



Programmeren met Java

VOOR
dummies[®]

Barry Burd



BBNC
uitgevers

Amersfoort, 2016

Inhoud

INLEIDING	1
Over dit boek	1
De gebruikte conventies in dit boek	2
Wat je niet hoeft te lezen	2
Wie ben jij?	3
De indeling van dit boek	4
Deel 1: Aan de slag met Java	4
Deel 2: Zelf Java-programma's schrijven	4
Deel 3: Het overzicht behouden: objectgeoriënteerd programmeren	5
Deel 4: Handige Java-technieken	5
Deel 5: Het deel van de tientallen	5
De pictogrammen in dit boek	5
Wat nu?	6
DEEL 1: AAN DE SLAG MET JAVA	7
HOOFDSTUK 1: Alles over Java	9
Wat je met Java kunt doen	10
Redenen om Java te gebruiken	11
Java in perspectief zien	12
Objectgeoriënteerd programmeren	14
Objectgeoriënteerde talen	15
Objecten en hun klassen	15
Wat zijn de voordelen van een objectgeoriënteerde taal?	18
Je kennis van klassen en objecten verfijnen	20
Wat volgt?	22
HOOFDSTUK 2: Alles over software	23
Snel aan de slag	23
Wat je op je computer installeert	25
Wat is een compiler?	26
Wat is een Java Virtual Machine?	29
Software ontwikkelen	35
Wat is een Integrated Development Environment?	36
HOOFDSTUK 3: Algemene bouwstenen gebruiken	39
Javaans leren spreken	39
De grammatica en veelgebruikte namen	40
De woorden in een Java-programma	42
Voor het eerst Java-code bestuderen	43
Een eenvoudig Java-programma begrijpen	45

	De Java-klasse	45
	De Java-methode	46
	De methode main in een programma	48
	De computer eindelijk iets laten doen	49
	Accolades	52
	Tijd voor wat commentaar	54
	Commentaar aan je code toevoegen	55
	Welk excuus heeft Barry?	58
	Commentaar toevoegen om met code te experimenteren	59
DEEL 2:	ZELF JAVA-PROGRAMMA'S SCHRIJVEN ...	61
HOOFDSTUK 4:	Variabelen en hun waarden	63
	Een variabele veranderen	64
	Toekenningsopdrachten	66
	De verschillende soorten waarden van variabelen	67
	Tekst weergeven	71
	Getallen zonder decimalen	71
	Declaratie en initialisatie van variabelen combineren	73
	Atomen: de primitieve typen van Java	74
	Het type char	75
	Het type boolean	77
	Moleculen en samenstellingen: referentietypen	79
	Een importdeclaratie	83
	Nieuwe waarden maken door operatoren toe te passen ..	85
	Eén keer initialiseren, vaak toekennen	89
	De operatoren increment en decrement	89
	Toekenningsoperatoren	93
HOOFDSTUK 5:	Het programmaverloop sturen met keuzeopdrachten	97
	Beslissingen nemen (if-opdrachten)	98
	Raad het getal	98
	Toetsaanslagen verwerken	99
	Willekeur creëren	102
	De opdracht if	103
	Het dubbele isgelijktken	104
	Zet je schrap	105
	If-opdrachten in je code laten inspringen	105
	Else uit Ifrica	106
	Met vergelijkingen en logische operatoren condities samenstellen	108
	Getallen vergelijken; tekens vergelijken	108
	Objecten vergelijken	109
	Alles in één klap importeren	112
	Logische operatoren	112

	Vive les nuls!	115
	(Conditioes tussen haakjes)	116
	Een nest bouwen	118
	Tussen veel alternatieven kiezen (de opdracht switch)	120
	Een eenvoudige switch-opdracht	121
	Wel of geen break	124
	Een verbeterde versie van switch	126
HOOFDSTUK 6:	Het programmaverloop sturen met lussen	129
	Opdrachten voortdurend herhalen (met while-opdrachten)	130
	Een vast aantal keer herhalen (for-opdrachten)	133
	De anatomie van de opdracht for	135
	De wereldpremière van 'Al's All Wet'	136
	Herhalen tot je krijgt wat je wilt (do-opdrachten)	138
	Eén teken lezen	141
	Bestanden verwerken in Java	142
	Variabelendeclaraties en blokken	144
DEEL 3:	HET OVERZICHT BEHOUDEN: OBJECT-GEORIËNTEERD PROGRAMMEREN	145
HOOFDSTUK 7:	In klassen en objecten denken	147
	Een klasse definiëren	148
	Variabelen declareren en objecten maken	150
	Een variabele initialiseren	153
	De velden van een object gebruiken	153
	Eén programma, meerdere klassen	153
	Klassen van het type public	154
	Een methode in een klasse definiëren	155
	Een rekening die zichzelf weergeeft	156
	De header van de methode display	158
	Waarden naar methoden sturen en ervan ontvangen	159
	Een waarde aan een methode meegeven	161
	Een waarde retourneren uit de methode getInterest	163
	Getallen er goed laten uitzien	165
	Verborgen details met accessor-methoden	169
	Goede programmeergewoonten	169
	Public en private: velden ontoegankelijk maken	172
	Regels afdwingen met accessor-methoden	174
HOOFDSTUK 8:	Tijd en geld besparen: bestaande code hergebruiken	175
	Een klasse definiëren	176
	Het laatste nieuws over werknemers	177

	Je klasse aan het werk zetten	178
	Een cheque uitschrijven	180
	Met schijfbestanden werken (een korte omleiding)	181
	Gegevens in een bestand opslaan	182
	Code kopiëren en plakken	182
	Gegevens uit een bestand lezen	184
	Wie heeft mijn bestand verplaatst?	187
	Mapnamen aan je bestandsnamen toevoegen	187
	Eén regel per keer lezen	188
	De verbinding met een schijfbestand sluiten	190
	Subklassen definiëren	191
	Een subklasse maken	193
	Het maken van subklassen kan een gewoonte worden	195
	Subklassen gebruiken	196
	Typen overeen laten komen	198
	De andere helft van het verhaal	199
	Bestaande methoden overschrijven	200
	Een Java-annotatie	202
	Methoden van klassen en subklassen gebruiken	203
HOOFDSTUK 9:	Nieuwe objecten construeren	205
	Constructors definiëren	206
	Wat is een temperatuur?	207
	Wat is een temperatuurschaal? (het type enum)	207
	Goed, maar wat is een temperatuur?	208
	Wat je met een temperatuur kunt doen	210
	Een studie van de aanroep <code>new Temperature(32.0)</code>	213
	Sommige dingen veranderen niet	216
	Meer subklassen	217
	Betere temperaturen bouwen	217
	Constructors voor subklassen	219
	Al deze dingen gebruiken	220
	De standaardconstructor	221
	Een constructor die meer doet	223
	Klassen en methoden uit de Java-API	226
	De annotatie <code>SuppressWarnings</code>	228
DEEL 4:	HANDIGE JAVA-TECHNIEKEN	229
HOOFDSTUK 10:	Variabelen en methoden op de juiste plek zetten	231
	Een klasse definiëren	232
	Een andere manier om getallen te verfraaien	233
	De klasse <code>Player</code> gebruiken	234
	Negen, echt waar, negen	236
	De gebruikersinterface	236

Een exception van methode naar methode gooien . . .	238
Statische zaken	239
Wat is hier zo statisch aan?	241
De statische initialisatie	242
Het complete teamgemiddelde tonen	243
Statisch is niets nieuws	245
Statisch of niet statisch?	246
Experimenten met variabelen	247
Een variabele op zijn plek zetten	248
Een variabele de weg wijzen	251
Parameters doorgeven	254
Als waarde doorgeven	254
Een resultaat retourneren	256
Als referentie meegeven	257
Een object uit een methode retourneren	258
Epiloog	260
HOOFDSTUK 11: In arrays met waarden goochelen	261
Alles op een rijtje zetten	261
In twee stappen een array maken	264
Waarden opslaan	264
Tabstops en andere speciale dingen	267
Een array-initialisator gebruiken	267
Een array doorlopen met een uitgebreide for-lus	268
Zoeken	270
Naar een bestand schrijven	272
Een bestand sluiten	274
Arrays met objecten	276
De klasse Room gebruiken	277
Nog een manier om getallen te verfraaien	281
De conditionele operator	282
Argumenten op de opdrachtregel	282
Opdrachtregelargumenten in een Java-programma gebruiken	284
Het juiste aantal opdrachtregelargumenten controleren	286
HOOFDSTUK 12: Collecties en streams (als arrays niet voldoen)	289
De beperkingen van arrays begrijpen	289
Collectieklassen brengen uitkomst	291
Een arraylijst gebruiken	291
Generics gebruiken	293
Controleren of er meer gegevens zijn	296
Een iterator gebruiken	297
De vele collectieklassen van Java	298
Nieuw in Java 8: functioneel programmeren	299

	Een probleem op de oude manier oplossen	302
	Streams	304
	Lambda-expressies	305
	Een taxonomie van lambda-expressies	308
	Streams en lambda-expressies gebruiken	309
	Waarom zou je?	314
	Methodereferenties	315
HOOFDSTUK 13:	Rekening houden met onverwachte wendingen	319
	Exceptions verwerken	320
	De parameter in een catch-clausule	324
	Soorten exceptions	325
	Wie gaat de exception opvangen?	327
	Een catch-clausule voor meerdere exceptions	334
	Je kunt te voorzichtig zijn	335
	Nuttige dingen doen	336
	Goede exceptions, onze vrienden	337
	Een exception verwerken of doorspelen	338
	De klus afmaken met een finally-clausule	343
	Een try-opdracht met bronnen	346
HOOFDSTUK 14:	Namen delen met andere delen van een programma	349
	Access-modifiers	350
	Klassen, toegang en programma's met meerdere delen	351
	Leden versus klassen	351
	Access-modifiers voor leden	352
	Een tekening op een frame zetten	354
	Mapstructuur	357
	Een frame maken	358
	Van de oorspronkelijke code wegsluipen	360
	Standaardtoegang	361
	Terugkruipen in het pakket	364
	Beschermde toegang	365
	Niet-subklassen in hetzelfde pakket zetten	367
	Access-modifiers voor Java-klassen	368
	Klassen van het type public	369
	Klassen die niet van het type public zijn	369
HOOFDSTUK 15:	Op toetsaanslagen en muisklikken reageren	371
	Toe dan... klik op de knop	372
	Events en event-handlers	374
	De Java-interface	375
	De verschillende threads tijdens het uitvoeren van code	376

	Het sleutelwoord this	378
	In de methode actionPerformed	379
	De serialVersionUID	379
	Op andere dingen dan muisklikken reageren	381
	Binnenklassen maken	387
HOOFDSTUK 16:	Java-applets schrijven	391
	Applets voor beginners	391
	Wachten tot je geroepen wordt	393
	Een klasse van het type public	393
	De Java-API (alweer)	394
	Dingen in beweging zetten	395
	De methoden in een applet	397
	Wat stop je in al die methoden?	398
	In een applet op events reageren	399
HOOFDSTUK 17:	Met databases werken in Java	403
	JDBC en Java DB	403
	Gegevens maken	404
	SQL-opdrachten gebruiken	406
	Verbinding maken en weer verbreken	407
	Gegevens ophalen	409
DEEL 5:	HET DEEL VAN DE TIENTALLEN	413
HOOFDSTUK 18:	Tien manieren om fouten te vermijden ..	415
	Hoofdletters op de juiste plek zetten	415
	Uit een switch-opdracht breken	416
	Waarden vergelijken met een dubbel isgelijktken	416
	Componenten aan een GUI toevoegen	417
	Listeners toevoegen om events te verwerken	417
	De vereiste constructors definiëren	417
	Niet-statische referenties gebruiken	418
	Binnen de grenzen van een array blijven	418
	Null-pointers anticiperen	419
	Java helpen zijn bestanden te vinden	420
HOOFDSTUK 19:	Tien websites over Java	421
	De website bij dit boek	421
	Dicht bij de bron	422
	Nieuws, recensies en voorbeeldcode	422
	Werken met Java	422
	De favorieten van bijna iedereen	422
INDEX		423

Inleiding

Java is prachtig; ik gebruik het al jaren. Ik doe dit omdat Java erg goed is geordend. Bijna alles volgt eenvoudige regels. Die regels kunnen soms intimiderend zijn, maar met dit boek heb je al snel door hoe ze werken. Dus als je van plan bent Java te gebruiken en op zoek bent naar een alternatief voor het traditionele technische boek, dan raad ik je aan te gaan zitten en rustig te beginnen met het lezen van *Programmeren met Java voor Dummies*.

Over dit boek

Ik zou graag zeggen, ‘Open dit boek op een willekeurige pagina en begin met het schrijven van Java-code. Voer de benodigde gegevens in en ga gewoon verder.’ Dit kan, tot op zekere hoogte. Je kunt niets kapot maken door Java-code te schrijven, dus je kunt er naar hartenlust op los experimenteren.

Maar je moet wel realistisch blijven. Als je de grote lijnen uit het oog verliest, is het schrijven van een programma erg lastig. Dit geldt voor elke programmeertaal, dus niet alleen voor Java. Schrijf je code zonder dat je precies weet hoe die werkt, dan loop je onherroepelijk vast op het moment dat de code niet direct precies doet wat je ervan verwacht.

Dit is dan ook de reden dat ik het programmeren in Java in dit boek in beheersbare stukken heb verdeeld. Elk stuk beslaat (ongeveer) een hoofdstuk. Je mag overal beginnen met lezen: hoofdstuk 5, hoofdstuk 10 of waar je maar wilt. Je kunt zelfs halverwege een hoofdstuk binnenvallen. Ik heb geprobeerd interessante voorbeelden te gebruiken zonder een hoofdstuk afhankelijk te maken van andere hoofdstukken. Als ik een belangrijk idee uit een ander hoofdstuk gebruik, laat ik je dit weten, zodat je snel weer verder kunt.

Over het algemeen raad ik je het volgende aan:

- » Doe geen moeite over iets te lezen wat je al weet.
- » Wees niet bang om stukken over te slaan als je nieuwsgierig bent. Je kunt altijd nog snel terugbladeren naar een eerder hoofdstuk op het moment dat dit nodig is.

De gebruikte conventies in dit boek

Bijna elk technisch boek begint met een kleine typografische uitleg; *Programmeren met Java voor Dummies*, vormt hierop geen uitzondering. In dit boek worden de volgende typografische aanduidingen gebruikt:

- » Nieuwe termen worden *cursief* geschreven.
- » Als je iets moet invoeren wat tussen de gewone tekst staat, dan heb ik die tekens vet gemaakt. Bijvoorbeeld: 'Typ **MijnNieuweProject** in het invoervak.'
- » Ik gebruik een *afwijkend lettertype* voor Java-code, bestandsnamen, adressen van webpagina's (URL's), schermberichten en andere zaken. Als je iets moet invoeren wat erg lang is, staat het ook in een *afwijkend lettertype* op een aparte regel (of op aparte regels).
- » Sommige dingen die je zelf invoert moet je veranderen. Ik vraag je bijvoorbeeld om het volgende in te voeren:

```
public class Eennaam
```

Dit betekent dat je eerst **public class** invoert en dan een naam typt die je zelf hebt verzonnen. Woorden die je met je eigen tekst moet vervangen, worden weergegeven in een *afwijkend cursief lettertype*.

Wat je niet hoeft te lezen

Kies het eerste hoofdstuk of de eerste paragraaf met onbekend materiaal en begin daar te lezen. Misschien heb je er net zo'n hekel aan als ik om beslissingen te nemen. In dat geval kun je gebruikmaken van de volgende richtlijnen:

- » Weet je al wat voor soort dier Java is en heb je besloten dat je ervan gebruik wilt maken, sla hoofdstuk 1 dan over en begin met hoofdstuk 2. Ik vind het echt niet vervelend.
- » Weet je al hoe je een Java-programma aan de praat krijgt en interesseert het je niet wat er tijdens de uitvoering van een Java-programma achter de schermen gebeurt, dan sla je hoofdstuk 2 over en begin je met hoofdstuk 3.
- » Begin met de hoofdstukken 2 en 3 als je beroepsmatig in een andere taal dan C of C++ programmeert. De hoofdstukken 5 en 6 vind je

waarschijnlijk eenvoudig, maar bij hoofdstuk 7 moet je je aandacht er weer bijhouden.

- » Schrijf je beroepsmatig C-programma's (niet C++), dan begin je met de hoofdstukken 2, 3 en 4 en kun je de hoofdstukken 5 en 6 snel doornemen.
- » Schrijf je C++-programma's dan neem je de hoofdstukken 2 en 3 snel door, blader je door de hoofdstukken 4 tot en met 6 en begin je serieus te lezen in hoofdstuk 7. Java gaat iets anders met klassen en objecten om dan C++.
- » Schrijf je beroepsmatig Java-programma's, dan mag je bij me langskomen om me te helpen bij het schrijven van de volgende editie van dit boek.

Sla gerust de kaderteksten en paragrafen met het pictogram Technische info over als je die niet wilt lezen. Je mag trouwens alles overslaan wat je niet wilt lezen.

Wie ben jij?

In dit boek doe ik enkele aannamen over jou, de lezer. Klopt een van deze aannamen niet, dan is er nog geen man overboord. Maar als al deze aannamen niet kloppen... nou, koop dan toch maar gewoon dit boek.

- » **Ik ga ervan uit dat je de beschikking hebt over een computer.** Het goede nieuws is dat je de code in dit boek op bijna elke computer kunt uitvoeren. De enige computers die je niet kunt gebruiken, zijn exemplaren die meer dan tien jaar oud zijn (ongeveer).
- » **Ik ga ervan uit dat je de algemene menu's en dialoogvensters van je computer kent.** Je hoeft geen kenner te zijn van Windows, Linux of Macintosh, maar je moet wel in staat zijn een programma te starten, een bestand terug te vinden, een bestand in een specifieke map te kunnen zetten... dat soort dingen. Tijdens het oefenen van het materiaal in dit boek typ je meestal code op je toetsenbord in plaats van dat je de muis gebruikt.

De weinige keren dat je moet slepen en neerzetten, knippen en plakken of het RAM-geheugen moet uitbreiden (niet schrikken, dat was een grapje), loods ik je zorgvuldig langs de benodigde stappen. Maar omdat je computer op een miljard verschillende manieren kan zijn geconfigureerd, zullen mijn instructies niet altijd precies passen bij jouw situatie. Dus als je een platformspecifieke taak tegenkomt, probeer je eerst de stappen in dit boek te volgen. Passen die niet bij jouw situatie, raadpleeg dan een boek met instructies die op jouw systeem zijn toegespitst.

- » **Ik ga ervan uit dat je logisch kunt nadenken.** Meer komt er niet bij kijken tijdens het programmeren in Java. Denk je op een logische wijze, dan

lukt programmeren ook. Maar lees verder als je gelooft dat je niet logisch kunt denken. Waarschijnlijk wacht je een aangename verrassing.

» **Ik doe bijna geen aannamen over je ervaring met programmeren (of over het ontbreken van programmeerervaring).** Tijdens het schrijven van dit boek heb ik geprobeerd het onmogelijke te bereiken. Ik heb geprobeerd het boek interessant te maken voor ervaren programmeurs, maar het toch toegankelijk te houden voor lezers met weinig of geen programmeerervaring. Ik ga er dus niet vanuit dat je een achtergrond in programmeren hebt. Het is geen enkel probleem als je nog nooit een lus hebt gemaakt of een array hebt geïndexeerd.

Maar als je deze zaken ooit wel hebt gedaan (misschien in Visual Basic, COBOL of C++), dan zul je enkele interessante afwijkingen in Java ontdekken. De ontwikkelaars van Java hebben de beste ideeën van objectgeoriënteerd programmeren gestroomlijnd, omgewerkt en opnieuw geordend in een fraaie, krachtige manier om problemen aan te pakken. Java zit vol nieuwe, uitdagende voorzieningen. Veel van deze voorzieningen komen vanzelfsprekend over als je ze leert kennen. Je zult dan ook helemaal geen probleem hebben om je draai te vinden in Java.

De indeling van dit boek

Dit boek is opgedeeld in alinea's, die gegroepeerd zijn in paragrafen, die deel uitmaken van hoofdstukken, die op hun beurt zijn samengevoegd in vijf delen. Tijdens het schrijven van een boek leer je de structuur ervan behoorlijk goed kennen. Na maanden te hebben geschreven, begin je 's nachts te dromen in paragrafen en hoofdstukken. Hier vind je een overzicht van de delen.

Deel 1: Aan de slag met Java

Dit deel kun je beschouwen als een uitgebreide samenvatting van Java. Je vind er materiaal van het type 'Wat is Java?' en een hoofdstuk met een snelle start; hoofdstuk 3. In hoofdstuk 3 maak je ook kennis met belangrijke technische ideeën en ga je een eenvoudig programma ontleden.

Deel 2: Zelf Java-programma's schrijven

De basisprincipes worden in hoofdstukken 4 tot en met 6 behandeld. In deze hoofdstukken lees je de dingen die je moet weten om je computer aan het werk te houden.

Heb je weleens programma's in Visual Basic, C++ of een andere programmeertaal geschreven, dan zullen sommige dingen in dit deel je bekend voorkomen. In zo'n geval sla je sommige paragrafen over of neem je ze snel door. Maar lees niet te snel, want Java verschilt van andere programmeertalen, vooral in de onderwerpen die in hoofdstuk 4 aan bod komen.

Deel 3: Het overzicht behouden: objectgeoriënteerd programmeren

In dit deel staan enkele van mijn favoriete hoofdstukken. Dit deel behandelt het zeer belangrijke onderwerp objectgeoriënteerd programmeren. Je leest hoe je oplossingen in kaart brengt voor grote problemen. De voorbeelden in deze hoofdstukken zijn niet groot, maar de achterliggende ideeën wel. In overzichtelijke stappen ontdek je hoe je klassen ontwerpt en objecten samenstelt. Heb je weleens een boek gelezen waarin objectgeoriënteerd programmeren in vage, algemene termen wordt beschreven? Ik kan je trots melden dat *Programmeren met Java voor Dummies* dit niet doet. In dit boek geef ik van elk concept een eenvoudig, maar concreet voorbeeld in programmavorm.

Deel 4: Handige Java-technieken

Heb je de smaak eenmaal te pakken, dan vind je in dit deel van het boek wat je over Java nodig hebt. De hoofdstukken in dit deel gaan over de details – de dingen die je niet te zien krijgt als je alleen oppervlakkig naar het materiaal kijkt. Dus nadat je de eerste delen hebt gelezen en zelf enkele programma's hebt geschreven, kun je met dit deel dieper op de materie ingaan.

Deel 5: Het deel van de tientallen

Het deel van de tientallen is de snoepwinkel van Java. In dit deel vind je traditiegetrouw lijstjes: lijstjes met tips over het vermijden van fouten, lijstjes met goede bronnen voor meer informatie en lijstjes met handige hulpmiddelen.

De pictogrammen in dit boek

Als je naar me had gekeken terwijl ik dit boek schreef, zou je me achter mijn computer hebben zien zitten terwijl ik in mezelf aan het praten was. Ik zeg elke zin in mijn hoofd op en soms mompel ik ze verschillende keren

voordat ik iets noteer. Kom ik op een extra opmerking, een gedachtesprong of iets anders wat niet in de lopende tekst thuishoort, dan draai ik mijn hoofd een beetje opzij. Op die manier weet degene die naar me luistert (meestal is dit niemand) dat ik een tijdelijk zijpad heb betreden.

Maar in dit boek zie je het natuurlijk niet als ik mijn hoofd wegdraai, dus heb ik een andere manier nodig om een losse gedachte in een apart hoekje te zetten. Dit doe ik met pictogrammen. Zie je het pictogram Tip of Belangrijk, dan weet je dat ik even een uitstapje maak.

In dit boek kom je de volgende pictogrammen tegen:



Een tip bestaat uit extra informatie, iets nuttigs wat andere schrijvers je misschien vergeten zijn te melden.



Iedereen maakt fouten. Je wilt niet weten hoeveel ik er op mijn geweten heb. Dus als ik de kans groot acht dat iets tot een fout kan leiden, markeer ik het met dit pictogram.



Vraag: wat is sterker dan een Tip, maar niet zo sterk als Pas op?

Antwoord: het pictogram Belangrijk.



Soms moet ik iets technisch met je delen, daar is weinig tegen te beginnen. Zulke informatie kan je helpen te begrijpen wat de mensen achter de schermen (die Java hebben ontworpen) in gedachten hadden. Je hoeft deze teksten niet te lezen, maar misschien vind je ze wel nuttig. Bovendien kunnen ze van pas komen als je van plan bent ook nog andere (technischere) boeken over Java te lezen.

Wat nu?

Ben je eenmaal hier aanbeland, dan ben je klaar om over het programmeren in Java te lezen. Beschouw mij (de auteur) als je gids, gastheer en persoonlijke assistent. Ik doe alles wat in mijn vermogen ligt om dingen interessant te houden en, belangrijker, om zaken goed uit te leggen.

Bevalt dit boek je, stuur me dan een e-mailtje. Mijn e-mailadres, dat ik speciaal voor vragen en opmerkingen over dit boek maakte, is `JavaForDummies@allmycode.com`. Ik ben trouwens ook beschikbaar op Twitter (`@allmycode`) en op Facebook (`www.facebook.com/allmycode`).