

Spis treści

Wprowadzenie	xi
Podziękowania	xiii

Część I

Paradygmat funkcyjny

1. Programowanie funkcyjne wraca do łask	3
Widzieliśmy już ten film	3
Nowe argumenty za programowaniem funkcyjnym	5
2. Myślenie funkcyjne dla imperatywnego umysłu	7
Wszystko sprowadza się do funkcji	7
Chodzi o niemutowalność	8
Chodzi o sposób myślenia	9

Część II

Scala – język hybrydowy

3. Scala i styl funkcyjny	15
Czystość funkcyjna	15
Funkcje wyższego rzędu	16
Prosty przykład	16
Przykład praktyczny	18
Co ze znajdowaniem błędów i wydajnością?	21
4. Praca z kolekcjami języka Scala	23
Kolekcje niemutowalne	23
Kolekcje zmienne (mutowalne)	27
Kolekcje leniwe	27

5. Tworzenie funkcji wyższego rzędu w języku Scala	31
Tworzenie funkcji wyższego rzędu	31
Wiele list parametrów	34
Wartości funkcji i wzorzec pożyczki	35

Część III

Clojure – nowy Lisp

6. Wywiad z Richem Hickeyem	41
Dlaczego Clojure?	41
Infrastruktura	42
Z czym go porównać?	43
7. Zrozumienie języka Clojure – dlaczego Lisp nadal się liczy	45
REPL	46
Wektory i słowa kluczowe	47
Makra	49
8. Tożsamość, wartość i stan w Clojure	53
Model obiektowy	53
Model Clojure	56
9. Programowanie współbieżne w Clojure	61
Prosty problem programowania współbieżnego	61
Rozwiązania w Clojure	63

Część IV

Elixir – aby programowanie znów było przyjemnością

10. Wzorce i przekształcenia w języku Elixir	71
Dopasowywanie do wzorców	72
Dopasowywanie do wzorców dla danych strukturalnych	73
Dopasowywanie do wzorców i funkcje	74
Przekształcenie to zadanie numer 1	75
11. Stawanie się funkcyjnym za pomocą Elixir	77
Funkcje anonimowe	77
Funkcje nazwane	78
Przykład praktyczny	80
Refaktoryzacja do stylu funkcyjnego	82
Co wyróżnia ten kod	85

12. Równoległość w języku Elixir	87
Model aktora	87
Aktory a Elixir	88
Komunikaty	89
Monitorowanie naszego procesu	93
Ostatni przykład	94
Współbieżność to istota Elixir	96

Część V

Haskell – plac ćwiczeń dla uczonych

13. Haskell i myślenie funkcyjne	99
O co w tym wszystkim chodzi	99
Szybkie ćwiczenie	100
Typy danych są niedrogie	101
Dopasowywanie do wzorców	104
Rekurencja, sterowanie i funkcje wyższego rzędu	105
Inne własności	107
14. Haskell w praktyce	113
Po jednym kroku	114
Generowanie kandydatów	117
Filtrowanie słownikowe	120
Wyszukiwanie wszcz	121
Użycie wyszukiwania	124
Wydajność i optymalizacja	125

Część VI

Swift – programowanie funkcyjne dla aplikacji mobilnych

15. Swift – co powinniśmy wiedzieć	129
Hello, Swift!	130
Funkcyjny Swift	131
16. Myślenie funkcyjne w języku Swift	137
Nie używamy nil, chyba że celowo	137
Unikanie stanu mutowalnego	139
Należy używać funkcji wyższego rzędu	140

Część VII

Idziemy głębiej

17. Protokoły – Swift kontra Ruby i Elixir	147
Problem z rozszerzeniami	148
Przypadek dla protokołów	149
Protokoły i rozszerzenia	151
18. Dopasowywanie do wzorca w Scali	153
Liczenie monet	153
Dopasowywanie wszystkich rzeczy	156
Korzystanie z wyodrębniania	157
19. Współbieżność w Scali	161
Korzystanie z kolekcji równoległych	161
Wiedza, kiedy użyć współbieżności	163
Powrót do wcześniejszego przykładu	165
20. Wyjątkowa obsługa wyjątków w Clojure	167
Prosty przykład	167
Problem z wyjątkami	168
Rozwiązanie – warunki	169
Ułatwmy życie dementom wywołującym	170
Lenistwo i błędy	171
21. Testowanie platformy dla Elixira	173
Inwestowanie w testowanie	173
Jeden eksperyment, kilka miar	174
Optymalizowanie konfiguracji za pomocą TrueStory	175
Zagęszczanie i łączenie miar	176
Kontrolowanie powtarzania konfiguracji z zagnieżdżonymi kontekstami	178
Kontrolowanie powtarzania konfiguracji za pomocą potoków historyjek	179
22. Tworzenie danych testowych w języku Elixir	183
Typowe podejścia	183
Piękne dane dla pięknych testów	184
Rejestrowanie szablonów i prototypów za pomocą Forge	185
Tworzenie instancji wpisów szablonów	186

Wzajemne atrybuty i having	186
Tworzenie struktur	187
Tworzenie niestandardowych jednostek	187
Przystosowywanie zapisywania	188
23. System typów w języku Haskell	191
TL;DR (<i>Too long; didn't read</i> – zbyt długie, nie dało się odczytać)	191
Do czego służą typy?	192
Konkretny przykład – sortowanie	193
Język systemu typów Haskell	195
Wnioskowanie i sprawdzanie typów	196
Kilka przykładów	197
Wygodna przerwa	201
Interfejsy i klasy typów	202
Rzeczywiste przykłady z interfejsami	206
Zalety i wady – reguła 80/20	208
Po Haskellu – typy zależne	209
Twierdzenia są typami, a dowody programami	211
Inne spojrzenie na sortowanie	212
Wracamy na ziemię	214
24. Projekt w Haskellu – testowanie kodu natywnego	217
Nasz kod natywny	218
Krótkie wprowadzenie do FFI Haskell	219
Opakowywanie naszego natywnego kodu w Haskellu	220
Eksperymentowanie z GHCi	220
Krótkie wprowadzenie do QuickCheck	221
Pisanie własności równoważności	221
Likwidacja usterek	224
25. Wiele twarzy funkcji Swifta	227
Anatomia funkcji Swifta	228
Wywoływanie wszystkich funkcji	228
Wywoływanie metod	229
Metody instancji są funkcjami rozwiniętymi	230
Init – uwaga specjalna	231
Wyszukane parametry	233
Kontrola dostępu	238
Wyszukane typy zwrotne	239
Funkcje zagnieżdżone	242

26. Funkcyjne podejście do Lua.	245
Funkcje pierwszoklasowe w Lua	245
Rekurencja w Lua	247
Budowanie za pomocą pierwotnych elementów funkcyjnych	248
Prosta animacja w grze	249
O autorach	253
Bibliografia	257
Indeks	259