

Spis treści

Wprowadzenie — bądź pragmatyczny	7
Część I. Dobry kod	17
Rozdział 1. Bezpieczeństwo	19
Temat 1. Ograniczaj modyfikowalność	20
Temat 2. Minimalizuj zasięg zmiennych	35
Temat 3. Jak najszybciej wyeliminuj typy z zewnętrznych platform	40
Temat 4. Nie udostępniaj wynioskowanych typów	46
Temat 5. Określaj oczekiwania co do argumentów i stanu	48
Temat 6. Preferuj standardowe błędy zamiast niestandardowych	56
Temat 7. Preferuj wyniki null lub Failure, gdy możliwy jest brak wyników	58
Temat 8. Dbaj o odpowiednią obsługę wartości null	61
Temat 9. Zamykaj zasoby za pomocą funkcji use	70
Temat 10. Pisz testy jednostkowe	72
Rozdział 2. Czytelność	75
Temat 11. Projektuj z myślą o czytelności	77
Temat 12. Znaczenie operatora powinno być zgodne z jego nazwą	82
Temat 13. Unikaj zwracania wartości Unit? lub operowania nimi	86
Temat 14. Podawaj typ zmiennej, gdy nie jest on oczywisty	88
Temat 15. Rozważ bezpośrednie podawanie odbiorców	89
Temat 16. Właściwości powinny reprezentować stan, a nie działanie	95
Temat 17. Rozważ stosowanie nazw dla argumentów	99
Temat 18. Przestrzegaj konwencji programistycznych	104

Część II. Projektowanie kodu	107
Rozdział 3. Wielokrotne używanie kodu	109
Temat 19. Nie powtarzaj wiedzy	111
Temat 20. Nie powtarzaj wspólnych algorytmów	118
Temat 21. Stosuj delegaty właściwości do wyodrębniania powtarzających się wzorców dotyczących właściwości	122
Temat 22. Używaj typów generycznych, gdy implementujesz powtarzające się algorytmy	127
Temat 23. Unikaj zakrywania parametrów określających typ	130
Temat 24. Rozważ wariację w typach generycznych	132
Temat 25. Wielokrotne używanie kodu na różnych platformach dzięki wyodrębnianiu wspólnych modułów	142
Rozdział 4. Projektowanie abstrakcji	147
Temat 26. Każdą funkcję pisz na jednym poziomie abstrakcji	151
Temat 27. Korzystaj z abstrakcji do ochrony kodu przed zmianami	157
Temat 28. Określaj stabilność API	169
Temat 29. Rozważ tworzenie nakładek na zewnętrzne API	173
Temat 30. Minimalizuj widoczność elementów	174
Temat 31. Definiuj kontrakty w dokumentacji	178
Temat 32. Przestrzegaj kontraktów abstrakcji	188
Rozdział 5. Tworzenie obiektów	191
Temat 33. Rozważ stosowanie funkcji fabrykujących zamiast konstruktorów	192
Temat 34. Rozważ tworzenie konstruktorów podstawowych z nazwanymi argumentami opcjonalnymi	204
Temat 35. Rozważ definiowanie języków dziedzicznych do tworzenia złożonych obiektów	212
Rozdział 6. Projektowanie klas	221
Temat 36. Preferuj kompozycję zamiast dziedziczenia	222
Temat 37. Stosuj modyfikator data, aby reprezentować zestaw danych	233
Temat 38. Do przekazywania operacji i akcji używaj typów funkcyjnych zamiast interfejsów	239
Temat 39. Preferuj hierarchie klas zamiast klas z trybami	242
Temat 40. Przestrzegaj kontraktu metody equals	247
Temat 41. Przestrzegaj kontraktu metody hashCode	258
Temat 42. Przestrzegaj kontraktu metody compareTo	265
Temat 43. Rozważ przeniesienie mniej istotnych fragmentów API do rozszerzeń	268
Temat 44. Unikaj tworzenia rozszerzeń jako składowych	272

Część III. Wydajność	275
Rozdział 7. Dbanie o niskie koszty	277
Temat 45. Unikaj niepotrzebnego tworzenia obiektów	278
Temat 46. Stosuj modyfikator inline dla funkcji z parametrami o typach funkcyjnych	289
Temat 47. Rozważ stosowanie klas z modyfikatorem inline	301
Temat 48. Usuwać referencje do nieużywanych obiektów	308
Rozdział 8. Wydajne przetwarzanie kolekcji	315
Temat 49. Preferuj sekwencje dla dużych kolekcji z więcej niż jednym etapem przetwarzania	318
Temat 50. Ograniczaj liczbę operacji	331
Temat 51. Rozważ stosowanie tablic z elementami typu podstawowego w kodzie krytycznym ze względu na wydajność	333
Temat 52. Rozważ używanie modyfikowalnych kolekcji	336
Słowniczek	339