

# Spis treści

	<b>Wstęp</b> . . . . .	9
<b>1</b>	<b>Klasy</b> . . . . .	13
	1.1. Abstrakcja i hermetyzacja . . . . .	15
	1.2. Enumeracje . . . . .	17
	1.3. Definiowanie klas . . . . .	22
	1.4. Wykorzystanie składowych statycznych . . . . .	25
	1.5. Przeciążanie metod i konstruktorów . . . . .	26
	1.6. Klasy i obiekty niezmiennicze . . . . .	27
	1.7. Inicjowanie . . . . .	28
	1.8. Singletony . . . . .	35
	1.9. Klasy opakowujące typy proste . . . . .	36
<b>2</b>	<b>Ponowne wykorzystanie klas</b> . . . . .	41
	2.1. Dziedziczenie . . . . .	43
	2.2. Konwersje referencyjne . . . . .	48
	2.3. Stwierdzanie typu . . . . .	50
	2.4. Dziedziczenie w Javie . . . . .	52
	2.5. Przedefiniowanie metod . . . . .	52
	2.6. Kowariancja typów wyników . . . . .	56
	2.7. Przedefiniowanie metod w wyliczeniach . . . . .	57
	2.8. Przedefiniowanie a wyjątki . . . . .	58
	2.9. Przedefiniowanie a przeciążanie, przestanie i pokrywanie . . . . .	59
	2.10. Adnotacja <code>@override</code> . . . . .	60
	2.11. Metody wirtualne i polimorfizm . . . . .	62
	2.12. Kompozycja . . . . .	67
	2.13. Kompozycja a dziedziczenie . . . . .	68
	2.14. Reguły ponownego wykorzystania klas . . . . .	71
<b>3</b>	<b>Wyjątki</b> . . . . .	73
	3.1. Obsługa wyjątków . . . . .	75
	3.2. Zgłaszanie wyjątków . . . . .	83
	3.3. Ponowne zgłaszanie wyjątków . . . . .	85
	3.4. Niskopoziomowe przyczyny i łańcuchowanie wyjątków . . . . .	90
	3.5. Wykorzystanie informacji o śladzie stosu . . . . .	91
<b>4</b>	<b>Interfejsy i klasy wewnętrzne</b> . . . . .	93
	4.1. Metody i klasy abstrakcyjne . . . . .	95
	4.2. Pojęcie interfejsu . . . . .	97
	4.3. Problem wielodziedziczenia . . . . .	97
	4.4. Definiowanie i implementowanie interfejsów . . . . .	99

4.5.	Interfejsy jako typy danych . . . . .	102
4.6.	Implementacja metod w interfejsach . . . . .	106
4.7.	Mixiny . . . . .	107
4.8.	Właściwości metod domyślnych . . . . .	108
4.9.	Prywatne metody w interfejsach . . . . .	111
4.10.	Pojęcie klasy wewnętrznej . . . . .	113
4.11.	Przykładowe zastosowanie klasy wewnętrznej . . . . .	114
4.12.	Anonimowe klasy wewnętrzne . . . . .	117
4.13.	Lokalne klasy wewnętrzne . . . . .	119
<b>5</b>	<b>Typy i metody sparametryzowane (generics)</b> . . . . .	<b>123</b>
5.1.	Definiowanie typów sparametryzowanych. Typy surowe i czyszczenie typów . . . . .	125
5.2.	Ograniczenia parametrów typu . . . . .	128
5.3.	Restrykcje . . . . .	129
5.4.	Metody sparametryzowane . . . . .	131
5.5.	Uniwersalne argumenty typu . . . . .	131
<b>6</b>	<b>Elementy programowania funkcyjnego w Javie 8. Przegląd pragmatyczny</b> . . . . .	<b>137</b>
6.1.	O programowaniu funkcyjnym . . . . .	139
6.2.	Interfejsy na pomoc . . . . .	140
6.3.	Lambda-wyrażenia: pierwsze spotkanie . . . . .	141
6.4.	O gotowych interfejsach funkcyjnych . . . . .	144
6.5.	O przetwarzaniu strumieniowym . . . . .	145
<b>7</b>	<b>Lambda-wyrażenia</b> . . . . .	<b>149</b>
7.1.	Interfejsy funkcyjne i lambda-wyrażenia . . . . .	151
7.2.	Składnia i cechy lambda-wyrażeń . . . . .	152
7.3.	Referencje do metod i konstruktorów . . . . .	158
7.4.	Gotowe interfejsy funkcyjne . . . . .	161
7.5.	Interfejsy z pakietu <i>java.util.function</i> a wyjątki kontrolowane . . . . .	169
7.6.	Lambda-wyrażenia i obiekty typu <i>Optional</i> . . . . .	172
<b>8</b>	<b>Proste narzędzia</b> . . . . .	<b>177</b>
8.1.	Analiza składniowa tekstów i wyrażenia regularne . . . . .	179
8.2.	Uproszczenia stosowania wyrażeń regularnych w klasach <i>String</i> i <i>Scanner</i> . . . . .	191
8.3.	Działania na liczbach . . . . .	199
8.4.	Daty i czas . . . . .	204
8.4.1.	Tradycyjna klasa <i>Calendar</i> i operacje na datach . . . . .	204
8.4.2.	Daty i czas w Javie 8 – elementy nowego API . . . . .	210
8.5.	Formatowanie liczb i dat . . . . .	217
8.6.	Metody tablicowe . . . . .	222
<b>9</b>	<b>Kolekcje</b> . . . . .	<b>225</b>
9.1.	Architektura kolekcji (JCF). Interfejsy i implementacje . . . . .	227
9.2.	Programowanie w kategoriach interfejsów . . . . .	231
9.3.	Ogólne operacje na kolekcjach . . . . .	233
9.4.	Operacje opcjonalne oraz wyjątki zgłaszane przez metody kolekcyjne . . . . .	234
9.5.	Przekształcanie kolekcji. Kolekcje z tablic . . . . .	235
9.6.	Przykłady ogólnych operacji na kolekcjach . . . . .	237
9.7.	Iterowanie po kolekcjach . . . . .	243
9.7.1.	Tradycyjny iterator i rozszerzone <i>for</i> . . . . .	243
9.7.2.	Iteracje wewnętrzne . . . . .	247
9.7.3.	Spliteratory . . . . .	249
9.7.4.	Konkurencyjne modyfikacje . . . . .	256

9.8.	Listy . . . . .	258
9.8.1.	Podstawowe implementacje i operacje . . . . .	258
9.8.2.	Szczególne implementacje list: niuanse metody <code>Arrays.asList(...)</code> . . . . .	264
9.8.3.	Iteratory listowe . . . . .	267
9.8.4.	Przykłady operacji na listach . . . . .	269
9.9.	Kolejki . . . . .	273
9.10.	Zbiory typu <code>HashSet</code> , metody <code>hashCode()</code> i <code>equals()</code> . . . . .	276
9.11.	Porównywanie i porządkowanie elementów kolekcji . . . . .	283
9.12.	Zbiory uporządkowane i nawigowalne . . . . .	291
9.13.	Mapy . . . . .	293
9.13.1.	Wprowadzenie . . . . .	293
9.13.2.	Implementacje i interfejsy. Ogólne operacje na mapach . . . . .	296
9.13.3.	Iterowanie po mapach . . . . .	301
9.13.4.	Użycie domyślnych metod interfejsu <code>Map</code> . . . . .	304
9.13.5.	Sortowanie map . . . . .	307
9.14.	Algorytmy, widoki, fabrykatory kolekcji . . . . .	313
9.15.	Własne implementacje kolekcji . . . . .	316
<b>10</b>	<b>Przetwarzanie strumieniowe</b> . . . . .	<b>317</b>
10.1.	Pojęcie strumienia. Rodzaje i cechy operacji strumieniowych . . . . .	319
10.2.	Uzyskiwanie strumieni . . . . .	321
10.3.	Przegląd operacji na strumieniach . . . . .	322
10.4.	Filtrowanie i leniwość strumieni . . . . .	325
10.5.	Metoda <code>forEach</code> dla strumieni . . . . .	326
10.6.	Sortowanie strumieni . . . . .	327
10.7.	Redukcja . . . . .	328
10.8.	Proste kolektory . . . . .	329
10.9.	Kolektory budujące mapy . . . . .	330
10.10.	Generatory . . . . .	332
10.11.	Strumienie równoległe . . . . .	334
10.12.	Przykłady innych użytecznych metod . . . . .	336
<b>11</b>	<b>Wejście-wyjście</b> . . . . .	<b>339</b>
11.1.	Programowanie wejścia-wyjścia: obraz ogólny . . . . .	341
11.2.	Abstrakcyjne strumienie wejścia-wyjścia. Operacje elementarne . . . . .	342
11.3.	Strumieniowe klasy przedmiotowe . . . . .	346
11.4.	Instrukcja <code>try-with-resources</code> . Automatyczne zarządzanie zasobami a obsługa tłumionych wyjątków . . . . .	349
11.5.	Strumieniowe klasy przetwarzające – przegląd . . . . .	356
11.6.	Buforowanie . . . . .	359
11.7.	Binarne strumienie wejścia-wyjścia . . . . .	360
11.8.	Kodowanie-dekodowanie przy użyciu strumieni wejścia-wyjścia. . . . .	362
11.9.	Serializacja obiektów . . . . .	364
11.10.	Potoki . . . . .	369
11.11.	Analiza składniowa strumieni – <code>StreamTokenizer</code> . . . . .	371
11.12.	Obiekty plikowe i klasa <code>File</code> . . . . .	373
11.13.	Wygodne metody klasy <code>java.nio.file.Files</code> . . . . .	374
11.14.	Skaner . . . . .	378
11.15.	Przeglądanie katalogów . . . . .	381
11.16.	Archiwa . . . . .	390
11.17.	Pliki o dostępie swobodnym . . . . .	397
11.18.	Nowe wejście-wyjście (NIO): przegląd . . . . .	398
11.19.	NIO: bufory . . . . .	401
11.20.	NIO: kanały i bufory. Kanały plikowe . . . . .	406
11.21.	Widoki buforów bajtowych . . . . .	410

11.22.	NIO: bufory – uporządkowanie bajtów ( <i>endianess</i> ) . . . . .	414
11.23.	NIO: bufory znakowe. Kodowanie i dekodowanie . . . . .	416
11.24.	NIO: operacje kanałowe na wielu buforach ( <i>scattering i gathering</i> ) . . . . .	418
11.25.	NIO: mapowanie plików . . . . .	420
11.26.	NIO: bezpośrednie transfery kanałowe . . . . .	424
<b>12</b>	<b>Programowanie współbieżne i równoległe . . . . .</b>	<b>425</b>
12.1.	Procesy i wątki . . . . .	427
12.2.	Uruchamianie równoległych działających. Tworzenie wątków . . . . .	428
12.3.	Zadania i wykonawcy . . . . .	432
12.4.	Zamykanie wykonawców. Oczekiwanie na zakończenie zadań i serwis kompletacji ( <i>CompletionService</i> ) . . . . .	439
12.5.	Zadania powtarzalne, opóźnione i okresowe . . . . .	444
12.6.	Wykonawcy a pule wątków . . . . .	449
12.7.	<i>ForkJoinPool</i> i zadania rekursywne . . . . .	452
12.8.	Kompletery typu <i>CountedCompleter</i> . . . . .	463
12.9.	<i>CountedCompleter</i> jako kontynuacja . . . . .	469
12.10.	Kiedy i jak używać zadań typu <i>ForkJoinTask</i> ? Praktyczny przykład użycia kompleterów . . . . .	473
12.11.	Zadania kompletowalne ( <i>CompletableFuture</i> ) . . . . .	478
12.12.	Przerywanie zadań z zewnątrz i kończenie pracy wątków . . . . .	491
12.13.	Synchronizacja . . . . .	498
12.14.	Synchronizacja za pomocą jawnego ryglowania . . . . .	507
12.15.	Rygle do odczytu i zapisu: <i>ReentrantReadWriteLock</i> i <i>StampedLock</i> . . . . .	510
12.16.	Synchronizatory wyższego poziomu . . . . .	514
12.17.	Unikanie synchronizacji: struktura kodu, <i>volatile</i> , atomiki i konkurencyjne kolekcje . . . . .	519
12.18.	Koordinacja pracy wątków – mechanizm <i>wait-notify</i> . . . . .	526
12.19.	Koordinacja: warunki . . . . .	532
<b>13</b>	<b>Dynamiczna Java . . . . .</b>	<b>535</b>
13.1.	Mechanizm refleksji . . . . .	537
13.2.	Uchwyty metod . . . . .	543
13.3.	Znaczenie refleksji – praktyczne przykłady . . . . .	551
13.4.	Refleksja a <i>generics</i> . . . . .	558
13.5.	Dynamiczne klasy <i>proxy</i> . . . . .	562
13.6.	JavaBeans . . . . .	568
13.6.1.	Koncepcja JavaBeans . . . . .	568
13.6.2.	Nastuch i wetowanie zmian właściwości za pomocą obsługi zdarzeń . . . . .	569
13.6.3.	Introspekcja . . . . .	575
13.7.	Adnotacje . . . . .	576
13.7.1.	Istota adnotacji i sposoby ich definiowania . . . . .	576
13.7.2.	Przetwarzanie adnotacji w fazie wykonania . . . . .	580
13.7.3.	Przetwarzanie adnotacji w fazie kompilacji. Transformowanie kodu bajtowego . . . . .	582
13.8.	Skrypty w <i>Javie</i> . . . . .	589
<b>14</b>	<b>Lokalizacja i internacjonalizacja aplikacji . . . . .</b>	<b>595</b>
14.1.	Lokalizacje . . . . .	597
14.2.	Jeszcze trochę o formatorach liczbowych . . . . .	602
14.3.	Waluty . . . . .	603
14.4.	Strefy czasowe . . . . .	604
14.5.	Kalendarze i zlokalizowane daty . . . . .	606
14.6.	Porównywanie i sortowanie napisów . . . . .	607
14.7.	Internacjonalizacja aplikacji i dodatkowe zasoby ( <i>resource bundle</i> ) . . . . .	608
	<b>Literatura . . . . .</b>	<b>614</b>