

Spis treści

Przedmowa 11

Wstęp 13

Podziękowania 15

O książce 17

O autorach 21

CZĘŚĆ I. WPROWADZENIE DO JĘZYKA KOTLIN 23

Rozdział 1. Kotlin: co to jest i po co? 25

1.1. Przedsmak Kotlin 25

1.2. Najważniejsze cechy języka Kotlin 26

1.2.1. Docelowe platformy: serwery, Android i wszystko, gdzie jest Java 26

1.2.2. Statyczne typowanie danych 27

1.2.3. Programowanie funkcyjne i obiektowe 28

1.2.4. Bezpłatne i otwarte oprogramowanie 29

1.3. Zastosowania języka Kotlin 30

1.3.1. Kotlin na serwerach 30

1.3.2. Kotlin w Androidzie 31

1.4. Filozofia języka Kotlin 32

1.4.1. Pragmatyzm 32

1.4.2. Zwięzłość 33

- 1.4.3. Bezpieczeństwo 34
- 1.4.4. Kompatybilność 35
- 1.5. Narzędzia języka Kotlin 35
 - 1.5.1. Kompilator kodu 36
 - 1.5.2. Wtyczki dla IntelliJ IDEA i Android Studio 36
 - 1.5.3. Interaktywna powłoka 37
 - 1.5.4. Wtyczka dla Eclipse 37
 - 1.5.5. Internetowy "plac zabaw" 37
 - 1.5.6. Konwerter Java-Kotlin 37
- 1.6. Podsumowanie 38

Rozdział 2. Podstawy języka Kotlin 39

- 2.1. Podstawowe elementy: funkcje i zmienne 39
 - 2.1.1. Witaj, świecie! 40
 - 2.1.2. Funkcje 40
 - 2.1.3. Zmienne 42
 - 2.1.4. Proste formatowanie ciągów znaków: szablony 43
- 2.2. Klasy i właściwości 44
 - 2.2.1. Właściwości 45
 - 2.2.2. Własne metody dostępne 47
 - 2.2.3. Układ kodu źródłowego: katalogi i pakiety 48
- 2.3. Kodowanie i dokonywanie wyborów: klasa wyliczeniowa i wyrażenie when 49
 - 2.3.1. Deklarowanie klasy wyliczeniowej 49
 - 2.3.2. Wyrażenie when i klasy wyliczeniowe 50
 - 2.3.3. Wyrażenie when i dowolne obiekty 51
 - 2.3.4. Wyrażenie when bez argumentów 52
 - 2.3.5. Inteligentne rzutowanie: połączenie sprawdzania i rzutowania typów 53

- 2.3.6. Refaktoryzacja kodu: zamiana if na when 55
- 2.3.7. Bloki kodu w odgałęzieniach wyrażeń if i when 56
- 2.4. Iteracje: pętle while i for 57
 - 2.4.1. Pętla while 57
 - 2.4.2. Iteracje liczb: zakresy i postępy 57
 - 2.4.3. Iterowanie elementów map 59
 - 2.4.4. Sprawdzanie przynależności do zakresu i kolekcji za pomocą słowa in 60
- 2.5. Wyjątki w Kotlinie 61
 - 2.5.1. Instrukcje try, catch i finally 62
 - 2.5.2. Słowo kluczowe try jako wyrażenie 63
- 2.6. Podsumowanie 64

Rozdział 3. Definiowanie i wywoływanie funkcji 65

- 3.1. Tworzenie kolekcji 66
- 3.2. Łatwiejsze wywoływanie funkcji 67
 - 3.2.1. Nazwane argumenty 68
 - 3.2.2. Domyślne wartości argumentów 68
 - 3.2.3. Koniec ze statycznymi klasami pomocniczymi, czyli funkcje i właściwości najwyższego poziomu 70
- 3.3. Dodawanie elementów do zewnętrznych klas: funkcje i właściwości rozszerzające 72
 - 3.3.1. Importowanie klas a funkcje rozszerzające 73
 - 3.3.2. Wywoływanie funkcji rozszerzających w kodzie Java 74
 - 3.3.3. Funkcje pomocnicze jako rozszerzenia 74
 - 3.3.4. Nienadpiszalność funkcji rozszerzających 75
 - 3.3.5. Właściwości rozszerzające 76
- 3.4. Przetwarzanie kolekcji: funkcjonalność varargs, wywołania infix i obsługa bibliotek 77
 - 3.4.1. Rozbudowa interfejsu API kolekcji Java 78
 - 3.4.2. Deklarowanie funkcji o dowolnej liczbie argumentów 78

- 3.4.3. Działania w parach: wywołania infix i deklaracje destrukuryzujące 79
- 3.5. Operacje na ciągach znaków i wyrażeniach regularnych 80
 - 3.5.1. Dzielenie ciągów znaków 81
 - 3.5.2. Wyrażenia regularne i potrójne cudzysłowy 81
 - 3.5.3. Potrójne cudzysłowy i wielowierszowe ciągi znaków 83
- 3.6. Wygładzanie kodu: lokalne funkcje i rozszerzenia 84
- 3.7. Podsumowanie 86

Rozdział 4. Klasy, obiekty i interfejsy 89

- 4.1. Definiowanie hierarchii klas 90
 - 4.1.1. Interfejsy w Kotlinie 90
 - 4.1.2. Modyfikatory open, final (domyślny) i abstract 92
 - 4.1.3. Modyfikatory widoczności, domyślny public 94
 - 4.1.4. Klasy wewnętrzne i zagnieżdżone (domyślnie) 96
 - 4.1.5. Klasy zapieczętowane: definiowanie ograniczonych hierarchii klas 98
- 4.2. Deklarowanie klas z nietrywialnymi konstruktorami i właściwościami 100
 - 4.2.1. Inicjowanie klas: konstruktor główny i bloki inicjatora 100
 - 4.2.2. Konstruktory dodatkowe i różne sposoby inicjowania klas nadrzędnych 102
 - 4.2.3. Implementowanie właściwości zadeklarowanych w interfejsie 104
 - 4.2.4. Dostęp do pól za pomocą getterów i setterów 105
 - 4.2.5. Zmienianie widoczności metody dostępowej 106
- 4.3. Metody generowane przez kompilator, klasy danych i delegowanie klas 108
 - 4.3.1. Metody uniwersalnych obiektów 108
 - 4.3.2. Klasy danych i automatyczne generowanie uniwersalnych metod 111
 - 4.3.3. Delegowanie klas i słowo kluczowe by 112
- 4.4. Słowo kluczowe object łączące deklarację klasy z utworzeniem jej instancji 114
 - 4.4.1. Łatwe tworzenie singletonów poprzez deklarowanie obiektów 114

- 4.4.2. Obiekty towarzyszące: miejsce dla metod wytwórczych i elementów statycznych 116
- 4.4.3. Obiekty towarzyszące jako zwykłe obiekty 118
- 4.4.4. Wyrażenia obiektowe, czyli anonimowe klasy wewnętrzne 121
- 4.5. Podsumowanie 122

Rozdział 5. Wyrażenia lambda 123

- 5.1. Wyrażenia lambda i odwołania do elementów obiektów 123
 - 5.1.1. Wprowadzenie do wyrażeń lambda: bloki kodu jako argumenty funkcji 124
 - 5.1.2. Lambdy i kolekcje 125
 - 5.1.3. Składnia wyrażenia lambda 126
 - 5.1.4. Odwołania do zmiennych w bieżącym kontekście 129
 - 5.1.5. Odwołania do elementów klas 131
- 5.2. Interfejsy funkcyjne do przetwarzania kolekcji 133
 - 5.2.1. Podstawy: filtry i mapy 133
 - 5.2.2. Warunki i funkcje `all()`, `any()`, `count()` oraz `find()` w kolekcjach 135
 - 5.2.3. Funkcja `groupBy()` i konwersja listy na mapę grup 136
 - 5.2.4. Funkcja `flatMap()`, spłaszczanie struktury danych i przetwarzanie zagnieżdżonych kolekcji 137
- 5.3. Leniwe operacje na kolekcjach oraz sekwencje 138
 - 5.3.1. Pośrednie i końcowe operacje na sekwencjach 139
 - 5.3.2. Tworzenie sekwencji 142
- 5.4. Interfejsy funkcyjne Java 143
 - 5.4.1. Umieszczanie wyrażeń lambda w argumentach metod Java 144
 - 5.4.2. Konstruktory SAM i jawna konwersja wyrażeń lambda na interfejsy funkcyjne 146
- 5.5. Wyrażenia lambda, odbiorniki oraz funkcje `with()` i `apply()` 147
 - 5.5.1. Funkcja `with()` 147
 - 5.5.2. Funkcja `apply()` 149
- 5.6. Podsumowanie 151

Rozdział 6. System typów danych 153

6.1. Zerowalność typów danych 153

6.1.1. Zerowalne typy danych 154

6.1.2. Znaczenie typów danych 156

6.1.3. Bezpieczny operator wywołania "?." 157

6.1.4. Operator Elvisa "?:" 158

6.1.5. Bezpieczne rzutowanie typów: operator "as?" 160

6.1.6. Asercja niezerowa "!!" 161

6.1.7. Funkcja let() 163

6.1.8. Właściwości inicjowane z opóźnieniem 164

6.1.9. Rozszerzenia typów zerowalnych 166

6.1.10. Zerowalność argumentów typowanych 167

6.1.11. Zerowalność typów i Java 168

6.2. Typy proste oraz inne typy podstawowe 172

6.2.1. Typy proste Int, Boolean i inne 172

6.2.2. Zerowalne typy proste Int?, Boolean? i inne 173

6.2.3. Przekształcanie liczb 174

6.2.4. Typy główne "Any" i "Any?" 176

6.2.5. Typ Unit, odpowiednik "void" 177

6.2.6. Typ Nothing, czyli "funkcja nigdy nie kończy działania" 178

6.3. Kolekcje i tablice 178

6.3.1. Zerowalność typów danych i kolekcje 178

6.3.2. Kolekcje tylko do odczytu i kolekcje mutowalne 181

6.3.3. Kolekcje w Kotlinie i w Javie 182

6.3.4. Kolekcje jako typy platformowe 184

6.3.5. Tablice obiektów i typów prostych 186

6.4. Podsumowanie 189

CZĘŚĆ II. WZBOGACANIE KOTLINA 191

Rozdział 7. Przeciążanie operatorów oraz inne konwencje 193

7.1. Przeciążanie operatorów arytmetycznych 194

7.1.1. Przeciążanie dwuargumentowych operatorów arytmetycznych 194

7.1.2. Przeciążanie złożonych operatorów przypisania 196

7.1.3. Przeciążanie operatorów jednoargumentowych 198

7.2. Przeciążanie operatorów porównania 199

7.2.1. Operatory równości 199

7.2.2. Przeciążanie operatorów nierówności: metoda `compareTo()` 200

7.3. Konwencje stosowane w kolekcjach i zakresach 202

7.3.1. Dostęp do elementu za pomocą indeksu, metod `get()` i `set()` 202

7.3.2. Konwencja operatora `in` 203

7.3.3. Metoda `rangeTo()` 204

7.3.4. Konwencja "iterator" w pętli `loop` 205

7.4. Deklaracje destrukuryzujące i metody komponentowe 206

7.4.1. Deklaracje destrukuryzujące i pętle 207

7.5. Współdzielenie metod dostępowych i delegowanie właściwości 208

7.5.1. Podstawy delegowania właściwości 209

7.5.2. Korzystanie z delegowanych właściwości: inicjalizacja z opóźnieniem i funkcja `lazy` 209

7.5.3. Implementacja delegowanych właściwości 211

7.5.4. Zasady translacji delegowanych właściwości 215

7.5.5. Przechowywanie wartości właściwości w mapie 215

7.5.6. Delegowane właściwości w bibliotekach 216

7.6. Podsumowanie 218

Rozdział 8. Funkcje wysokopoziomowe: wyrażenia lambda jako argumenty oraz wyniki 219

8.1. Deklarowanie funkcji wysokopoziomowych 220

8.1.1. Typy funkcyjne 220

8.1.2. Wywoływanie funkcji podanych w argumentach 221

8.1.3. Stosowanie typów funkcyjnych w kodzie Java 222

8.1.4. Wartość domyślna i wartość null w argumentach typów funkcyjnych 223

8.1.5. Funkcje zawierające w wynikach inne funkcje 226

8.1.6. Usuwanie duplikatów kodu za pomocą wyrażen lambda 227

8.2. Funkcje śródwierszowe i wydajność wyrażen lambda 229

8.2.1. Wstawianie kodu funkcji 230

8.2.2. Ograniczenia funkcji śródwierszowych 232

8.2.3. Wstawianie operacji na kolekcjach 233

8.2.4. Kiedy należy stosować funkcje śródwierszowe 234

8.2.5. Zarządzanie zasobami za pomocą śródwierszowych wyrażen lambda 234

8.3. Sterowanie realizacją kodu w funkcjach wysokopoziomowych 236

8.3.1. Instrukcja return w wyrażeniach lambda: wyjście z nadrzędnej funkcji 236

8.3.2. Wyjście z wyrażenia lambda: instrukcja return z etykietą 237

8.3.3. Funkcje anonimowe i domyślne wyjścia lokalne 239

8.4. Podsumowanie 240

Rozdział 9. Typy generyczne 241

9.1. Generyczne argumenty typowane 242

9.1.1. Generyczne funkcje i właściwości 243

9.1.2. Deklarowanie klas generycznych 244

9.1.3. Ograniczenia argumentów typowanych 245

9.1.4. Deklarowanie niezerowalnego argumentu typowanego 247

9.2. Typy generyczne w działającym kodzie, wymazane i urzeczowione argumenty typowane 248

9.2.1. Typy generyczne w działającym kodzie: sprawdzanie i rzutowanie typów 248

9.2.2. Deklarowanie funkcji z urzeczowionymi argumentami typowanymi 250

9.2.3. Zastępowanie odwołań do klas urzeczowionymi argumentami typowanymi 252

9.2.4. Ograniczenia urzeczowionych argumentów typowanych 253

9.3. Wariacje, typy generyczne i podtypy 254

9.3.1. Idea wariacji i umieszczanie wartości w argumentach funkcji 254

9.3.2. Klasy, typy i podtypy 255

9.3.3. Kowariancja: zachowanie zależności między podtypami 257

9.3.4. Kontrawariancja: odwrotna zależność podtypów 261

9.3.5. Wariacja typu w miejscu deklaracji 263

9.3.6. Projekcja z gwiazdką: symbol * zamiast argumentu typowanego 266

9.4. Podsumowanie 270

Rozdział 10. Adnotacje i refleksja 271

10.1. Deklarowanie i stosowanie adnotacji 272

10.1.1. Stosowanie adnotacji 272

10.1.2. Adres adnotacji 273

10.1.3. Dostosowywanie procesu serializacji JSON za pomocą adnotacji 275

10.1.4. Deklarowanie adnotacji 277

10.1.5. Metaadnotacje: kontrolowanie procesu przetwarzania adnotacji 277

10.1.6. Klasy jako argumenty adnotacji 278

10.1.7. Klasy generyczne jako argumenty adnotacji 279

10.2. Refleksja: badanie obiektów w trakcie działania kodu 280

10.2.1. Interfejs API refleksji w Kotlinie: interfejsy KClass, KCallable, KFunction i KProperty 281

10.2.2. Serializacja obiektów z wykorzystaniem refleksji 285

10.2.3. Dostosowywanie serializacji za pomocą adnotacji 286

10.2.4. Analiza danych JSON i deserializacja obiektów 289

10.2.5. Ostatni etap deserializacji: wywołanie metody callBy() i utworzenie obiektu za pomocą refleksji 293

10.3. Podsumowanie 297

Rozdział 11. Definiowanie języka DSL 299

11.1. Od interfejsu API do języka DSL 300

11.1.1. Idea języków domenowych 301

11.1.2. Wewnętrzny język DSL 302

11.1.3. Struktura języka DSL 303

11.1.4. Generowanie kodu HTML za pomocą wewnętrznego języka DSL 304

11.2. Tworzenie strukturalnego interfejsu API: wyrażenia lambda z odbiornikami w języku DSL 305

11.2.1. Wyrażenie lambda z odbiornikiem i typ funkcyjny rozszerzający 305

11.2.2. Wyrażenia lambda z odbiornikami w generatorze HTML 309

11.2.3. Generatory w Kotlinie, abstrakcje i powtarzalny kod 313

11.3. Bardziej elastyczne zagnieżdżanie bloków kodu dzięki konwencji invoke 316

11.3.1. Konwencja invoke, czyli obiekty wywoływane tak jak funkcje 316

11.3.2. Konwencja invoke i typy funkcyjne 317

11.3.3. Konwencja invoke w języku DSL: deklarowanie zależności w narzędziu Gradle 318

11.4. Język DSL w praktyce 319

11.4.1. Łączenie wywołań infix i asercja should w platformach testowych 319

11.4.2. Rozszerzenia typów prostych i przetwarzanie dat 321

11.4.3. Funkcje rozszerzające i wewnętrzny język DSL do obsługi zapytań SQL 322

11.4.4. Biblioteka Anko i dynamiczne tworzenie interfejsu użytkownika w systemie Android 325

11.5. Podsumowanie 327

DODATKI 329

Dodatek A. Kompilowanie projektów Kotlin 331

A.1. Kompilowanie kodu Kotlin za pomocą narzędzia Gradle 331

A.1.1. Kompilowanie za pomocą narzędzia Gradle aplikacji dla systemu Android 332

A.1.2. Kompilowanie projektów wykorzystujących adnotacje 332

A.2. Kompilowanie kodu Kotlin za pomocą narzędzia Maven 333

A.3. Kompilowanie kodu Kotlin za pomocą narzędzia Ant 333

Dodatek B. Dokumentowanie kodu Kotlin 335

B.1. Umieszczanie komentarzy dokumentacyjnych 335

B.2. Generowanie dokumentacji interfejsu API 336

Dodatek C. Ekosystem Kotlin 339

C.1. Testowanie kodu 339

C.2. Wstrzykiwanie zależności 340

C.3. Serializacja JSON 340

C.4. Klienci HTTP 340

C.5. Aplikacje WWW 340

C.6. Operacje na bazach danych 341

C.7. Narzędzia i struktury danych 341

C.8. Aplikacje stacjonarne 341

Skorowidz 343