen . . . . .	287

21.4.7	Knotensets für Ziele . . . . .	288
21.4.8	Transformation in HTML . . . . .	289
<b>22</b>	<b>XBuch als Online-Buch . . . . .</b>	<b>291</b>
22.1	Vorbereitungen . . . . .	291
22.2	Templates für HTML-Dokumente . . . . .	291
22.3	Ein Template für den Dokumentknoten . . . . .	292
22.4	Templates für Dokumentteile. . . . .	293
22.5	Unterscheidung gleichbenannter Elemente . . . . .	294
22.6	Noch mehr Dynamik. . . . .	295
22.6.1	CSS-Klassenname in div . . . . .	295
22.6.2	Die Überschrift h . . . . .	296
22.6.3	Ein positiver Nebeneffekt. . . . .	298
22.6.4	Alles mit einem Template . . . . .	299
22.7	Dynamische Inhaltsverzeichnisse . . . . .	300
22.7.1	Template für Element buch vorbereiten. . . . .	300
22.7.2	Ein Anfang: Verzeichnis für die erste Ebene . . . . .	301
22.7.3	Inhaltsverzeichnis mit benanntem Template erzeugen . . . . .	303
22.7.4	Der rekursive Aufruf . . . . .	305
<b>Teil III</b>	<b>Übungen und Miniprojekte . . . . .</b>	<b>307</b>
<b>23</b>	<b>Die Pythagoras-Klasse . . . . .</b>	<b>309</b>
23.1	Aufgabenstellung. . . . .	309
23.1.1	Ein Modell. . . . .	309
23.1.2	Das Klassenziel. . . . .	310
23.2	Ein Lösungsvorschlag . . . . .	311
23.2.1	Member, Methoden und API . . . . .	311
23.2.2	Komplettierung von Dreieckseigenschaften. . . . .	312
23.2.3	Initialisierung . . . . .	313
23.2.4	API-Methoden. . . . .	313
23.3	Testprogramm . . . . .	314
23.4	Selbstständige Übungen . . . . .	316
<b>24</b>	<b>Finale: Eine endgültige Klasse . . . . .</b>	<b>317</b>
24.1	Ein Objekt für die MySQL-Datenbank. . . . .	317
24.1.1	Motivierung. . . . .	317
24.1.2	Recherche . . . . .	317

24.1.3	Fokus . . . . .	317
24.1.4	Die Datenbanksitzung . . . . .	318
24.2	Aufbau der MySQL-Klasse . . . . .	318
24.2.1	Parameter zum Aufbau einer Sitzung . . . . .	318
24.2.2	Eröffnen und Schließen einer Datenbanksitzung . . . . .	319
24.2.3	Fehler durch Störungen . . . . .	319
24.2.4	SQL-Statements ausführen . . . . .	320
24.2.5	Die Cursor-Technik . . . . .	320
24.2.6	Das API . . . . .	322
24.3	Erster Entwurf . . . . .	323
24.3.1	Unser Beispiel als finale Klasse . . . . .	323
24.4	Das Klassendesign ausbauen . . . . .	325
24.4.1	Einrichten eines Fehlerprotokolls . . . . .	326
24.4.2	Fehlende Verbindungsdaten . . . . .	326
24.4.3	Verbindung zur Datenbank nicht möglich . . . . .	326
24.4.4	Datenbestand nicht verfügbar . . . . .	327
24.4.5	Statement nicht ausführbar . . . . .	327
24.4.6	Statement vorbehandeln . . . . .	327
24.5	Eigenschaften der Datenbank berücksichtigen . . . . .	328
24.5.1	Ein neuer Identifier wurde angelegt . . . . .	328
24.5.2	Korrekte Rückgabe für jedes Statement . . . . .	329
24.6	Die Datenbankklasse testen . . . . .	330
24.6.1	Einen Datenbestand einrichten . . . . .	330
24.6.2	Eine Tabelle anlegen . . . . .	331
25	<b>Der XPath-Viewer . . . . .</b>	<b>333</b>
25.1	Ein Lösungsansatz . . . . .	333
25.2	Arbeiten mit dem XPath-Viewer . . . . .	333
25.3	Programmbeschreibung . . . . .	336
25.3.1	Der Programmrahmen . . . . .	336
25.3.2	Auswertung der Formulare Daten . . . . .	337
25.3.3	Übungsdatei in DOM und XPATH-Objekte überführen . . . . .	337
25.3.4	Die Ausgabe des Formulars . . . . .	338
25.3.5	Abfrage ausführen und Darstellungen erzeugen . . . . .	338
25.4	Klasse DomReaderExt . . . . .	339
25.5	Die Klasse XPQueryDom . . . . .	341
25.5.1	Der Konstruktor . . . . .	341
25.5.2	Enthalten im Abfrageergebnis? . . . . .	341
25.5.3	Ein Objekt kopieren . . . . .	342

25.5.4	Die Informationsdarstellung . . . . .	342
25.5.5	Die Gewinnung der Knoteninformation . . . . .	344
25.5.6	Attributinformationen holen . . . . .	345
<b>26</b>	<b>Miniprojekt Fotoalbum . . . . .</b>	<b>349</b>
26.1	Ein Lösungsansatz . . . . .	349
26.1.1	Alles auf CD-ROM . . . . .	349
26.1.2	Aufteilung der Fotosammlung . . . . .	349
26.2	Das Konzept . . . . .	352
26.2.1	Seitenaufbau . . . . .	352
26.2.2	Die Quellen . . . . .	352
26.3	Überblick zur Beispiellösung . . . . .	354
26.3.1	Die Konfiguration . . . . .	356
26.3.2	Von den Ausgangsdaten zu XML . . . . .	357
26.3.3	Generierung der HTML-Seiten . . . . .	358
26.4	Die Klasse DirLoader . . . . .	360
26.4.1	Übersicht . . . . .	360
26.4.2	Dateien und Verzeichnisse lesen . . . . .	361
26.4.3	Verzeichnis- und Dateinamen in Folge festhalten . . . . .	361
26.4.4	Folgen aktualisieren . . . . .	366
26.4.5	Bildfolge-Informationen gewinnen . . . . .	369
26.5	Die Klasse DirFindGal . . . . .	371
26.6	Die Klasse CreateXML . . . . .	373
26.6.1	Die Struktur des XML-Dokuments . . . . .	373
26.6.2	Informationen zum Album zusammenstellen . . . . .	376
26.6.3	Galerie-Informationen . . . . .	377
26.6.4	Bild-Informationen . . . . .	379
26.7	Generierung der Startseite . . . . .	381
26.7.1	Vorbereitungen zur Transformation . . . . .	381
26.7.2	Template für die Startseite . . . . .	382
26.7.3	Template für die Galerieeinträge . . . . .	383
26.8	Generierung der Galerieseiten . . . . .	384
26.8.1	Ausführung im Hauptprogramm . . . . .	384
26.8.2	Das Problem . . . . .	385
26.8.3	Die Lösung mit registrierter PHP-Funktion . . . . .	386
26.8.4	Weitere Templates . . . . .	387
26.9	Generierung der Bildseiten . . . . .	389
26.9.1	Anweisungen im Hauptprogramm . . . . .	389
26.9.2	Das Template zum Generieren der Seite . . . . .	390

26.9.3	Ein benanntes Template .....	391
26.I0	Die Methode DirConf::xslth() .....	393
26.II	Abschluss,Varianten und Ausblick .....	394
<b>27</b>	<b>Grafikbearbeitung</b> .....	<b>397</b>
27.1	Die GD-Bibliothek .....	397
27.2	Funktionsauswahl .....	398
27.3	Basisklasse für Pixelbilder .....	400
27.3.1	Member und Konstruktor .....	400
27.3.2	Erstellen eines neuen Images .....	401
27.3.3	Das Management der Farben .....	402
27.3.4	Ein erster Test. ....	404
27.3.5	Die Bildfläche füllen .....	406
27.3.6	Archivierung .....	407
27.3.7	Speichern und Laden .....	409
27.3.8	Kopieren von Bildern und Bildteilen .....	413
27.3.9	Füll- und Kopiertest .....	414
27.4	Linien, Ellipsen und Rechtecke .....	415
27.5	Text in der Grafik .....	418
27.5.1	Anpassungsprobleme .....	418
27.5.2	Methoden für True-Type-Text .....	418
27.5.3	Test der Textklasse .....	420
27.6	Bildbearbeitung durch Filtern .....	421
27.6.1	Das Prinzip des Convolution-Filters. ....	422
27.6.2	Bibliotheksfunktion zur Filterung .....	423
27.6.3	Ein Tool zum Ausprobieren der Filterung. ....	424
27.7	Mehrere Bearbeitungsschritte .....	428
27.7.1	Bildgröße verändern .....	428
27.7.2	Verkleinern, Kopieren und Schärfen .....	430
27.8	Farbverläufe, Aufgabenstellung .....	431
27.8.1	Modell zum Farbverlauf .....	431
27.8.2	Ein Pixel lesen und schreiben .....	432
27.8.3	Farbverläufe realisieren .....	433
27.9	Iteratoren für Images .....	433
27.9.1	Wahlweise x oder y .....	433
27.9.2	Methoden für Countable und SeekableIterator .....	435
27.9.3	Methoden für ArrayAccess .....	436
27.9.4	Weitere Methoden .....	438
27.I0	Flächen iterieren .....	438

27.10.1	Farbverlauf mit zwei Iteratoren . . . . .	438
27.10.2	Iteration über nicht gerade Linien . . . . .	440
27.10.3	Farbverlauf von einer Linie ableiten . . . . .	440
27.II	Eine eigene Convolution-Filterung . . . . .	443
27.II.1	Das Prinzip . . . . .	443
27.II.2	Ein Matrix-Iterator . . . . .	444
27.II.3	Lese- und Schreibpositionen . . . . .	447
27.II.4	Die Methode MatrixIter::filter() . . . . .	449
27.II.5	Test mit einer 5x5-Matrix . . . . .	451
27.II.6	Übungen und Ausblick . . . . .	453
27.I2	Die Klasse ColOp . . . . .	453
28	<b>XSLT-Funktionen im Fokus</b> . . . . .	457
28.1	Das Ausgangsdokument . . . . .	457
28.2	Ansätze zur Lösung . . . . .	457
28.3	Vom XML-Dokument zum Formular . . . . .	459
28.3.1	Das Hauptprogramm . . . . .	459
28.3.2	Die Formulardaten fließen ein . . . . .	461
28.4	Die Rücktransformation ins Original . . . . .	464
28.5	Transformation in die erweiterte Form . . . . .	465
28.5.1	Die Templates im XSLT-Dokument . . . . .	465
28.5.2	Wirkungsweise der registrierten PHP-Funktion goelement() . . . . .	467
28.6	Transformation in HTML . . . . .	468
28.6.1	Der äußere Teil des HTML-Dokuments . . . . .	468
28.6.2	Benanntes Template mit Parametern aufrufen . . . . .	469
28.6.3	Benanntes Template zur Wiederverwendung . . . . .	470
29	<b>OOP Specials</b> . . . . .	473
29.1	Type Hinting . . . . .	473
29.2	Das Überladen . . . . .	473
29.2.1	Überladen von Methoden . . . . .	473
29.2.2	Überladen von Methoden nach dem Handbuch . . . . .	475
29.2.3	Überladen von Operatoren . . . . .	476
29.2.4	Überladen von Mitgliedern . . . . .	476
29.3	Magische Funktionen und Methoden . . . . .	478
	<b>Stichwortverzeichnis</b> . . . . .	481