

Spis treści

Wstęp	XIII
Wszystko, czego potrzebujesz, to Java	1
<i>Anders Noras</i>	
Testowanie zatwierdzające	3
<i>Emily Bache</i>	
Rozszerzenie Javadoc przez AsciiDoc	5
<i>James Elliott</i>	
Uważaj na otoczenie swojego kontenera	7
<i>David Delabasse</i>	
Zachowanie jest „łatwe”, a stan jest trudny	9
<i>Edson Yanaga</i>	
Benchmarking jest trudny – pomaga w nim JMH	11
<i>Michael Hunger</i>	
Korzyści z kodyfikowania i zapewniania jakości architektonicznej	14
<i>Daniel Bryant</i>	
Podziel problemy i zadania na małe kawałki	17
<i>Jeanne Boyarsky</i>	
Buduj różnicowane zespoły	19
<i>Ixchel Ruiz</i>	
Procesy budowania nie muszą być powolne ani zawodne	21
<i>Jenn Strater</i>	
„Ale u mnie to działa!”	23
<i>Benjamin Muschko</i>	
Argumenty przeciw fat JAR-om	25
<i>Daniel Bryant</i>	

Restaurator kodu <i>Abraham Marin-Perez</i>	27
Współbieżność w JVM <i>Mario Fusco</i>	29
CountDownLatch – przyjaciel czy wróg? <i>Alexey Soshin</i>	31
Wyrażenia deklaratywne drogą do równoległości <i>Russel Winder</i>	33
Dostarczaj szybciej lepsze oprogramowanie <i>Burk Hufnagel</i>	35
Czy wiesz, która jest godzina? <i>Christin Gorman</i>	37
Nie ukrywaj swoich narzędzi <i>Gail Ollis</i>	39
Nie zmieniaj swoich zmiennych <i>Steve Freeman</i>	41
Przyjmij myślenie z SQL-a <i>Dean Wampler</i>	44
Zdarzenia między komponentami Javy <i>A. Mahdy Abdel Aziz</i>	46
Pętle informacji zwrotnych <i>Liz Keogh</i>	48
Praca na najwyższych obrotach <i>Michael Hunger</i>	50
Postępuj zgodnie z nudnymi standardami <i>Adam Bien</i>	52
Częste wydania zmniejszają ryzyko <i>Chris O'Dell</i>	54
Od łamigłówek do produktów <i>Jessica Kerr</i>	56
„Full-stack developer” to sposób myślenia <i>Maciej Walkowiak</i>	58
Odśmiecanie pamięci jest twoim przyjacielem <i>Holly Cummins</i>	60

Popraw nazywanie rzeczy <i>Peter Hilton</i>	62
Hej Fred, możesz mi przekazać HashMap? <i>Kirk Pepperdine</i>	64
Jak unikać null <i>Carlos Obregón</i>	66
Jak spowodować awarię swojej JVM <i>Thomas Ronzon</i>	68
Poprawa powtarzalności i audytowalności dzięki ciągniętemu dostarczaniu <i>Billy Korando</i>	70
W wojnach językowych Java nie daje za wygraną <i>Jennifer Reif</i>	72
Myślenie wkomponowane <i>Patricia Aas</i>	74
Interoperacyjność z Kotlinem <i>Sebastiano Poggi</i>	76
Jest ukończone, ale... <i>Jeanne Boyarsky</i>	78
Certyfikacja w Javie: kamień probierczy technologii <i>Mala Gupta</i>	80
Java to dziecko lat dziewięćdziesiątych <i>Ben Evans</i>	82
Programowanie w Javie z perspektywy wydajności JVM <i>Monica Beckwith</i>	84
Java powinna być przyjemna <i>Holly Cummins</i>	86
Nieoznaczone typy Javy <i>Ben Evans</i>	88
JVM jest platformą wieloparadygmatową: wykorzystaj to, aby ulepszyć swoje programowanie <i>Russel Winder</i>	90
Trzymaj rękę na pulsie <i>Trisha Gee</i>	92

Rodzaje komentarzy	94
<i>Nicolai Parlog</i>	
Poznaj twą flatMap	96
<i>Daniel Hinojosa</i>	
Poznaj swoje kolekcje	99
<i>Nikhil Nanivadekar</i>	
Kotlin to jest coś	101
<i>Mike Dunn</i>	
Poznaj idiomy Javy i umieść je w pamięci podręcznej swojego umysłu	104
<i>Jeanne Boyarsky</i>	
Nauka kata i kata do nauki	106
<i>Donald Raab</i>	
Naucz się kochać swój stary kod	109
<i>Uberto Barbini</i>	
Naucz się korzystać z nowych funkcji Javy	111
<i>Gail C. Anderson</i>	
Naucz się swojego IDE, aby zmniejszać obciążenie poznawcze	114
<i>Trisha Gee</i>	
Zawrzyjmy umowę: sztuka projektowania API w Javie	116
<i>Mario Fusco</i>	
Uczyń kod prostym i czytelnym	118
<i>Emily Jiang</i>	
Uczyń swoją Javę jeszcze bardziej odlatową	120
<i>Ken Kousen</i>	
Minimalne konstruktory	123
<i>Steve Freeman</i>	
Nazwij datę	126
<i>Kevlin Henney</i>	
Potrzeba stosowania technologii przemysłowych	128
<i>Paul W. Homer</i>	
Zbuduj tylko te fragmenty, które się zmieniają, a resztę wykorzystuj ponownie	130
<i>Jenn Strater</i>	

Projekty open source nie są magiczne	132
<i>Jenn Strater</i>	
Optional to monada łamiąca reguły, ale i dobry typ	134
<i>Nicolai Parlog</i>	
Pakiet na funkcjonalność z domyślnym modyfikatorem dostępu	137
<i>Marco Beelen</i>	
Produkcja to najszczęśliwsze miejsce na Ziemi	139
<i>Josh Long</i>	
Program z dobrymi testami jednostkowymi	141
<i>Kevlin Henney</i>	
Czytaj OpenJDK codziennie	143
<i>Heinz M. Kabutz</i>	
Warto zajrzeć pod maskę	145
<i>Rafael Benevides</i>	
Odrodzenie Javy	147
<i>Sander Mak</i>	
Odkryj JVM na nowo dzięki Clojure	149
<i>James Elliott</i>	
Refaktoryzuj wartości logiczne na wyliczenia	151
<i>Peter Hilton</i>	
Refaktoryzacja w kierunku szybkiego czytania	153
<i>Benjamin Muskalla</i>	
Proste obiekty wartości	155
<i>Steve Freeman</i>	
Dbaj o swoje deklaracje modułów	158
<i>Nicolai Parlog</i>	
Dbaj o swoje zależności	160
<i>Brian Vermeer</i>	
Potraktuj poważnie „separację zagadnień”	162
<i>Dave Farley</i>	
Techniczne rozmowy kwalifikacyjne to umiejętność, którą warto rozwijać	164
<i>Trisha Gee</i>	

Programowanie sterowane testami	166
<i>Dave Farley</i>	
W twoim katalogu bin/ są świetne narzędzia	168
<i>Rod Hilton</i>	
Wyjdź poza piaskownicę Javy	170
<i>Ian F. Darwin</i>	
Myślenie w koprocedurach	172
<i>Dawn Griffiths i David Griffiths</i>	
Wątki to infrastruktura i tak je traktuj	174
<i>Russel Winder</i>	
Trzy cechy naprawdę, naprawdę dobrych programistów	176
<i>Jannah Patchay</i>	
Kompromisy w architekturze mikroserwisowej	178
<i>Kenny Bastani</i>	
Nie kontroluj swoich wyjątków	180
<i>Kevlin Henney</i>	
Odblokowywanie ukrytego potencjału testowania integracyjnego przy użyciu kontenerów	182
<i>Kevin Witte</i>	
Nieuzasadniona skuteczność fuzz testingu	184
<i>Nat Pryce</i>	
Użyj pokrycia, aby ulepszyć swoje testy jednostkowe	187
<i>Emily Bache</i>	
Swobodnie używaj niestandardowych adnotacji tożsamościowych	189
<i>Mark Richards</i>	
Użyj testowania, aby szybciej tworzyć lepsze oprogramowanie	192
<i>Marit van Dijk</i>	
Używanie zasad obiektowych w kodzie testów	194
<i>Angie Jones</i>	
Wykorzystywanie mocy społeczności do rozwijania swojej kariery	197
<i>Sam Hepburn</i>	

Czym jest program JCP i jak w nim uczestniczyć <i>Heather VanCura</i>	199
Dlaczego nie traktuję certyfikatów poważnie <i>Colin Vipurs</i>	201
Pisz jednozdaniowe komentarze <i>Peter Hilton</i>	203
Napisz „czytelny kod” <i>Dave Farley</i>	205
Młoda, stara i odśmiecacz <i>Maria Arias de Reyna</i>	207
Indeks	209
O autorach i redaktorach	218

Co powinien wiedzieć każdy programista Javy? To zależy. Zależy od tego, kogo pytasz i kiedy pytasz. Sugestia jest co najmniej tyle, ile punktów widzenia. W zakresie języka, platformy, ekosystemu i społeczności, która wpływa na sposób myślenia ludzi. I dzieje się tak od stułecia do stułecia. Jeden do wielu rdzeni, od megabajtów do gigabajtów. Jest to szersze, niż można byłoby się spodziewać, że zostanie ujęte w jednej książce przez jednego autora.

Natomiast w tej książce wykorzystamy niektóre z tych wielu punktów widzenia, aby stworzyć razem pewien przekrój i przedstawić sposób myślenia w technologii Java. To nie będzie *jedynym* słusznym podejściem, ale 97 spojrzeńami 73 autorów. Cytując przedmowę książki *97 Things Every Programmer Should Know* (O'Reilly):

jest tak wiele rzeczy do poznania i do zrobienia i tak wiele na to sposobów, że żadna osoba ani żaden byt nie może rościć sobie praw do „jedynnej słusznej drogi”. Poszczególne spojrzenia nie muszą zachęcać się jak części pewnego modułu – może być nawet odartowanie. Wartość każdego spojrzenia wynika z jego odrębności. Wartość kolekcji wynika z tego, jak poszczególne elementy uzupełniają się, potwierdzają, a nawet zaprzeczają sobie nawzajem. Nie ma nadrzędnej paradygmy: Ty poszukujesz odpowiedzi, zastanawiasz się i łączysz to, co czytasz, porównując z własnym kontekstem, wiedzą i doświadczeniem.

Co powinien wiedzieć każdy programista Javy? 97 rzeczy, które będziemy poruszać, obejmują język, JVM, techniki testowania, IDE, społeczność, historię, zwmine myślenie, wiedzę wdrożeniową, profesjonalizm, styl, treść, paradygmaty programowania, programistów jako ludzi, architekturę oprogramowania, umiejętności wykraczające poza kod, narzędzia, mechanikę GC, języki JVM inne niż Java... I nie tylko.

Własność intelektualna

Zgodnie z duchem pierwszej książki *97 Things*, każde spojrzenie na pewne zagadnienie w tej publikacji jest zgodne z nieograniczonym modelem open source. Każde jest