

Inhaltsübersicht

Über den Autor	25
Über die Fachlektoren	27
Einleitung	29
Danksagungen	31
1 So funktioniert Drupal	33
2 Ein Modul schreiben	45
3 Hooks, Aktionen und Trigger	71
4 Das Menüsystem	97
5 Mit Datenbanken arbeiten	129
6 Mit Benutzern arbeiten	159
7 Mit Nodes arbeiten	183
8 Das Theme-System	213
9 Mit Blöcken arbeiten	255
10 Die Formular-API	273
11 Benutzereingaben bearbeiten: Das Filtersystem	329
12 Inhalte durchsuchen und indizieren	345
13 Mit Dateien arbeiten	365
14 Mit Taxonomien arbeiten	385
15 Caching	409
16 Sitzungen (Sessions)	427
17 jQuery	439
18 Lokalisierung und Übersetzung	471
19 XML-RPC	507
20 Sicheren Code schreiben	521
21 Bewährte Vorgehensweisen in der Entwicklung	549

22	Drupal optimieren	607
23	Installationsprofile	629
A	Referenz zu Datenbanktabellen	655
B	Quellen	693
	Stichwortverzeichnis	701

Inhaltsverzeichnis

Über den Autor	25
Über die Fachlektoren	27
Einleitung	29
Danksagungen	31
1 So funktioniert Drupal	33
1.1 Was ist Drupal?	33
1.2 Die Systemarchitektur	33
1.3 Der Core	35
1.4 Die Verwaltungsoberfläche	35
1.5 Module	36
1.6 Hooks	36
1.7 Themes	38
1.8 Nodes	38
1.9 Blöcke	39
1.10 Dateilayout	39
1.11 Ein Drupal-Seitenaufruf	41
1.11.1 Die Rolle des Webservers	41
1.11.2 Der Bootstrap-Prozess	42
1.11.3 Eine Anforderung verarbeiten	44
1.11.4 Daten mit Themes versehen	44
1.12 Zusammenfassung	44
2 Ein Modul schreiben	45
2.1 Die Dateien erstellen	45
2.2 Einen Hook implementieren	48
2.3 Modulspezifische Einstellungen hinzufügen	49
2.4 Das Formular zur Dateneingabe hinzufügen	53
2.4.1 Daten in einer Datenbanktabelle speichern	57
2.5 Den eigenen Verwaltungsabschnitt definieren	61
2.6 Dem Benutzer ein Formular mit Einstellungen anzeigen	64
2.7 Vom Benutzer übermittelte Einstellungen prüfen	66

2.8	Einstellungen speichern	67
2.8.1	Verwenden der Drupal-Tabelle variables	68
2.8.2	Abrufen von gespeicherten Werten mit variable_get()	69
2.9	Weitere Schritte	69
2.10	Zusammenfassung	69
3	Hooks, Aktionen und Trigger	71
3.1	Grundlagen von Ereignissen und Triggern	71
3.2	Grundlagen von Aktionen	73
3.2.1	Die Benutzeroberfläche für Trigger	74
3.2.2	Eine erste Aktion	76
3.2.3	Zuweisen der Aktion	77
3.2.4	Ändern der von einer Aktion unterstützten Trigger	78
3.2.5	Aktionen, die alle Trigger unterstützen	79
3.2.6	Erweiterte Aktionen	79
3.3	Den Kontext für Aktionen verwenden	84
3.3.1	Wie das Trigger-Modul den Kontext vorbereitet	84
3.3.2	Einrichten des Kontextes	86
3.4	Untersuchen des Kontextes	88
3.5	Wie Aktionen gespeichert werden	89
3.5.1	Die Tabelle actions	90
3.5.2	Aktions-IDs	90
3.6	Eine Aktion direkt mit actions_do() aufrufen	91
3.7	Eigene Trigger mit hook_hook_info() definieren	92
3.8	Trigger zu bestehenden Hooks hinzufügen	94
3.9	Zusammenfassung	96
4	Das Menüsystem	97
4.1	Callback-Zuordnung	97
4.1.1	URLs zu Funktionen zuordnen	98
4.1.2	Seitencallback-Argumente	103
4.2	Menüs verschachteln	107
4.3	Zugriffssteuerung	108
4.4	Titellokalisierung und Anpassung	109
4.4.1	Einen Titelcallback definieren	110
4.4.2	Titelargumente	112

4.5	Platzhalter in Menüelementen	113
4.5.1	Einfache Platzhalter	113
4.5.2	Platzhalter und Seitencallback-Parameter	114
4.5.3	Den Wert eines Platzhalters verwenden	114
4.5.4	Platzhalter- und Parameterersetzung	115
4.5.5	Mit to_arg()-Funktionen Pfade aus Platzhaltern erstellen	118
4.6	Menüelemente von anderen Modulen ändern	119
4.7	Menülinks von anderen Modulen ändern	121
4.8	Arten von Menüelementen	121
4.9	Häufige Aufgaben	123
4.9.1	Callbacks ohne Hinzufügen eines Links zuweisen	123
4.9.2	Menüelemente als Registerkarten anzeigen	123
4.9.3	Menüelemente verbergen	126
4.10	menu.module verwenden	126
4.11	Häufige Fehler	127
4.12	Zusammenfassung	127
5	Mit Datenbanken arbeiten	129
5.1	Datenbankparameter definieren	129
5.2	Grundlagen der Datenbankabstraktionsschicht	129
5.3	Verbindungen zur Datenbank	131
5.4	Einfache Abfragen durchführen	132
5.5	Abfrageergebnisse abrufen	134
5.5.1	Einen einzelnen Wert abrufen	134
5.5.2	Mehrere Zeilen abrufen	134
5.5.3	Das Ergebnis einschränken	135
5.5.4	Ergebnisse seitenweise anzeigen	135
5.6	Die Schema-API	136
5.6.1	.install-Moduldateien verwenden	136
5.6.2	Tabellen erstellen	137
5.6.3	Das Schema-Modul verwenden	139
5.6.4	Feldtypzuordnung zwischen Schema und Datenbank	141
5.6.5	Deklarieren eines Spaltentyps mit mysql_type	144
5.6.6	Tabellen pflegen	145
5.6.7	Tabellen beim Deinstallieren löschen	146
5.6.8	Vorhandene Schemas mit hook_schema_alter() löschen	147
5.7	Einfüge- und Aktualisierungsvorgänge mit drupal_write_record()	148

5.8	Abfragen mit <code>hook_db_rewrite_sql()</code> für andere Module ermöglichen	150
5.8.1	<code>hook_db_rewrite_sql()</code> verwenden	151
5.8.2	Abfragen anderer Module ändern	152
5.9	Verbindungen mit mehreren Datenbanken in Drupal	153
5.10	Eine temporäre Tabelle verwenden	154
5.11	Einen eigenen Datenbanktreiber schreiben	155
5.12	Zusammenfassung	157
6	Mit Benutzern arbeiten	159
6.1	Das Objekt <code>\$user</code>	159
6.1.1	Daten in <code>\$user</code> speichern	162
6.1.2	Ist der Benutzer angemeldet?	162
6.2	Einführung in <code>hook_user()</code>	163
6.2.1	Grundlagen von <code>hook_user('view')</code>	164
6.3	Die Benutzerregistrierung	166
6.3.1	Benutzerinformationen mit <code>profile.module</code> erfassen	168
6.4	Der Anmeldevorgang	169
6.4.1	Daten zur Ladezeit zu <code>\$user</code> hinzufügen	171
6.4.2	Kategorien für Benutzerinformationen bereitstellen	174
6.5	Externe Anmeldung	175
6.5.1	Einfache externe Authentifizierung	175
6.6	Zusammenfassung	181
7	Mit Nodes arbeiten	183
7.1	Was genau ist ein Node?	183
7.2	Nicht alles ist ein Node	186
7.3	Ein Node-Modul erstellen	187
7.3.1	Die <code>.install</code> -Datei erstellen	187
7.3.2	Die <code>.info</code> -Datei erstellen	188
7.3.3	Die <code>.module</code> -Datei erstellen	189
7.3.4	Informationen über unseren Node-Typ bereitstellen	189
7.3.5	Den Menücallback ändern	191
7.3.6	Berechtigungen für den Node-Typ mit <code>hook_perm()</code> definieren	192
7.3.7	Zugriff auf einen Node-Typ mit <code>hook_access()</code> einschränken	192
7.3.8	Das Node-Formular für den Node-Typ anpassen	193
7.3.9	Unterstützung für Filterformate hinzufügen	195
7.3.10	Felder mit <code>hook_validate()</code> validieren	196

7.3.11	Daten mit hook_insert() speichern	197
7.3.12	Daten mit hook_update() auf dem neuesten Stand halten	197
7.3.13	Aufräumen mit hook_delete()	198
7.3.14	Nodes eines Typs mit hook_load() ändern	198
7.3.15	Und nun die Pointe: hook_view()	199
7.3.16	Nodes eines fremden Typs mit hook_nodeapi() bearbeiten	203
7.4	Wie Nodes gespeichert werden	206
7.5	Node-Typen mit CCK erstellen	207
7.6	Den Zugriff auf Nodes einschränken	208
7.6.1	Node-Rechte definieren	208
7.6.2	Zugriff auf Nodes	210
7.7	Zusammenfassung	211
8	Das Theme-System	213
8.1	Komponenten des Theme-Systems	213
8.1.1	Template-Sprachen und Theme-Engines	213
8.1.2	Themes	215
8.2	Ein Theme installieren	217
8.3	Ein PHP-Template-Theme erstellen	217
8.3.1	Vorhandene HTML- und CSS-Dateien verwenden	218
8.3.2	Eine .info-Datei für das Theme erstellen	220
8.4	Grundlagen von Template-Dateien	225
8.4.1	Überblick	225
8.4.2	Für Themes geeignete Elemente überschreiben	229
8.4.3	Überschreiben mit Template-Dateien	231
8.4.4	Template-Variablen hinzufügen und bearbeiten	232
8.4.5	Variablen für alle Templates	235
8.4.6	page.tpl.php	235
8.4.7	node.tpl.php	239
8.4.8	block.tpl.php	242
8.4.9	comment.tpl.php	243
8.4.10	box.tpl.php	244
8.4.11	Weitere tpl.php-Dateien	245
8.4.12	Templates mit mehreren Seiten	245
8.5	Erweiterte Theme-Gestaltung in Drupal	246
8.5.1	Die Theme-Registry	246
8.5.2	Eine ausführliche Beschreibung von theme()	248

8.5.3	Neue Blockregionen definieren	252
8.5.4	Drupal-Formulare mit Themes versehen	252
8.5.5	Das Theme-Developer-Modul verwenden	252
8.6	Zusammenfassung	253
9	Mit Blöcken arbeiten	255
9.1	Was ist ein Block?	255
9.2	Optionen zur Blockkonfiguration	257
9.3	Platzierung von Blöcken	258
9.4	Einen Block definieren	258
9.4.1	Blöcke mit Themes versehen	260
9.4.2	Einen Block-Hook verwenden	261
9.5	Einen Block aufbauen	263
9.5.1	Bonusbeispiel: Einen Block für noch nicht aktivierte Benutzerkonten hinzufügen	270
9.6	Einen Block während der Installation eines Moduls aktivieren	271
9.7	Beispiele für die Sichtbarkeit von Blöcken	271
9.7.1	Einen Block nur für angemeldete Benutzer anzeigen	272
9.7.2	Einen Block nur für anonyme Benutzer anzeigen	272
9.8	Zusammenfassung	272
10	Die Formular-API	273
10.1	Grundlagen der Formularverarbeitung	274
10.1.1	Den Vorgang initialisieren	275
10.1.2	Ein Token festlegen	275
10.1.3	Eine ID festlegen	275
10.1.4	Definitionen aller möglichen Formularelemente erfassen	275
10.1.5	Eine Validierungsfunktion suchen	277
10.1.6	Eine Übermittlungsfunktion suchen	277
10.1.7	Das Formular vor dem Aufbau durch Module ändern lassen	278
10.1.8	Das Formular aufbauen	278
10.1.9	Das Formular nach dem Aufbau durch Funktionen ändern lassen	278
10.1.10	Die Übermittlung des Formulars prüfen	278
10.1.11	Eine Theme-Funktion für das Formular finden	279
10.1.12	Das Formular vor dem Rendern durch Module ändern lassen	279
10.1.13	Das Formular rendern	279
10.1.14	Das Formular validieren	280

10.1.15	Das Formular übertragen	281
10.1.16	Den Benutzer umleiten	281
10.2	Einfache Formulare erstellen	282
10.2.1	Formulareigenschaften	284
10.2.2	Formular-IDs	285
10.2.3	Feldgruppen	286
10.2.4	Formulare mit Themes versehen	288
10.2.5	Validierungs- und Übermittlungsfunktionen mit hook_forms() festlegen	291
10.2.6	Die Aufrufreihenfolge von Theme-, Validierungs- und Übermittlungsfunktionen	292
10.2.7	Eine Validierungsfunktion schreiben	293
10.2.8	Das Formular neu aufbauen	297
10.2.9	Eine Übermittlungsfunktion schreiben	298
10.2.10	Formulare mit hook_form_alter() ändern	299
10.2.11	Formulare programmgesteuert mit drupal_execute() übertragen	300
10.2.12	Mehrseitige Formulare	300
10.3	Eigenschaften der Formular-API	305
10.3.1	Eigenschaften für den Formularstamm	305
10.3.2	Zu allen Elementen hinzugefügte Eigenschaften	308
10.3.3	In allen Elementen zugelassene Eigenschaften	309
10.3.4	Formularelemente	310
10.3.5	Die Eigenschaft #ahah	321
10.4	Zusammenfassung	327
11	Benutzereingaben bearbeiten: Das Filtersystem	329
11.1	Filter	329
11.2	Filter und Eingabeformate	330
11.2.1	Einen Filter installieren	333
11.2.2	Wann sind Filter notwendig?	334
11.3	Einen benutzerdefinierten Filter erstellen	336
11.3.1	hook_filter() implementieren	337
11.3.2	Die Operation list	338
11.3.3	Die Operation description	338
11.3.4	Die Operation settings	339
11.3.5	Die Operation no cache	339
11.3.6	Die Operation prepare	340
11.3.7	Die Operation process	340
11.3.8	Die Operation default	340
11.3.9	hook_filter_tips()	341

11.4	Schutz gegen potenziell gefährliche Daten	343
11.5	Zusammenfassung	343
12	Inhalte durchsuchen und indizieren	345
12.1	Eine benutzerdefinierte Suchseite erstellen	345
12.1.1	Das standardmäßige Suchformular	346
12.1.2	Das erweiterte Suchformular	347
12.1.3	Das Suchformular erweitern	347
12.2	Den HTML-Indexer für die Suche verwenden	353
12.2.1	Wann ist der Indexer geeignet?	354
12.2.2	Die Funktionsweise des Indexers	354
12.3	Zusammenfassung	364
13	Mit Dateien arbeiten	365
13.1	Wie Drupal Dateien bereitstellt	365
13.1.1	Öffentliche Dateien	366
13.1.2	Private Dateien	367
13.2	PHP-Einstellungen	367
13.3	Umgang mit Medien	368
13.3.1	Das Upload-Modul	368
13.3.2	Andere generische Module zur Dateiverarbeitung	369
13.3.3	Bilder und Bildergalerien	369
13.3.4	Video und Audio	370
13.4	Die Datei-API	370
13.4.1	Das Datenbankschema	370
13.4.2	Häufige Aufgaben und Funktionen	371
13.4.3	Authentifizierungshooks für den Download	382
13.5	Zusammenfassung	384
14	Mit Taxonomien arbeiten	385
14.1	Was ist eine Taxonomie?	385
14.1.1	Begriffe	385
14.1.2	Vokabulare	386
14.2	Arten von Taxonomien	388
14.2.1	Flach	389
14.2.2	Hierarchisch	390
14.2.3	Mehrfach hierarchisch	391

14.3	Inhalte nach Begriffen anzeigen	392
14.3.1	AND und OR in URLs verwenden	392
14.3.2	Die Tiefe für hierarchische Vokabulare angeben	393
14.3.3	Automatische RSS-Feeds	394
14.4	Taxonomien speichern	395
14.5	Modulgestützte Vokabulare	396
14.5.1	Ein modulgestütztes Vokabular erstellen	396
14.5.2	Benutzerdefinierte Pfade für Begriffe bereitstellen	397
14.5.3	Mit hook_taxonomy() über Vokabularänderungen informiert bleiben	398
14.6	Häufige Aufgaben	399
14.6.1	Taxonomiebegriffe in einem Knotenobjekt finden	400
14.6.2	Eigene Taxonomieabfragen erstellen	400
14.7	Taxonomiefunktionen	401
14.7.1	Informationen über Vokabulare abrufen	401
14.7.2	Vokabulare hinzufügen, ändern und löschen	401
14.7.3	Informationen über Begriffe abrufen	402
14.7.4	Begriffe hinzufügen, ändern und löschen	403
14.7.5	Informationen über die Begriffshierarchie abrufen	404
14.7.6	Informationen über Begriffssynonyme abrufen	406
14.7.7	Nodes mit bestimmten Begriffen finden	407
14.8	Weitere Quellen	408
14.9	Zusammenfassung	408
15	Caching	409
15.1	Wann muss zwischengespeichert werden?	409
15.2	Wie Caching funktioniert	410
15.3	Caching im Drupal-Core	412
15.3.1	Das Menüsystem	412
15.3.2	Gefilterte Eingabeformate	412
15.3.3	Verwaltungsvariablen und Moduleinstellungen	413
15.3.4	Seiten	413
15.3.5	Blöcke	420
15.3.6	Abfrageweises Caching mit statischen Variablen	422
15.3.7	Die Cache-API verwenden	422
15.4	Zusammenfassung	426

16 Sitzungen (Sessions)	427
16.1 Was sind Sitzungen?	427
16.2 Verwendung	428
16.3 Einstellungen für Sitzungen	430
16.3.1 In .htaccess	430
16.3.2 In settings.php	430
16.3.3 In bootstrap.inc	431
16.3.4 Die Verwendung von Cookies erzwingen	432
16.4 Speicherung	432
16.5 Lebenszyklus von Sitzungen	433
16.6 Konversationen in Sitzungen	435
16.6.1 Erster Besuch	436
16.6.2 Zweiter Besuch	436
16.6.3 Benutzer mit einem Konto	436
16.7 Häufige Aufgaben	436
16.7.1 Die Gültigkeitsdauer eines Cookies verlängern	436
16.7.2 Den Namen der Sitzung ändern	437
16.7.3 Daten in der Sitzung speichern	437
16.8 Zusammenfassung	438
17 jQuery	439
17.1 Was ist jQuery?	439
17.2 Die herkömmliche Vorgehensweise	440
17.3 So funktioniert jQuery	441
17.3.1 Einen CSS-ID-Selektor verwenden	441
17.3.2 Einen CSS-Klassenselektor verwenden	442
17.4 jQuery in Drupal	443
17.4.1 Ein erstes Codebeispiel mit jQuery	444
17.4.2 Ein Element über die ID ansprechen	447
17.4.3 Methodenverkettung	448
17.4.4 Klassen hinzufügen und entfernen	448
17.4.5 Vorhandene Elemente mit einem Wrapper versehen	448
17.4.6 Die Werte von CSS-Elementen ändern	449
17.4.7 Wohin mit dem JavaScript-Code?	450
17.4.8 Überschreibbarer JavaScript-Code	453

17.5	Ein Abstimmungs-Widget mit jQuery erstellen	456
17.5.1	Das Modul erstellen	459
17.5.2	Drupal.behaviors verwenden	468
17.5.3	Erweiterungsmöglichkeiten für dieses Modul	468
17.5.4	Kompatibilität	469
17.6	Die nächsten Schritte	469
17.7	Zusammenfassung	469
18	Lokalisierung und Übersetzung	471
18.1	Das Locale-Modul aktivieren	471
18.2	Übersetzung der Benutzeroberfläche	471
18.2.1	Strings	471
18.2.2	Strings mit t() übersetzen	472
18.2.3	Eingebaute Strings durch benutzerdefinierte ersetzen	474
18.3	Eine neue Übersetzung beginnen	485
18.3.1	.pot-Dateien für Drupal herunterladen	486
18.3.2	.pot-Dateien mit dem Translation Template Extractor erstellen	487
18.4	Eine Übersetzung installieren	490
18.4.1	Eine Übersetzung zur Installationszeit einrichten	490
18.4.2	Eine Übersetzung auf einer bestehenden Site installieren	491
18.5	Unterstützung für von rechts nach links geschriebene Sprachen	492
18.6	Sprachbehandlung	493
18.6.1	Keine	494
18.6.2	Nur Pfadpräfix	495
18.6.3	Pfadpräfix mit Ausweichsprache	497
18.6.4	Nur Domain-Name	497
18.7	Übersetzung des Inhalts	498
18.7.1	Einführung in das Modul zur Inhaltsübersetzung	498
18.7.2	Unterstützung für mehrere Sprachen	498
18.7.3	Unterstützung für mehrere Sprachen mit Übersetzung	500
18.8	Dateien für Lokalisierung und Übersetzung	503
18.9	Weitere Quellen	504
18.10	Zusammenfassung	504
19	XML-RPC	507
19.1	Was ist XML-RPC?	507
19.2	Voraussetzungen für XML-RPC	507

19.3	XML-RPC-Clients	508
19.3.1	Ein Beispiel für einen XML-RPC-Client: Die Uhrzeit abrufen	508
19.3.2	Ein Beispiel für einen XML-RPC-Client: Den Namen eines Bundesstaats abrufen	510
19.3.3	Vorgehensweise bei XML-RPC-Clientfehlern	511
19.3.4	Parametertypen umwandeln	513
19.4	Ein einfacher XML-RPC-Server	514
19.4.1	Die Methode mit <code>hook_xmlrpc()</code> zuordnen	515
19.4.2	Automatische Validierung von Parametertypen mit <code>hook_xmlrpc()</code>	516
19.5	Eingebaute XML-RPC-Methoden	518
19.5.1	<code>system.listMethods</code>	518
19.5.2	<code>system.methodSignature</code>	519
19.5.3	<code>system.methodHelp</code>	519
19.5.4	<code>system.getCapabilities</code>	519
19.5.5	<code>system.multiCall</code>	520
19.6	Zusammenfassung	520
20	Sicheren Code schreiben	521
20.1	Benutzereingaben handhaben	521
20.1.1	Überlegungen zu Datentypen	522
20.1.2	Gesunde Ausgaben mit <code>check_plain()</code> und <code>t()</code>	524
20.1.3	Mit <code>filter_xss()</code> Angriffe durch siteübergreifendes Scripting verhindern	527
20.1.4	<code>filter_xss_admin()</code> verwenden	529
20.2	URLs sicher handhaben	529
20.3	Abfragen mit <code>db_query()</code> sicher gestalten	530
20.4	Private Daten mit <code>db_rewrite_sql()</code> schützen	535
20.5	Dynamische Abfragen	536
20.6	Berechtigungen und Seitencallbacks	537
20.7	Cross-Site Request Forgery (CSRF)	538
20.8	Dateisicherheit	539
20.8.1	Dateiberechtigungen	539
20.8.2	Geschützte Dateien	539
20.8.3	Dateiuploads	540
20.8.4	Dateinamen und Pfade	541
20.9	Mailheader verschlüsseln	542
20.9.1	Dateien für Produktionsumgebungen	542

20.10	cron.php schützen	543
20.11	SSL-Unterstützung	544
20.12	Eigenständiges PHP	544
20.13	AJAX-Sicherheit	545
20.14	Sicherheit in der Formular-API	546
20.15	Das Superuser-Konto schützen	547
20.16	eval() verwenden	547
20.17	Zusammenfassung	548
21	Bewährte Vorgehensweisen in der Entwicklung	549
21.1	Standards für die Programmierung	549
21.1.1	Zeileneinzug	549
21.1.2	Öffnende und schließende Tags in PHP	549
21.1.3	Steuerstrukturen	550
21.1.4	Funktionsaufrufe	551
21.1.5	Funktionsdeklarationen	552
21.1.6	Funktionsnamen	552
21.1.7	Arrays	553
21.1.8	Konstanten	554
21.1.9	Globale Variablen	554
21.1.10	Modulnamen	554
21.1.11	Dateinamen	555
21.2	PHP-Kommentare	556
21.2.1	Beispiele zur Dokumentierung	557
21.2.2	Konstanten dokumentieren	557
21.2.3	Funktionen dokumentieren	558
21.2.4	Hookimplementierungen dokumentieren	560
21.3	Den Programmierstil per Programm überprüfen	560
21.3.1	code-style.pl verwenden	560
21.3.2	Das Coder-Modul verwenden	561
21.4	Orientierung im Code mit egrep	562
21.5	Die Vorteile der Versionssteuerung nutzen	563
21.5.1	CVS-fähiges Drupal installieren	564
21.5.2	CVS-fähiges Drupal verwenden	565
21.5.3	Einen CVS-Client installieren	565
21.5.4	Drupal in CVS auschecken	565
21.5.5	Zweige und Tags	567

21.5.6	Code mit CVS aktualisieren	572
21.5.7	Änderungen im Drupal-Code nachverfolgen	574
21.5.8	CVS-Konflikte lösen	575
21.5.9	Kerncode sauber ändern	575
21.6	Patches erstellen und anwenden	576
21.6.1	Einen Patch anwenden	577
21.7	Ein Modul warten	578
21.7.1	Ein Drupal-CVS-Konto einrichten	578
21.7.2	Das Beitragsrepository auschecken	579
21.7.3	Eigene Module zum Repository hinzufügen	581
21.7.4	Der Anfangscommit	582
21.7.5	Das eigene Modul auschecken	583
21.7.6	Ein Projekt auf drupal.org erstellen	583
21.7.7	Commit eines Bugfix	584
21.7.8	Den Verlauf einer Datei anzeigen	586
21.7.9	Einen Zweig erstellen	586
21.7.10	Einen Drupal-6-kompatiblen Zweig erstellen	590
21.7.11	Erweiterte Verzweigung	594
21.7.12	Einen Release-Node erstellen	596
21.8	SVN mit CVS für die Projektverwaltung kombinieren	597
21.9	Code testen und entwickeln	598
21.9.1	Das Devel-Modul	598
21.9.2	Abfragen anzeigen	599
21.9.3	Zeitintensive Abfragen	600
21.9.4	Andere Anwendungen für das Devel-Modul	601
21.10	Das Modul-Builder-Modul	602
21.11	Profilerstellung für Anwendungen und Debugging	602
21.12	Zusammenfassung	605
22	Drupal optimieren	607
22.1	Engpässe finden	607
22.1.1	Eine erste Untersuchung	607
22.1.2	Andere Formen der Webserveroptimierung	611
22.1.3	Datenbankengpässe	612
22.2	Drupal-spezifische Optimierungen	617
22.2.1	Seitencaching	618
22.2.2	Bandbreitenoptimierung	618

22.2.3	Die Sitzungstabelle aufräumen	618
22.2.4	Den Datenverkehr authentifizierter Benutzer verwalten	619
22.2.5	Fehlerprotokolle straffen	619
22.2.6	cron ausführen	620
22.2.7	Automatische Drosselung	621
22.3	Architekturen	624
22.3.1	Einzelner Server	624
22.3.2	Eigener Datenbankserver	624
22.3.3	Eigenständiger Datenbankserver und Webservercluster	624
22.3.4	Mehrere Datenbankserver	626
22.4	Zusammenfassung	627
23	Installationsprofile	629
23.1	Der Speicherort der Profile	629
23.2	Funktionsweise von Installationsprofilen	630
23.2.1	Die zu aktivierenden Module angeben	631
23.2.2	Zusätzliche Installationsaufgaben definieren	633
23.2.3	Zusätzliche Installationsaufgaben ausführen	636
23.2.4	Quellen	652
23.3	Zusammenfassung	654
A	Referenz zu Datenbanktabellen	655
A.1	access (User-Modul)	655
A.2	accesslog (Statistics-Modul)	656
A.3	actions (Trigger-Modul)	656
A.4	actions_aid (Trigger-Modul)	657
A.5	aggregator_category (Aggregator-Modul)	657
A.6	aggregator_category_feed (Aggregator-Modul)	657
A.7	aggregator_category_item (Aggregator-Modul)	658
A.8	aggregator_feed (Aggregator-Modul)	658
A.9	aggregator_item (aggregator-Modul)	659
A.10	authmap (User-Modul)	659
A.11	batch (batch.inc)	660
A.12	blocks (Block-Modul)	660
A.13	blocks_roles (Block-Modul)	662
A.14	book (Book-Modul)	662

A.15	boxes (Block-Modul)	662
A.16	cache	663
A.17	cache_block (Block-Modul)	663
A.18	cache_filter (Filter-Modul)	664
A.19	cache_form	664
A.20	cache_menu	665
A.21	cache_page	665
A.22	cache_update	666
A.23	comments (Comment-Modul)	666
A.24	contact (Contact-Modul)	667
A.25	files (Upload-Modul)	668
A.26	filter_formats (Filter-Modul)	668
A.27	filters (Filter-Modul)	669
A.28	flood (Contact-Modul)	669
A.29	forum (Forum-Modul)	669
A.30	history (Node-Modul)	670
A.31	languages (Locale-Modul)	670
A.32	locales_source (Locale-Modul)	671
A.33	locales_target (Locale-Modul)	671
A.34	menu_custom (Menu-Modul)	672
A.35	menu_links (Menu-Modul)	672
A.36	menu_router	674
A.37	node (Node-Modul)	676
A.38	node_access (Node-Modul)	677
A.39	node_comment_statistics (Comment-Modul)	677
A.40	node_counter (Statistics-Modul)	678
A.41	node_revisions (Node-Modul)	678
A.42	node_type (Node-Modul)	679
A.43	openid_association (Openid-Modul)	680
A.44	permission (User-Modul)	680
A.45	poll (Poll-Modul)	681
A.46	poll_choices (Poll-Modul)	681
A.47	poll_votes (Poll-Modul)	681

A.48	profile_fields (Profile-Modul)	682
A.49	profile_values (Profile-Modul)	683
A.50	role (User-Modul)	683
A.51	search_dataset (Search-Modul)	683
A.52	search_index (Search-Modul)	684
A.53	search_node_links (Search-Modul)	684
A.54	search_total (Search-Modul)	684
A.55	sessions	685
A.56	system	685
A.57	term_data (Taxonomy-Modul)	686
A.58	term_hierarchy (Taxonomy-Modul)	687
A.59	term_node (Taxonomy-Modul)	687
A.60	term_relation (Taxonomy-Modul)	687
A.61	term_synonym (Taxonomy-Modul)	688
A.62	trigger_assignments (Trigger-Modul)	688
A.63	upload (Upload-Modul)	688
A.64	url_alias (Path-Modul)	689
A.65	users (User-Modul)	689
A.66	users_roles (users)	690
A.67	variable	691
A.68	vocabulary (Taxonomy-Modul)	691
A.69	vocabulary_node_types (Taxonomy-Modul)	692
A.70	watchdog (Dblog-Modul)	692
B	Quellen	693
B.1	Code	693
B.1.1	Drupal-CVS	693
B.1.2	Drupal-API-Referenz	693
B.1.3	Sicherheitsratschläge	693
B.1.4	Module aktualisieren	694
B.1.5	Themes aktualisieren	694
B.2	Handbücher	694
B.3	Foren	694

B.4	Mailinglisten	695
B.4.1	development	695
B.4.2	documentation	695
B.4.3	drupal-cvs	695
B.4.4	infrastructure	695
B.4.5	support	695
B.4.6	themes	695
B.4.7	translations	696
B.4.8	webmasters	696
B.4.9	CVS-applications	696
B.4.10	consulting	696
B.5	Benutzer- und Interessengruppen	696
B.6	Internet Relay Chat	696
B.6.1	#drupal-support	697
B.6.2	#drupal-themes	697
B.6.3	#drupal-ecommerce	697
B.6.4	#drupal	697
B.6.5	#drupal-dev	697
B.6.6	#drupal-consultants	698
B.6.7	#drupal-dojo	698
B.7	Videocasts	698
B.8	Weblogs	698
B.8.1	Planet Drupal	698
B.9	Konferenzen	698
B.10	Eigene Beiträge	699
	Stichwortverzeichnis	701