

ACADEMIC SERVICE

BASIS CURSUS

PHP 5.4

*Ook geschikt voor PHP 5.3
en developers-versie 6.0*

VICTOR PETERS

Basiscursussen verschenen bij Academic Service:

Basiscursus Access 2007
Basiscursus Access 2003
Basiscursus Access 2002
Basiscursus ASP.NET
Basiscursus AutoCAD 2010 en LT 2010
Basiscursus AutoCAD 2009 en LT 2009
Basiscursus AutoCAD 2008 en LT 2008
Basiscursus AutoCAD 2007 en LT 2007
Basiscursus AutoCAD 2005 en LT 2005
Basiscursus AutoCAD 2004
Basiscursus AutoCAD LT 2004
Basiscursus C++ 3e herziene druk
Basiscursus Cascading Style Sheets
Basiscursus Contribute
Basiscursus Dreamweaver CS4
Basiscursus Dreamweaver CS3
Basiscursus Dreamweaver 8
Basiscursus Dreamweaver MX 2004
Basiscursus Dreamweaver MX
Basiscursus Excel 2007
Basiscursus Excel 2003
Basiscursus Excel 2002
Basiscursus Flash CS4
Basiscursus Flash CS3
Basiscursus Flash 8
Basiscursus Flash MX 2004
Basiscursus Flash MX
Basiscursus Flash ActionScript
Basiscursus FrontPage 2003
Basiscursus FrontPage 2002
Basiscursus HTML 4.01
Basiscursus Illustrator CS4
Basiscursus Illustrator CS3
Basiscursus Illustrator CS2
Basiscursus Illustrator 10/CS
Basiscursus InDesign CS4
Basiscursus InDesign CS3
Basiscursus InDesign CS2
Basiscursus InDesign CS
Basiscursus Internet, 3e herziene druk
Basiscursus Internet Explorer 6
Basiscursus Java, 2e herziene druk
Basiscursus JavaScript 1.5
Basiscursus Joomla! 1.5
Basiscursus Mac OSX 10.5 Leopard
Basiscursus Mac OSX 10.3 Panther
Basiscursus Mac OSX 10.2 Jaguar
Basiscursus Outlook 2007

Basiscursus Outlook 2003
Basiscursus Outlook 2002
Basiscursus Paint Shop Pro X
Basiscursus Paint Shop Pro 9
Basiscursus Photoshop Elements 3.0
Basiscursus Photoshop CS4
Basiscursus Photoshop CS3
Basiscursus Photoshop CS2
Basiscursus Photoshop CS
Basiscursus Photoshop 7
Basiscursus PHP 5.4
Basiscursus PHP 6
Basiscursus PHP 5
Basiscursus PHP 4.2
Basiscursus Pinnacle Liquid Edition
Basiscursus Pinnacle Studio 10
Basiscursus Pinnacle Studio 9
Basiscursus PowerPoint 2007
Basiscursus PowerPoint 2003
Basiscursus PowerPoint 2002
Basiscursus Premiere Elements
Basiscursus Premiere Pro
Basiscursus Premiere 6.5
Basiscursus Project 2003
Basiscursus QuarkXPress 5
Basiscursus SQL, 2e herziene druk
Basiscursus SUSE Linux 10
Basiscursus Visio 2002
Basiscursus Visual Basic 2005 EE
Basiscursus Visual Basic.NET
Basiscursus Visual Basic 6.0
Basiscursus Windows 7
Basiscursus Windows Vista
Basiscursus Windows XP
Basiscursus Word 2007
Basiscursus Word 2003
Basiscursus Word 2002
Basiscursus XHTML 1.0
Basiscursus XML herziene editie

Voor meer informatie en bestellingen:

Sdu Klantenservice
Postbus 20014
2500 EA Den Haag
Tel.: 070-3789880
Website: <http://www.academicservice.nl>

Victor Peters

Basiscursus

PHP 5.4

**Ook geschikt voor PHP 5.3
en developers-versie 6.0**



Meer informatie over deze en andere uitgaven kunt u verkrijgen bij:
Sdu Klantenservice
Postbus 20014
2500 EA Den Haag
tel.: (070) 378 98 80
www.sdu.nl/service

© 2012 Sdu Uitgevers bv, Den Haag
Academic Service is een imprint van Sdu Uitgevers bv

Redactie en zetwerk: Redactie bureau Ron Heijer, Markelo
Ontwerp omslag: Sjef Nix, Amsterdam

ISBN: 978 90 12 58499 9
NUR: 980

Alle rechten voorbehouden. Alle auteursrechten en databankrechten ten aanzien van deze uitgave worden uitdrukkelijk voorbehouden. Deze rechten berusten bij Sdu Uitgevers bv.

Behoudens de in of krachtens de Auteurswet gestelde uitzonderingen, mag niets uit deze uitgave worden verveelvoudigd, opgeslagen in een geautomatiseerd gegevensbestand of openbaar gemaakt in enige vorm of op enige wijze, hetzij elektronisch, mechanisch, door fotokopieën, opnamen of enige andere manier, zonder voorafgaande schriftelijke toestemming van de uitgever.

Voor zover het maken van reprografische verveelvoudigingen uit deze uitgave is toegestaan op grond van artikel 16 h Auteurswet, dient men de daarvoor wettelijk verschuldigde vergoedingen te voldoen aan de Stichting Reprorecht (Postbus 3051, 2130 KB Hoofddorp, www.reprorecht.nl). Voor het overnemen van gedeelte(n) uit deze uitgave in bloemlezingen, readers en andere compilatiewerken (artikel 16 Auteurswet) dient men zich te wenden tot de Stichting PRO (Stichting Publicatie- en Reproductierechten Organisatie, Postbus 3060, 2130 KB Hoofddorp, www.cedar.nl/pro). Voor het overnemen van een gedeelte van deze uitgave ten behoeve van commerciële doeleinden dient men zich te wenden tot de uitgever.

Hoewel aan de totstandkoming van deze uitgave de uiterste zorg is besteed, kan voor de afwezigheid van eventuele (druk)fouten en onvolledigheden niet worden ingestaan en aanvaarden de auteur(s), redacteur(en) en uitgever deswege geen aansprakelijkheid voor de gevolgen van eventueel voorkomende fouten en onvolledigheden.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the publisher's prior consent.

While every effort has been made to ensure the reliability of the information presented in this publication, Sdu Uitgevers neither guarantees the accuracy of the data contained herein nor accepts responsibility for errors or omissions or their consequences.

Inhoud

	Inleiding	1
Hoofdstuk 1	Aan de slag met PHP	3
	1.1 In dit hoofdstuk	3
	1.2 Webserver en browser	3
	1.3 HTML, Ajax, XML, CSS en JavaScript	4
	1.4 WordPress, Joomla en PHPBB	4
	1.5 PHP is een rommelpot	4
	1.6 Model, View, Controller	5
	1.7 OOP	5
	1.8 SQL en MySQL	5
	1.9 WAMP, LAMP, MAMP op uw eigen computer	6
	1.10 Editors	6
	1.11 'You will be assimilated'	7
	1.11.1 Syntaxis	8
	1.11.2 Herhaling	9
	1.11.3 Vele wegen naar Rome	10
	1.11.4 Functies	11
	1.11.5 Complexe factoren	12
	1.11.6 Denken als een computer	13
	1.12 Top-down refinement, stapsgewijze verfijning	14
	1.13 Tot slot	14
Hoofdstuk 2	De WAMP-server installeren	15
	2.1 In dit hoofdstuk	15
	2.2 De juiste PHP-versie	15
	2.3 Wampserver downloaden	16
	2.4 Wampserver met PHP 6 als snapshot downloaden	17
	2.5 Wampserver installeren	20
	2.5.1 Wampserver actief of inactief	21
	2.5.2 Online of offline	23
	2.6 De PHP 6-snapshot installeren	24
	2.7 Uw eigen server	24
	2.8 De PHP-versie instellen	25
	2.9 Een gebruiker voor MySQL instellen	27
	2.10 Bestandsextensies zichtbaar maken in Windows	30
	2.11 Een eerste project	31
	2.12 Interpreter of compiler	36
	2.13 Tot slot	36

Hoofdstuk 3	Een handige PHP-editor	37
3.1	In dit hoofdstuk	37
3.2	Twee soorten PHP-editors	37
3.2.1	Ontwikkelomgeving	38
3.3	Scite	39
3.3.1	Scite downloaden	39
3.3.2	Scite installeren	41
3.4	De editor koppelen aan Windows	42
3.5	Tot slot	46
Hoofdstuk 4	De andere webtalen	47
4.1	In dit hoofdstuk	47
4.2	HTML 4 of 5?	47
4.3	De HTML-basispagina	47
4.3.1	Doctype en meta-tags	49
4.4	Het CSS-stijlblad	50
4.5	Een JavaScript-pagina	53
4.6	Het resultaat	54
4.7	Tot slot	55
Hoofdstuk 5	PHP in HTML of andersom	57
5.1	In dit hoofdstuk	57
5.2	Vorbereidingen	57
5.3	Echo en Print	60
5.4	Dubbele of enkele aanhalingstekens	61
5.5	PHP invoegen met <code><?php ?></code>	63
5.6	Tekst in een Heredoc of Nowdoc	64
5.7	Variabelen binnen een string	66
5.8	De functies <code>include()</code> en <code>include_once()</code>	67
5.9	Regeleinden in PHP	70
5.10	Tot slot	71
Hoofdstuk 6	Variabelen en datatypen	73
6.1	In dit hoofdstuk	73
6.2	Datatypen	73
6.3	Enkelvoudige datatypen	74
6.4	Meervoudige datatypen	74
6.5	Strings	75
6.5.1	Het escape-teken <code>\</code>	76
6.6	Integers en floats	80
6.6.1	Strings en integers koppelen	81
6.6.2	Rekenen met integers	82
6.6.3	Functies voor getallen	83
6.7	Booleans: waar of onwaar	84
6.8	Vergelijkingen	85
6.8.1	Rekenen met booleans	86
6.9	Conversie van datatypen	87
6.10	Constanten	89
6.10.1	Magic constants	90

6.11	Arrays	90
6.11.1	Itereren	92
6.11.2	Associatieve arrays	93
6.11.3	Array-functies	94
6.11.4	Speciale arrays	96
6.12	Expressies, statements, operatoren en ander jargon	97
6.13	Tot slot	98
Hoofdstuk 7	Beslissingen en herhalingen	99
7.1	In dit hoofdstuk	99
7.2	If-then-else	99
7.2.1	Else en elseif	101
7.3	Kiezen met switch-case	102
7.4	Wat een omslachtige code...	103
7.5	While	103
7.6	Do-while	107
7.7	For	108
7.7.1	Foreach	108
7.8	Een lus onderbreken met break	109
7.9	Oneindige lus	110
7.10	=, ==, ===, != en !	111
7.11	Yahze spelen in de browser	112
7.11.1	Init	113
7.11.2	View	113
7.11.3	Controller	114
7.11.4	POST en GET	115
7.11.5	Uitbreiding in de view	117
7.11.6	PHP of JavaScript	119
7.12	Model, view, controller	119
7.13	Tot slot	119
Hoofdstuk 8	Functies	121
8.1	In dit hoofdstuk	121
8.2	Bereik of scope	121
8.3	Een functie declareren	122
8.4	Waarom functies?	124
8.5	Parameters	124
8.5.1	By Value of By Reference	126
8.5.2	Parameters declareren	127
8.5.3	Meerdere parameters	128
8.6	Een array als parameter	129
8.6.1	Global	131
8.6.2	\$GLOBALS	132
8.6.3	Retourwaarden	133
8.7	Recursie	135
8.7.1	De faculteitsmachine	137
8.7.2	Op elke plek	139
8.8	Tot slot	141

Hoofdstuk 9	Rekenen met tijd	143
	9.1 In dit hoofdstuk	143
	9.2 UNIX-tijd	143
	9.3 Time()	144
	9.4 Date()	145
	9.5 Getdate()	147
	9.6 Gettimeofday()	147
	9.7 Mktime()	148
	9.8 Tot slot	150
Hoofdstuk 10	Sessies en cookies	151
	10.1 In dit hoofdstuk	151
	10.2 Een sessie maken met cookies	151
	10.2.1 Verlopen en laten verlopen	155
	10.2.2 Cookies lezen met de global \$_COOKIE	156
	10.3 Een sessie maken met session	158
	10.3.1 PHP 6 of een oudere versie	161
	10.4 Een sessie onderhouden met \$_SESSION	164
	10.4.1 Drie formulieren	167
	10.4.2 De formulieren verwerken	171
	10.4.3 Vele wegen naar Rome	178
	10.5 Tot slot	182
Hoofdstuk 11	Objectgeoriënteerd programmeren	183
	11.1 In dit hoofdstuk	183
	11.2 Het principe achter OOP: klassen en objecten	183
	11.2.1 __construct()	186
	11.2.2 Een variabel aantal dieren	188
	11.3 Afgeleide klassen	193
	11.3.1 Een functieaanroep met call_user_func()	196
	11.3.2 Kleinkinderen	200
	11.3.3 Objectje pesten	202
	11.4 Tot slot	203
Hoofdstuk 12	De MySQL-database	205
	12.1 In dit hoofdstuk	205
	12.2 MySQL en SQL	205
	12.2.1 Database voor beginners	206
	12.3 phpMyAdmin	207
	12.3.1 Een gebruiker aanmaken	208
	12.3.2 Een database maken	210
	12.4 Een tabel ontwerpen	211
	12.4.1 Relatieeel	212
	12.5 Data klaarmaken voor MySQL	215
	12.6 Een PHP-project voorbereiden op MySQL	218
	12.7 SQL in PHP	220
	12.8 INSERT, UPDATE, DELETE en SELECT	221
	12.8.1 INSERT	221
	12.8.2 UPDATE	225
	12.8.3 DELETE	226
	12.8.4 SELECT	226

	12.9 Een db-klasse maken	229
	12.10 Gebruikersbeheer	232
	12.11 Tot slot	237
Hoofdstuk 13	Paarden voeren met klassen en MySQL	239
	13.1 In dit hoofdstuk	239
	13.2 Drie PHP-documenten	239
	13.3 Relationale database	241
	13.4 Weergeven, maken, wijzigen en wissen	242
	13.5 JavaScript en CSS	253
	13.6 Tot slot	255
Hoofdstuk 14	Snippers en bronnen	257
	14.1 In dit hoofdstuk	257
	14.2 Bronnen	257
	14.3 Links en rechts	260
	14.4 Variabele variabelen	260
	14.5 Variabele functies	261
	14.6 Wandelen en zoeken door een array	261
	14.7 Regular Expressions	262
	14.8 Functies en parameters	263
	14.9 If-then-else	265
	14.10 Mail sturen met PHP	265
	14.11 Browser-gegevens	266
	14.12 Phpinfo()	267
	14.13 Php.ini	267
	14.14 Tot slot	269
	Register	271

Inleiding

Weer een nieuwe versie van PHP en dus weer een nieuw boek over PHP. Is dat nodig? Veranderen er nog wezenlijk zaken aan PHP waardoor een programmeur opnieuw aan de studie moet? En als PHP nieuw is voor u, moet u dan kiezen voor de nieuwste versie van PHP – versie 5.4 of zelfs PHP 6 – in plaats van starten met oudere versies als PHP 5.3? Het antwoord op al deze vragen is: nee.

PHP 5.4 is ontstaan na een ontwikkeling van jaren, maar is nog steeds een soort tussenfase. PHP 5.4 bevat al wel het goede en nieuwe van PHP 5.3 op het gebied van Object Oriented Programming (OOP) en op het gebied van beveiliging.

PHP 5.3 was echter nog vrijwel volledig compatibel met lagere PHP-versies. Er zaten daardoor nogal wat archaïsche en ongewenste elementen in PHP 5.3 die een erfenis waren van PHP 4 en oudere versies.

Met de komst van PHP 5.4 is dit grotendeels voorbij. PHP 5.4 is zogezegd opgeruimd en van ongewenste elementen ontdaan. Aan nieuwigheden vindt u maar bar weinig in PHP 5.4, op enkele details na. Aanvankelijk was ervoor gekozen om deze opgeruimde variant van PHP 5 geen versienummer als 5.x maar een eigen versienummer te geven: 6. Oude PHP-scripts zullen namelijk niet zonder slag of stoot draaien op een server met PHP 6. PHP 6 is dus – ondanks de relatief kleine veranderingen – een volstrekt nieuwe versie, juist door het ontbreken van compatibiliteitsvoorzieningen voor oudere scripts. Met 5.4 is een tussenfase gecreëerd.

Lekker complex, want PHP 6 was al half gelanceerd, toen het weer werd teruggetrokken. PHP 5.4 werd in het leven geroepen om de bereikte ontwikkelingen in onder te brengen. Leest u in dit boek, of ergens op het web ‘PHP 6’, dan kunt u daar vrijwel altijd ook 5.4 voor invullen.

Moet u nu overstappen? Dat hangt af van de webserver waarop uw applicatie draait. Om te leren pakt u natuurlijk altijd de nieuwste versie. Om te ontwikkelen zult u voorlopig waarschijnlijk met 5.3 moeten werken, want dat is momenteel de meest gangbare versie.

Een nieuwe Basiscursus

Dit boek is een echte Basiscursus voor mensen die voor het eerst gaan programmeren met PHP en zelfs voor mensen die voor het eerst met het fenomeen programmeren in aanraking komen.

Hebt u nog nooit een regel code geprogrammeerd, dan kunt u met dit boek uw eerste schreden zetten in de wereld van het binaire denken. Bent u nieuw op het gebied van PHP, dan smeedt u met dit boek een solide frame om webapplicaties mee op te bouwen. Bent u al bekend met een oudere versie van PHP, dan helpt dit boek u met het aanwennen van nieuwe gewoonten en goede gebruiken.

Dit boek loodst u stap voor stap door het oerwoud van mogelijkheden met PHP en de aanverwante programmeertalen als HTML en SQL. Een basiskennis van HTML is echter wel handig.

Waarvoor gebruikt u PHP?

PHP is zonder twijfel de meest gebruikte server-sided programmeertaal in de wereld. PHP automatiseert allerlei processen aan de achterkant van een website, zoals het reageren op verzoeken, gegevens vastleggen in een database of het verzenden van mail. PHP gebruikt u dus als u een website bouwt die meer moet kunnen dan statische teksten en plaatjes weergeven. Zodra een website dynamisch wordt, komt PHP om de hoek kijken. Fameuze systemen als WordPress en Joomla zijn volledig geschreven in PHP.

Indeling van dit boek

In de eerste vijf hoofdstukken van dit boek zorgt u dat uw Windows-computer als ontwikkelbasis kan dienen voor websites. Uw pc wordt als het ware ingericht als webserver. In de hoofdstukken 6 tot en met 12 krijgt u een brede en grondige basis voor het werken met PHP, waarbij ook complexe onderwerpen behandeld worden. In de hoofdstuk 13 en 14 worden de laatste puntjes op de i gezet en krijgt u bronnen voor meer informatie aangereikt.

De scripts die in de hoofdstukken 10 tot en met 14 worden gemaakt, zijn via www.academicservice.nl te downloaden in een gecompriemde map. U kunt uw eigen resultaten daar handig mee vergelijken en verder bouwen aan de projecten.

In dit boek komen tips, waarschuwingen of opmerkingen voor. Deze staan in een kader en zijn te herkennen aan de volgende symbolen: 

Toetsen die u indrukt of sneltoetscombinaties staan in een **vet** lettertype. Bijvoorbeeld: druk op de **Enter** of druk op **Ctrl+S**. Teksten en opdrachten die u letterlijk moet invoeren worden eveneens **vet** weergegeven. Bijvoorbeeld: geef het bestand de naam **index.php**.

De namen van menu's, opties, dialoogvensters, knoppen en werkbalken worden in dit **lettertype** weergegeven. Bijvoorbeeld het dialoogvenster **Opslaan als** of de knop optie **Next**. PHP taal is herkenbaar aan dit lettertype.

1 Aan de slag met PHP

PHP is voor een interactieve website wat de motor is voor een auto. Hoe mooi een auto ook is vormgegeven, hoe schitterend de lak en hoe comfortabel de stoelen ook zijn, je hebt er niets aan als er onder de motorkap geen goed lopende motor ligt.

Zou u in deze vergelijking HTML en CSS vergelijken met het uiterlijk van de auto – met HTML en CSS bouwt u immers de zichtbare kant van een website – zo kunt u PHP vergelijken met de motor die onmerkbaar in de achtergrond voor allerlei zaken zorgt, waar de argeloze gebruiker geen weet van heeft.

1.1 In dit hoofdstuk

- Kennismaken met begrippen als server en client
- Andere talen voor het web
- Het paradigma Model, View, Controller
- Objectgeoriënteerd programmeren
- Databases en MySQL
- WAMP – de webserver voor uw pc
- Editors voor PHP
- Denken als een computer

1.2 Webserver en browser

Websites worden altijd geserveerd door een webserver. De webserver stuurt alle benodigde gegevens voor een website over het internet naar uw computer, waarin de browser de ontvangen webpagina weergeeft – dat is de client.

De webpagina die u ziet is opgebouwd uit HTML en andere talen voor de browser en bevat geen enkele regel PHP-code. Uw browser zou ook niet weten wat hij met PHP-code aanmoet. De webserver is het domein van PHP. Met PHP-code bepaalt de programmeur welke handelingen de server moet verrichten als een bezoeker in de browser op een knop of hyperlink klikt. De PHP-code zorgt voor afhandeling van webformulieren, bewaart gegevens van de bezoekers in een database en stuurt weer nieuwe gegevens als HTML-code naar de browser.

PHP is dus niet een op zichzelf staande programmeertaal. De programmeur zorgt dat zijn (of haar) PHP-code HTML-code genereert, of CSS-code (voor stijlbladen) of zelfs JavaScript-code.

1.3 HTML, Ajax, XML, CSS en JavaScript

PHP is een taal met veel mogelijkheden, maar kan weinig uitrichten zonder de drie basistalen voor browsers: HTML, CSS en JavaScript. PHP doet eigenlijk niets anders dan verzoeken die van de gebruiker komen, verwerken en opslaan, en vervolgens daarop reageren door het terugzenden van stukken HTML en JavaScript. Het stijlblad (gemaakt met CSS) zorgt voor het juiste uiterlijk van de HTML-pagina.

Ook XML speelt hierin een rol. XML is een taal waarmee gegevens tussen server en browser gestructureerd uitgewisseld kunnen worden. De XML-structuur zorgt ervoor dat de ontvangen gegevens correct begrepen en verwerkt kunnen worden. Ajax zorgt in de browser meestal voor verzending en ontvangst van met XML gecodeerde gegevens.

In dit boek ontkomt u dus niet aan de nodige HTML-code. CSS, XML en JavaScript zult u minder tegenkomen, omdat die met de browserafhandeling te maken hebben, wat dus niet het domein van PHP is. Wel komt aan de orde hoe u met PHP JavaScript-code en CSS-stijlelementen aan HTML kunt toevoegen. Kennis van HTML is onontbeerlijk om zelfstandig met PHP te kunnen werken. In dit boek wordt uitgegaan van praktische kennis van HTML.

1.4 WordPress, Joomla en PHPBB

PHP mag gerust martkleider worden genoemd als ontwikkeltaal voor het web. Er zijn natuurlijk vele programmeertalen die zich lenen voor het ontwikkelen van interactieve websites, zoals Perl, Java, ASPnet, Ruby on Rails, of Django. PHP heeft zich in de loop der jaren echter op een aantal fronten bewezen als 'winnaar'. De taal is veruit het meest toegankelijk, is zeer flexibel en vergevingsgezind voor de programmeur, draait op vrijwel elke webserver – ook bij heel goedkope hostingpakketten – en wordt toegepast in veel grote platforms als WordPress, Joomla en PHPBB. Wie goed kan programmeren in PHP, is in deze tijden verzekerd van een goede bron van inkomsten. Aan het eind van deze Basiscursus bent u al een heel eind op weg om dat te bereiken.

1.5 PHP is een rommelpot

Voor dat laatste is het echter wel nodig dat de programmeur meer kan dan wat regels code achter elkaar plakken. Waarmee ook meteen de grote zwakte van PHP wordt aangestipt. Doordat PHP zo flexibel is als elastiek, nodigt het ook uit om 'snel' even wat code te bakken, ad hoc-aanpassingen aan een webproject te doen of zonder vooropgezet plan iets in elkaar te knutselen.

Veel doe-het-zelfprogrammeurs gaan op deze manier te werk en menigeen durft zichzelf PHP-programmeur te noemen zonder planmatig te kunnen programmeren. Waakt u ervoor in deze zelfde valkuil te springen. Zomaar iets programmeren, betekent meestal beduidend meer nawerk en soms zelfs helemaal opnieuw beginnen, voordat het project ook goed draait.

1.6 Model, View, Controller

Dit boek gaat om deze reden niet alleen over een grondige basis van PHP, maar ook over de basis van goed programmeerwerk.

Een goed gebruik bij allerlei ontwikkelplatformen en frameworks voor het web, is ontwikkelen volgens het paradigma MVC: Model, View, Controller. Daarvoor is een goede reden: het web werkt namelijk volgens dit paradigma. In het kort komt het erop neer, dat de View bepaalt wat de gebruiker van een website in zijn browser ziet; het Model bepaalt hoe de gegevens die in de website gebruikt worden, op de server zijn opgeslagen; de Controller koppelt beide zaken aan elkaar en regelt wat er moet gebeuren met de input van de gebruiker.

Ook als u een heel klein programmaatje voor een website maakt, bijvoorbeeld een peiling (poll) of een gebruikerslogin, kunt u werken met dit paradigma. Het grote voordeel is dat uw systeem weinig fouten of verrassingen kan bevatten en bijvoorbeeld dat het makkelijk uitbreidbaar is.

1.7 OOP

Goed en gestructureerd programmeren betekent ook dat u gebruikmaakt van klassen en objecten. Objectgeoriënteerd programmeren is niet voorbehouden aan complexe talen als Java of C en is ook niet alleen bedoeld voor grote projecten. Het kleinste project, zoals de eerdergenoemde peiling of gebruikerslogin heeft voordeel bij het werken met klassen. Klassen maken een project overzichtelijk en eenvoudig te veranderen of uit te breiden. Klassen maken het bovendien makkelijk om eerder ontwikkelde onderdelen in nieuwe projecten toe te passen. Minder dubbel werk dus. Ook Object Oriented Programming (OOP) wordt in dit boek besproken en waar mogelijk toegepast.

1.8 SQL en MySQL

Een PHP-project is nauwelijks denkbaar zonder database. De database bewaart en ordent de gegevens van de gebruikers. PHP is geen database en heeft ook geen database. Daarvoor moet u dus gebruikmaken van een van de bekende databases voor het web: SQLite, MySQL, Postgre SQL of Oracle. MySQL wordt veruit het meest gebruikt op gewone webservern en is bovendien gratis. Een voor de hand liggende keuze dus. Maar daarmee bent u er nog niet.

PHP zelf kan geen database aanspreken. Daarvoor is speciale databasetaal nodig: SQL. Het leuke van SQL is dat alle genoemde databases deze verstaan.

In dit boek leert u tevens de basis van SQL, omdat SQL en PHP vrijwel onlosmakelijk met elkaar verbonden zijn.

1.9 WAMP, LAMP, MAMP op uw eigen computer

Om PHP-projecten te kunnen draaien, hebt u eigenlijk een webserver nodig. PHP draait immers op een webserver en niet in de browser. Op het web zijn allerlei manieren te vinden om van uw eigen computer een webserver te maken. Zo'n webserver is echter volstrekt niet geschikt voor het permanent hosten van een website op het web – daar komt veel meer bij kijken – maar is prima voor het ontwikkelen van webprojecten.

De meest hapklare oplossing is zonder twijfel WAMP voor Windows of MAMP voor de Mac. WAMP is het prachtige acroniem voor Windows Apache (de webserver), MySQL (de database) en PHP. XAMP is de OS X-versie daarvan.

Werkt u met een Linux-werkstation, dan bent u ongetwijfeld ook bekend met de ingebouwde Apache-server en MySQL-component. In dit boek wordt uitgegaan van Windows als werkcomputer. In een volgend hoofdstuk wordt WAMP geïnstalleerd als webserver. Daarin komt ook aan de orde hoe u WAMP met PHP 6 kunt installeren – een hapklare versie van WAMP met PHP 6 is bij het ter perse gaan van dit boek nog niet beschikbaar.

1.10 Editors

Een ander punt van belang bij het programmeren in PHP is de tekstverwerker waarmee u de code schrijft. PHP wordt geschreven en opgeslagen als gewone tekst. Met Windows Kladblok kunt u dus PHP-code schrijven. Met WordPad of Word kan dat niet, omdat deze editors allerlei opmaakcode onzichtbaar rond uw teksten plaatsen.

Het kan natuurlijk wel veel handiger dan met Kladblok. Er zijn editors die u helpen met het indelen van de PHP-code. Er zijn er die helpen PHP-code voor u te typen en er zijn editors die complete projecten met meerdere bestanden voor u managen en zelfs de code voor u kunnen uitvoeren om te debuggen (fouten opsporen en oplossen). In een volgend hoofdstuk komen deze editors aan de orde. In dit boek wordt een eenvoudige editor gebruikt, zodat u zich kunt concentreren op het leren van PHP en niet eerst ook nog eens verward raakt door de duizend mogelijkheden van de editor.

1.11 'You will be assimilated'

Leren programmeren is leren denken als een computer. Liever zouden we de computer leren denken als een mens, maar aangezien mensen het onderling al niet eens kunnen worden over 'hoe te denken', is de eerste optie veruit het handigst. U moet uw denken dus aanpassen aan de beperkingen van de computer. You will be assimilated...

U moet als programmeur dus denken als een computer. Dat is binair denken, tweewaardig denken. Iets kan, of iets kan niet. Iets staat aan of uit. Een beetje, misschien, of ongeveer bestaan niet in het binaire denken. Wilt u programmeren, dan moet u denken in keuzes en cycli. Laten we dit concept eens loslaten op de manier waarop kinderen geleerd wordt over te steken:

```
kijk naar links
kijk naar rechts
kijk weer naar links
oversteken
klaar
```

Een mooie methode, maar zou een computer deze methode toepassen, dan gaat het dus meteen al mis als er een auto aankomt. We zijn namelijk in de code vergeten in te calculeren, dat er niet alleen gekeken maar ook geëvalueerd moet worden. Kijken is niet voldoende, er moet bepaald worden wat de computer ziet en er moeten beslissingen genomen worden. Dat wordt dan dus als volgt:

```
kijk naar links
ALS( er niets aankomt ) DAN:
    kijk naar rechts
    ALS( er niets aankomt ) DAN:
        kijk naar links
        ALS( er niets aankomt ) DAN:
            oversteken
KLAAR
```

Dat lijkt al beter. Er wordt immers nu alleen maar overgestoken als er echt van links en rechts niets aankomt. Maar wat moet onze computer doen, als er wel iets aankomt? Ook dat moet worden ingebakken:

```
kijk naar links
ALS( er niets aankomt ) DAN:
    kijk naar rechts
    ALS( er niets aankomt ) DAN:
        kijk naar links
        ALS( er niets aankomt ) DAN:
            oversteken
```



```

        ANDERS:
            Wacht
        KLAAR
    ANDERS:
        wacht
    KLAAR
ANDERS:
    wacht
KLAAR

```

1.11.1 Syntaxis

Door regels code in te laten springen en door het woordje KLAAR toe te voegen, moet in het voorbeeld hierboven duidelijk zijn wat er in deze code bij de verschillende voorwaarden hoort. U ziet dat het bij dit kleine stukje code al minder overzichtelijk wordt, nu er meerdere voorwaarden in elkaar genest zijn. Tijd dus om meteen maar over te stappen op de syntaxis van PHP. Woorden zoals ALS en ANDERS worden dus ook meteen in het Engels vermeld, want PHP is zoals vrijwel alle programmeertalen opgebouwd in het Engels.

Een syntaxis definieert de manier waarop een programmeertaal (of eigenlijk elke taal) is opgebouwd. De ene programmeertaal gebruikt inspringen om structuur in de code aan te brengen (zoals Python), PHP gebruikt accolades, haakjes en puntkomma's hiervoor. De ene programmeertaal gebruikt woorden als IF THEN ELSE voor de beslissingen, terwijl andere (waaronder PHP) het ook zonder THEN kunnen. Oversteken in PHP zou er als volgt uit kunnen zien – met natuurlijk hier en daar nog veel Nederlands ertussen, waar geen PHP-interpret iets van snapt:

```

kijk_naar_links;
if( er_niets_aankomt ){
    kijk_naar_rechts;
    if( er_niets_aankomt ){
        kijk_naar_links;
        if( er_niets_aankomt ){
            oversteken;
        }else{
            wacht;
        }
    }else{
        wacht;
    }
}else{
    wacht;
}

```

Dezelfde code, maar nu meer in PHP-stijl. Deze beslissingsstructuur is opgebouwd rond de IF-THEN-ELSE-structuur. In PHP ziet die er als volgt uit:

```
if( voorwaarde ){
    opdrachten_1;
}else{
    opdrachten_2;
}
```

Als aan de voorwaarde wordt voldaan, wordt de reeks opdrachten_1 uitgevoerd. Dat kan één regeltje code zijn, maar ook een compleet programma van duizenden regels. Als niet aan de voorwaarde wordt voldaan, wordt de reeks opdrachten_2 uitgevoerd – dat zijn dus de opdrachten tussen de accolades bij else.

De cursieve tekst in de code is natuurlijk geen PHP. PHP houdt ook niet van spaties in namen, dus hier zijn underscores (_) geplaatst. Het inspringen is niet nodig voor de werking van een PHP-programma, maar wel voor de leesbaarheid van de code voor u als programmeur, of voor iemand anders die na u met de code verder wil. De puntkomma's (;) achter de statements zijn wel noodzakelijk voor PHP. Ze maken het ook mogelijk om code anders op te maken:

```
kijk_naar_links; if( er_niets_aankomt ){ kijk_naar_rechts; if( er_niets_aankomt ){
kijk_naar_links; if( er_niets_aankomt ){ oversteken; }else{ wacht; }else{ wacht;
}}else{ wacht;}
```

Wat PHP betreft gebeurt er nu hetzelfde als in het vorige oversteekvoorbeeld, maar voor een programmeur is deze code onleesbaar. Dit soort broddelwerk vraagt gewoon om fouten.

1.11.2 Herhaling

Helaas staat onze computer met deze code, ondanks alle duidelijke voorwaarden en beslissingen nog steeds op de stoep te wachten en is nog niet overgestoken. Er kwam namelijk een fietser van links. De computer moest dus de opdracht wacht uitvoeren en toen was de code afgelopen.

We zijn vergeten te vertellen dat hij deze beslissingsstructuur net zolang moet herhalen, tot hij met succes is overgestoken. Er moet dus naast alle conditionals (voorwaarden) ook nog een loop (lus) worden gemaakt. En niet zomaar een loop maar een conditional loop – een lus met een voorwaarde. Namelijk de voorwaarde nog_niet_overgestoken. Want zolang de computer niet is overgestoken, moet gekeken worden naar links en naar rechts, net zolang tot wel overgestoken kan worden. De benodigde code kan ook meteen in correct PHP worden gezet:

```

while( nog_niet_overgestoken ){
    kijk_naar_links;
    if( er_niets_aankomt ){
        kijk_naar_rechts;
        if( er_niets_aankomt ){
            kijk_naar_links;
            if( er_niets_aankomt ){
                oversteken;
            }else{
                wacht;
            }
        }else{
            wacht;
        }
    }else{
        wacht;
    }
}

```

1.11.3 Vele wegen naar Rome

Om heelhuids over te steken zijn er in programmacode heel wat mogelijkheden. Zo is in het voorbeeld hierboven ervoor gekozen om de drie beslissingen te nesten. Dat is ook het meest logisch, omdat het weinig zin heeft om verkeer van rechts te controleren als er van links al een vrachtwagen aan komt denderen. In programmacode kan het echter toch handig zijn om zo'n beslissingstructuur anders op te bouwen, zonder dat het resultaat anders wordt:

```

while( nog_niet_overgestoken ){
    kijk_naar_links;
    if( er_iets_aankomt ){
        continue;
    }
    kijk_naar_rechts;
    if( er_iets_aankomt ){
        continue;
    }
    kijk_naar_links;
    if( er_iets_aankomt ){
        continue;
    }
    oversteken;
}

```

Deze code is al een stuk beter leesbaar. De opdracht `continue` is een PHP-opdracht die ervoor zorgt dat alle volgende opdrachten binnen de `while`-lus niet meer worden uitgevoerd en dat de lus vanaf het begin gecontinueerd wordt, al-

thans, als aan de conditie van de `while`-lus voldaan wordt. Komt er dus iets van links, dan wordt er niet meer opnieuw naar rechts of links gekeken, maar begint de lus opnieuw met naar links kijken. De ‘menselijke’ opdracht wacht uit de code hiervoor deed natuurlijk niets. Want tijdens dat wachten wordt gewoon opnieuw geobserveerd.

1.11.4 Functies

In feite wordt dus driemaal dezelfde handeling uitgevoerd: kijken en oordelen. Nu gebruiken we de computer om ons werk makkelijker te maken en vooral om minder herhalend werk te doen. Daar kunnen we dus nu meteen mee beginnen.

Vrijwel alle moderne programmeertalen hebben een manier om stukken code te bundelen. Naast allerlei programmeertechnische voordelen (dit onderdeel komt later aan de orde), maakt dit de code ook beter leesbaar. In ons oversteekprogramma zou het onderdeelje ‘kijken en oordelen’ als volgt kunnen worden:

```
function erKomtIetsVan($richting){
    kijk_naar($richting);
    return(er_iets_aankomt);
}
```

Alleen de cursieve woorden zijn nu nog ‘mensentaal’. De functie (`function`) `erKomtIetsVan($richting)` is gedefinieerd volgens de PHP-syntaxis:

- `function` – hiermee wordt een nieuwe functie gedefinieerd;
- `erKomtIetsVan` – de naam van de functie;
- `()` – staat altijd achter de naam van een functie, tussen de haken staan eventuele parameters;
- `$richting` – de naam van een variabele (die in dit voorbeeld staat voor de richting waarin gekeken moet worden). De naam van een variabele begint in PHP altijd met het dollarteken (`$`).
- tussen de `{` en `}` staan de opdrachten die achtereenvolgens in de functie worden uitgevoerd;
- `return` – geeft een waarde terug aan het eind van de functie (in dit voorbeeld dus of er iets aankomt of niet – Ja of Nee bijvoorbeeld).

In gewone mensentaal komt het erop neer dat deze functie naar `richting` kijkt net als in de voorbeelden hierboven en het antwoord teruggeeft met de opdracht `return`. Dat antwoord kan zijn: Ja, er komt iets aan, of Nee, er komt niets aan.

Ook het testen of we overgestoken zijn, moet natuurlijk met een functie gebeuren:

```
function nietOvergestoken(){
    return(nog_niet_overgestoken);
}
```

Tot slot moet ook nog het oversteken zelf in een functie worden ondergebracht:

```
function veiligOversteken(){
    heel_veel_moeilijke_code_om_over_te_steken;
}
```

De code van ons oversteekprogramma ziet er met deze drie nieuwe functies als volgt uit:

```
while( nietOvergestoken() ){
    if( erKomtIetsVan('links') ){
        continue;
    }
    if( erKomtIetsVan('rechts') ){
        continue;
    }
    if( erKomtIetsVan('links') ){
        continue;
    }
    veiligOversteken();
}
```

Dit stuk code bevat geen cursieve tekst, omdat het een correct werkend stuk PHP-code zou kunnen zijn.

In de functies wordt dus het kijken, het oversteken en het resultaat geregeld en in de while-lus de hoofdstructuur. De waarden 'links' en 'rechts' staan tussen aanhalingstekens. Deze worden binnen de while-lus als parameters meegegeven aan de functie `erKomtIetsVan()`, waarin ze worden bewaard in de variabele `$richting`.

Het hoofdprogramma is duidelijk leesbaar dankzij de veelzeggende namen van onze functies, zoals `erKomtIetsVan()` en `veiligOversteken()`. Maak er een gewoonte van om de namen van functies, variabelen en klassen op deze manier duidelijk en beschrijvend te maken.

1.11.5 Complexe factoren

Zolang er links of rechts iets aankomt, zal onze computer blijven wachten op de stoep. En telkens nadat er iets aankwam, begint het hele verhaal weer opnieuw. En zo hoort het ook als een computer moet leren oversteken. Want risico's incalculeren, is voor 'gevorderde overstekers'. De bijbehorende code van een ge-

vorderde oversteker is ook een stuk complexer. De risico's worden bepaald aan de hand van de conditie (`er_iets_aankomt`). Bij het calculeren van het risico betrekken we ook hoe snel iets nadert. Een voortrazende auto geeft een ander risico dan oma op de fiets.

Heeft de gevorderde oversteker haast, dan kan ook nog aan de hand van de snelheid van het naderende object worden bepaald hoe snel het oversteken moet gebeuren. Ons eenvoudige stukje code wordt dankzij onze mooie structuur met de drie functies echter nauwelijks anders als we al deze factoren gaan toevoegen.

1.11.6 Denken als een computer

Inmiddels hebt u, denkend als een computer, de computer geleerd veilig over te steken. U hebt ook al kennisgemaakt met conditions, loops en functions en hebt de variabele `$richting` gebruikt. Kortom: programmeren in een notendop. In de loop van dit boek komen deze en nog veel meer zaken uitgebreid aan de orde.

Onze code om over te steken zit nog vol met 'menselijke' aspecten. PHP begrijpt natuurlijk nog geen snars van het statement 'er iets snels aankomt' of van het statement 'oversteken'. Ook 'kijk naar links' betekent weinig.

Stel dat onze computer al een oog zou hebben om te kijken en stel dat de computer al onderscheid kan maken tussen een stoep, een lantaarnpaal en een voortrazende auto, dan nog moet er heel wat code geschreven worden om de vraag te beantwoorden: 'er iets aankomt' en hoe snel dat dan is en wanneer het object ons pad kruist.

Maar uitgaande van het bestaan van al deze technologie, zou je de computer inderdaad kunnen vragen: komt er een object aan (1), hoe ver is dat object nu verwijderd (2) en op welk moment kruist dat object ons pad (3)?

Want aan de hand van deze drie gegevens kan worden berekend met welke snelheid de lijn van een eventueel aankomende object moet worden gekruist, om het object niet te raken (1), en of die snelheid voor de wandelaar haalbaar en wenselijk is (2). Het oversteken zelf is ook nog weer een 'menselijke' handeling.

Maar hoe complex al deze zaken ook zijn: aan de structuur van ons simpele hoofdprogramma in de `while`-lus hoeft weinig te veranderen om ook deze complexe factoren toe te voegen. Daarvoor zijn de drie functies, die eindeloos zijn uit te breiden. In feite hoeft alleen maar een oversteekadvies te worden bepaald dat zegt: niet oversteken, of oversteken met een bepaalde snelheid, die dan weer binnen de gewenste en mogelijke snelheid van de wandelaar ligt.

1.12 Top-down refinement, stapsgewijze verfijning

Deze methode van programmeren heet top-down refinement. Eerst de grote lijnen vastleggen, waarbij goed nadenken en uittekenen van de structuren erg belangrijk is, en dan pas de details aanbrengen. Net als het ontwerpen van een gebouw:

- Eerst wordt bepaald wat de mogelijkheden van een gebouw moeten zijn en of er nog uitbreidingen mogelijk moeten zijn.
- De architect bedenkt een passende structuur, maar bemoeit zich niet met waar welke baksteen moet liggen.
- De bouwkundige berekent en tekent de technische structuur en de substructuren, maar bemoeit zich evenmin met waar welke baksteen moet liggen.
- De uitvoerder bepaalt wie wat, wanneer en waar doet binnen de gedefinieerde structuren, wie muren metselt, wie kozijnen timmert en in welke volgorde dat moet gebeuren, maar bemoeit zich nog steeds niet met waar welke baksteen moet liggen.
- de metselaar legt tot slot stenen en cement op de juiste plek en bepaalt welke baksteen waar ligt.

Top-down refinement is een handige manier om te programmeren met structuur en zonder later tegen verrassingen aan te lopen. Simpelweg beginnen met code bakken, is vrijwel altijd een lange en heilloze weg. Komen er later factoren bij, dan moet opnieuw naar de hoofdstructuur van het programma gekeken worden om de nieuwe zaken in te voegen.

1.13 Tot slot

Hopelijk bent u inmiddels warmgelopen om met PHP aan de slag te gaan. In het volgende hoofdstuk wordt WAMP gedownload en geïnstalleerd, zodat u op uw eigen computer met PHP, MySQL en de browser uw site kunt ontwikkelen en testen.

Register

Symbolen

! 111
!= 111
\$_COOKIE 156
\$_SESSION 164
\$GLOBAL 132
\$GLOBALS 127
* 228
// 59
<?php?> 59, 63
<body> 50
<head> 49
<html> 49
<link> 50
<title> 50
= 111
== 111
=== 111
\ (backslash) 62
\n 70
__construct() 186

A

aanhalingstekens 61
 dubbele 61
 enkele 61
abs() 84
afgeleide klassen 193
Ajax 4
alfabet 50
AND 228
argument 97
array 74, 90, 103, 129, 261
 associatief 93
 element 91
 foreach 129
 functie 94
 itereren 129
 sleutel 91

 speciale 96
 waarde 91
array-functie 94
 array_key_exists() 95
 arsort() 95
 asort() 95
 count() 95
 in_array() 95
 krsort() 95
 ksort() 95
array_key_exists() 95, 262
array_walk() 262
arsort() 95
ASCII 227
ASCII 50
asort() 95
associatieve arrays 93
asterisk (*) 228
attribuut 117
 checked 117
 onchange() 253
 onclick 254
automatisch 155

B

bereik 121, 125
 Engels: scope 121
 testen 125
bestandsextensies 30
binair denken 7
body 50, 51
 -tag 48
bool 92
booleaanse algebra 84
booleaanse operatoren 85
boolean 74, 84
 onwaar 84
 rekenen 86
 waar 84
break 102, 109

broncode 60
 bronnen 257
 browser-gegevens 266
 By Ref 263
 By Reference 126
 By Value 126, 263

C

call_user_func() 196
 case-blok 102
 checked 117
 compiler 36
 constante 73, 89
 constanten 89, 122
 () 90
 underscore 90
 continue 109
 controller 5, 114, 166, 172
 conversie 87
 cookie 151, 158
 dialoogvenster 153, 155
 functie 155
 global 156
 PHPSESSID 158
 sessie 158
 superglobal 156
 verlopen 155
 count() 95
 CSS 4, 253
 -sijlblad 47, 50
 bestand 50
 code 50
 document 50
 sijlblad 50
 current() 262

D

database 5, 205, 206
 -verbinding 218
 basishandelingen 242
 maken 210
 relationeel 241
 twee tabellen 241
 datatype 73, 74, 87, 90, 212
 arrays 90
 conversie 87
 enkelvoudig 74
 meervoudig 74, 90
 date() 143, 145, 146, 266
 datetime() 221

datum 143
 db-klasse 229
 default 102
 DELETE 221, 226, 242
 DESC 227
 die() 109
 do-while 107
 dobbelen 114
 Doctype 49
 double 74, 82
 Download
 Wampserver 16
 dubbele aanhalingstekens 61
 dubbele slashes 59

E

echo 59, 60
 -opdracht 123
 eclipse 38
 editors 6
 else 101
 elseif 101
 em 53
 end() 262
 endif 101
 enkele aanhalingstekens 61
 enkelvoudige datatypen
 Boolean 74
 Float 74
 Integer 74
 String 74
 escape-teken 76
 exit() 109
 expressie 97, 111
 ! 111
 != 111
 = 111
 == 111
 === 111
 expression (expressie) 97
 extends 194

F

false 84
 favicon 50
 float 74, 80
 floor() 84
 for 108
 foreach() 92, 108, 129, 198

- formulier 167
 - verwerken 171
 - FROM 227
 - func_get_arg() 264
 - func_num_args() 264
 - functie (function) 11, 121, 124, 133, 263
 - __construct() 186
 - abs() 84
 - array_key_exists() 262
 - array_walk() 262
 - break 109
 - call_user_func() 196
 - continue 109
 - cookie 152
 - current() 262
 - date() 143, 145, 146, 266
 - datetime() 221
 - declareren 122
 - die() 109
 - end() 262
 - exit() 109
 - floor() 84
 - foreach() 198
 - func_get_arg() 264
 - func_num_args() 264
 - getallen 83
 - getdate() 143, 147
 - gettimeofday() 144, 147
 - gmdate() 146
 - gmmktime() 149
 - goto label 109
 - in_array() 262
 - include() 140
 - isset() 114, 157, 262
 - mail() 265
 - mktime() 144, 148
 - mod() 82
 - mt_rand() 84, 100
 - mysql_affected_rows 227
 - mysql_close() 219
 - mysql_connect() 219
 - mysql_fetch_array() 227
 - mysql_num_rows 227
 - mysql_query() 227
 - mysql_select() 219
 - next() 262
 - phpinfo() 267
 - preg_match() 263
 - prev() 262
 - querydb() 232
 - rand() 84
 - recursie 135
 - registerglobals() 115
 - reset() 262
 - return() 109
 - round() 84
 - session_destroy() 159, 164
 - session_id() 158
 - session_start() 158, 164
 - session_unset() 164
 - setcookie() 152, 164
 - setlocale() 266
 - shuffle() 262
 - signin() 172
 - signup() 172
 - sort() 262
 - str_replace() 78
 - strftime() 146, 266
 - stripos() 160
 - strlen() 78
 - strtolower() 78
 - strtoupper() 78
 - substr() 79
 - time() 143, 144, 145, 221
 - toon_formulieren() 187
 - toondatum() 146
 - toontijd() 146
 - variabelen 121
 - wordwrap() 80
 - functieaanroep (function call) 123, 196
 - function (functie) 97, 133
-
- ## G
-
- gebruiker aanmaken 208
 - gebruikersbeheer 232
 - GET 115, 156, 167
 - getallen
 - afkappen 83
 - afronden 83
 - random-getallengenerator 83
 - getdate() 143, 147
 - gettimeofday() 144, 147
 - global 127, 131, 156
 - \$_COOKIE 156
 - gmdate() 146
 - gmmktime() 149
 - goto 109
 - goto label 109
 - Greenwich Mean Time 146, 149
-
- ## H
-
- head 49
 - header 49

header() 180
 heredoc 64
 hoofdlettergevoelig 90
 href 62
 HTML 4, 47, 49
 -broncode 60
 -formulieren 167
 in PHP 57
 opmaak sjabloon 51
 pagina 47
 sjabloon 48
 HTML 4 47
 HTML 5 47

I

id 207
 if 99, 100
 if-statements 242
 if-then-else 99, 265
 -structuur 9
 in_array() 95, 262
 include() 67, 140
 include_once() 67, 69
 init 113
 initialiseren 113
 inlogscript 165, 178
 INSERT 221, 242
 installeren als addon 17
 integer 74, 80, 81
 long 145
 rekenen 82
 string koppelen 81
 isset() 114, 157, 262
 itereren 92, 129

J

jargon 97
 JavaScript 4, 53, 119, 151, 253
 bestand 53
 code 53
 cookie 151
 pagina 53

K

keuzestructuur 102
 kindklasse 193
 kladblok 31
 klassen 5, 183, 239

knop
 dobbelen 114
 opnieuw 114
 krsort() 95
 ksort() 95

L

labelnaam 64
 LAMP 15
 LIKE 228
 LIMIT 227
 link 50
 lokale variabele 126
 long 145
 lus 92
 break 109
 for 108
 foreach() 92, 108
 oneindige 110

M

magic constants 90
 mail() 265
 MAMP 6, 15
 meervoudige datatypen 74
 Array 74
 Object 74
 meta-tag 49, 50
 Microsoft SQL-server 205
 mktime() 144, 148
 mod() 82
 Model 5
 mt_rand() 84, 100
 MTA (Mail Transfer Agent) 265
 MUA (Mail User Agent) 265
 MVC 5, 119
 MVC – model, view, controller 119
 MySQL 5, 27, 205, 215, 218, 239
 data maken 215
 voorbereiden PHP-project 218
 MySQL-database 27, 205
 beheer 27
 mysql_affected_rows 227
 mysql_close() 219
 mysql_connect() 219
 mysql_fetch_array() 227
 mysql_num_rows 227
 mysql_query() 227
 mysql_select() 219

N

netbeans 38
 next() 262
 NOT 228
 NOW() 221
 nowdoc 64

O

object 74, 183, 202
 objectgeoriënteerd programmeren 183
 onchange() 253
 onclick 254
 oneindige lus 110
 ontwikkelomgevingen 38
 OOP – Object Oriented Programming 5, 183
 afgeleide klassen 193
 kindklasse 193
 klassen 183
 objecten 183
 opdracht 97
 header() 180
 include 67
 return 134
 operator 97
 opmaak
 HTML-sjabloon 51
 opnieuw 114
 op tijd 155
 OR 228
 Oracle 205
 ORDER BY 227
 or die() 219

P

paradigma 119
 MVC 119
 parameter 97, 125, 127, 128, 129, 263
 Pascal 133
 functie 133
 procedure 133
 PCRE 263
 PHP 151, 258
 code 36, 59
 cookie 151
 documenten 239
 function 133
 in html 57
 interpreter 36, 59
 JavaScript 119

mail versturen 265
 parser 64
 procedure 133
 project 218
 PHP-editor 37
 Scintilla 39
 Scite 39
 Smultron 39
 php.exe 36
 php.ini 267
 PHP_INT_MAX 82
 PHP 4 186
 PHP 5 161
 PHP 5.3 15, 164
 PHP 5.4 161, 164
 PHP 6
 snapshot 24
 phpinfo() 267
 phpMyAdmin 207
 PHPSESSID 158
 POSIX 263
 POST 115, 156
 PostGRE SQL 205
 precedence 260
 predefined variables 97
 preg_match() 263
 prev() 262
 Primary ID 212
 print 60
 procedure 133

Q

query's 205
 querydb() 232

R

rand() 84
 random 83
 real 82
 record 206
 recursie 124, 135
 recursieve functie 136
 regel 206
 regelen 57, 70
 regexps 262
 registerglobals() 115
 Regular Expressions 262
 rekenen 86, 143
 datum 143
 tijd 143

rekenteken 97
 relationele database 212, 241
 reset() 262
 RET 138
 retourwaarden 133
 return 127, 134
 return() 109
 round() 84

S

Scintilla 39, 41
 Scite 39
 installeren 41
 koppelen 42
 scope 121
 SELECT 221, 226, 242
 server Offline 23
 server Online 23
 sessie 151, 158, 164
 controleren 159
 functie 158
 onderhouden 164
 session 158
 session_destroy() 159, 164
 session_id() 158
 session_start() 158, 164
 session_unset() 164
 setcookie() 152, 164
 setlocale() 266
 shuffle() 262
 signin() 172
 signup() 172
 Site
 downloaden 39
 Smultron 39
 sort() 262
 SQL 6, 205
 -opdracht 221
 -statement 221
 database 206
 opdrachten 206
 record 206
 tabel 206
 SQL-functie
 NOW() 221
 SQL-opdrachten 221
 SQL-statement 221
 DELETE 221, 226
 INSERT 221
 SELECT 221, 226
 UPDATE 221, 225

statement 97
 statements 97
 static 263
 stijlblad 50, 51
 str_replace() 78
 strftime() 146, 266
 string 66, 74, 75, 78, 81
 functie 78
 integer koppelen 81
 lengte 78
 str_replace() 78
 strl_replace() 78
 strlen() 78
 strtolower() 78
 strtoupper() 78
 substr() 79
 waarde 125
 stripos() 160
 strlen() 78
 strtolower() 78
 strtoupper() 78
 Structured Query Language 205
 structuur
 do-while 107
 while 103
 stylesheet 50
 substr() 79
 superglobals 97, 122
 switch-case 102
 syntaxis 8
 syntaxiscorrecties 37

T

tabel 206, 211
 tabel ontwerpen 211
 tekstverwerker 6
 tijd 143
 tijdelijke internetbestanden 151
 time() 143, 144, 145, 224
 titelbalk 50
 title 50
 toon_formulieren() 187
 toondatum() 146
 toontijd() 146
 top-down refinement 14
 true 84
 type cast 87, 92, 97
 (bool) 92
 typecast (int) 116

U

uitloggen 175
underscore 90
Unique 212
UNIX-tijd 143
unset 88
UPDATE 221, 225, 242
UTF-8 50

V

variabele 66, 73
 \$GLOBAL 132
 GET 115
 global 131
 lokale 126
 POST 115
vergelijking 85, 86
verlopen
 automatisch 155
 op tijd 155
 via PHP 155
via PHP 155
view 5, 113, 117, 166

W

waarde 73
 \$GLOBALS 127
 false 84, 86
 global 127
 return 127
 true 84, 86

WAMP 6, 15, 207
Wampserver
 actief 21
 basiswebpagina 24
 inactief 21
 menu 25
 pictogram 21
 poort 21
webserver 3, 6
while 103
Windows Apache MySQL PHP 15
Windows Kladblok 6
WordPress 101
wordwrap() 80
wrapper 50, 51

X

XAMPP 208
XML 4
XOR 87

Y

Yahze-spel 112

Z

Zend Studio 38

De basis voor uw succes!

Als beginnende gebruiker maakt u in deze basiscursus kennis met PHP 5.4 en MySQL. PHP is een programmeertaal die in aanvulling op HTML bij uitstek geschikt is voor het bouwen van dynamische websites. En met de nieuwste versie van PHP gaat het programmeren nog gestructureerder en gestroomlijnder. Deze basiscursus is overigens ook geschikt voor gebruikers van PHP 5.3 en developers-versie 6.0.

Een greep uit de inhoud:

- Installeren van een webserver met PHP 5.4 en MySQL.
- Werken met de programmeertaal PHP 5.4.
- Objectgeoriënteerd programmeren met PHP 5.4.
- Werken met de database MySQL.
- Interactieve websites maken.
- Beveiligen van webpagina's.

De auteur heeft alle basisvaardigheden stapsgewijs en begrijpelijk uitgelegd en aangevuld met oefeningen. Zo leert u snel wat u in de praktijk nodig heeft.

Victor Peters is auteur van veel ICT-boeken, waaronder *Basicursus HTML5*.



ISBN 978 90 12 58499 9

NUR 989



www.academicservice.nl