

УДК 004.43
ББК 32.973.26-018.1
Д40

Последнюю версию этой книги на английском языке можно скачать с сайта: <https://goalkicker.com/JavaBook>. Пожалуйста, не стесняйтесь поделиться этим PDF-файлом с кем угодно бесплатно.

*Книга Java® Notes for Professionals составлена на основе документации Stack Overflow (<https://archive.org/details/documentation-dump.7z>), содержание которой написано прекрасными людьми из Stack Overflow. **Текстовые материалы публикуются на условиях Creative Commons BY-SA.** В конце книги указаны авторы, внесшие вклад в создание различных глав. Изображения могут являться объектами авторского права соответствующих владельцев, если не указано иное.*

Java. Самое полное руководство по разработке в примерах от сообщества Stack Overflow. — Москва : Издательство АСТ, 2024. — 672 с. : ил. — (Программирование от экспертов).

ISBN 978-5-17-160268-0.

Эта книга — не академический учебник по Java, а скорее сборник своеобразных рецептов по применению этого универсального языка программирования, которые могут пригодиться в самых разных случаях, связанных с написанием кода. Ее можно использовать для разрешения сложных ситуаций, возникающих у пользователей при работе с Java. Многие из представленного здесь материала ранее не публиковались в русскоязычных учебниках по языку Java, например, в книге рассмотрены особенности разработки приложений на основе применения параллельного программирования, работа с потоками, а также использование Java при вызове цепочки методов.

Данное издание может стать незаменимым помощником как для начинающего программиста, стремящегося разобраться во всех тонкостях языка Java, так и для более опытных разработчиков, которые смогут использовать книгу в качестве справочника для решения повседневных задач при написании кода.

УДК 004.43

ББК 32.973.26-018.1

ISBN 978-5-17-160268-0

Перевод на русский язык: ООО «Интеджер»

Издание на русском языке: ООО «Издательство АСТ»

Содержание

Глава 1. Начало работы с языком Java	18
Раздел 1.1. Создание первой Java-программы	18
Глава 2. Преобразование типов	23
Раздел 2.1. Приведение числовых примитивов	23
Раздел 2.2. Основные числовые преобразования	23
Раздел 2.3. Приведение нечисловых примитивов	24
Раздел 2.4. Приведение объектов	24
Раздел 2.5. Проверка возможности приведения объекта с помощью метода instanceof	24
Глава 3. Геттеры и сеттеры	25
Раздел 3.1. Использование сеттера или геттера для наложения определенных ограничений при доступе к переменным	25
Раздел 3.2. Зачем использовать геттеры и сеттеры?	26
Раздел 3.3. Добавление геттеров и сеттеров	27
Глава 4. Ссылочные типы данных	28
Раздел 4.1. Разыменование указателя	28
Раздел 4.2. Создание переменной ссылочного типа	28
Глава 5. Компилятор Java — ‘javac’	29
Раздел 5.1. Команда ‘javac’ — начало работы	29
Раздел 5.2. Компиляция для другой версии Java	31
Глава 6. Документирование Java-кода	32
Раздел 6.1. Создание документации с помощью Javadoc из командной строки	32
Раздел 6.2. Документирование классов	33
Раздел 6.3. Документирование методов	34
Раздел 6.4. Документирование пакета	34
Раздел 6.5. Ссылки	35
Раздел 6.6. Фрагменты кода внутри документации	36
Раздел 6.7. Документирование полей	37
Раздел 6.8. Документация по встроенному коду	37
Глава 7. Обработка аргументов командной строки	38
Раздел 7.1. Обработка аргументов с помощью GWT ToolBase	38
Раздел 7.2. Обработка аргументов вручную	39
Глава 8. Команды Java — ‘java’ и ‘javaw’	41
Раздел 8.1. Класс с точкой входа	41
Раздел 8.2. Устранение ошибок при использовании команды ‘java’	41
Раздел 8.3. Запуск Java-приложения с зависимостями от библиотек	43
Раздел 8.4. Опции для команды java	44
Раздел 8.5. Пробелы и другие специальные символы в аргументах	46
Раздел 8.6. Запуск исполняемого JAR-файла	47
Раздел 8.7. Запуск Java-приложений с помощью класса, содержащего метод “main”	47
Глава 9. Литералы	48
Раздел 9.1. Использование подчеркивания для улучшения читабельности	48
Раздел 9.2. Шестнадцатеричные, восьмеричные и двоичные литералы	49
Раздел 9.3. Логические литералы (типа boolean)	49
Раздел 9.4. Строковые литералы	49
Раздел 9.5. Литерал Null	50
Раздел 9.6. Управляющие последовательности в литералах	51
Раздел 9.7. Символьные литералы	52
Раздел 9.8. Целочисленные десятичные литералы	52
Раздел 9.9. Литералы с плавающей точкой	53
Глава 10. Примитивные типы данных	55
Раздел 10.1. Примитив char	55
Раздел 10.2. Шпаргалка по примитивным типам	56
Раздел 10.3. Примитивный тип данных float	57
Раздел 10.4. Примитивный тип данных int	58

Раздел 10.5. Преобразование примитивов	59
Раздел 10.6. Сравнение использования памяти примитивами и классами-обертками	59
Раздел 10.7. Примитивный тип данных double	60
Раздел 10.8. Примитивный тип данных long	61
Раздел 10.9. Примитивный тип данных boolean	62
Раздел 10.10. Примитивный тип данных byte	62
Раздел 10.11. Представление отрицательных значений	63
Раздел 10.12. Примитивный тип данных short	64
Глава 11. Строки	64
Раздел 11.1. Сравнение строк	64
Раздел 11.2. Изменение регистра символов в строке	67
Раздел 11.3. Проверка вхождения строки в состав другой строки	68
Раздел 11.4. Пул строк и область памяти Java Heap	69
Раздел 11.5. Разделение строк на подстроки	70
Раздел 11.6. Объединение строк с помощью разделителя	73
Раздел 11.7. Конкатенация строк и класс StringBuilder	73
Раздел 11.8. Подстроки	75
Раздел 11.9. Платформенезависимая реализация добавления символа перевода строки	76
Раздел 11.10. Выполнение реверса строк	76
Раздел 11.11. Добавление метода toString() в объекты пользователя	77
Раздел 11.12. Удаление пробельных символов из начала и конца строки	78
Раздел 11.13. Оператор switch, нечувствительный к регистру	78
Раздел 11.14. Замена частей строк	79
Раздел 11.15. Получение длины строки	80
Раздел 11.16. Получение n-го символа в строке	80
Раздел 11.17. Подсчет количества вхождений подстроки / символа в строку	80
Глава 12. Класс StringBuffer	81
Раздел 12.1. Класс StringBuffer	81
Глава 13. Класс StringBuilder	82
Раздел 13.1. Сравнение StringBuffer, StringBuilder, Formatter и StringJoiner	82
Раздел 13.2. Повторение строки n раз	83
Глава 14. Класс StringTokenizer	84
Раздел 14.1. Разделение по пробелам с помощью StringTokenizer	84
Раздел 14.2. Разделение запятой ',' с помощью StringTokenizer	84
Глава 15. Разбиение строки на части фиксированной длины	85
Раздел 15.1. Разбиение строки на подстроки заданной длины	85
Раздел 15.2. Разбиение строки на подстроки произвольной длины	85
Глава 16. Класс Date	85
Раздел 16.1. Преобразование экземпляра класса java.util.Date в экземпляр класса java.sql.Date	85
Раздел 16.2. Вывод даты в основном формате	86
Раздел 16.3. Объекты LocalDate и LocalDateTime в версии Java 8	87
Раздел 16.4. Создание конкретной даты	88
Раздел 16.5. Преобразование даты в определенный формат строки	88
Раздел 16.6. LocalTime	89
Раздел 16.7. Преобразование отформатированной строки, содержащей дату, в объект Date	89
Раздел 16.8. Создание объектов Date	90
Раздел 16.9. Сравнение объектов Date	90
Раздел 16.10. Преобразование строки в дату	93
Раздел 16.11. Временные зоны и объект java.util.Date	94
Глава 17. Библиотека для работы с датой и временем (java.time.*)	94
Раздел 17.1. Вычислить разницу между двумя датами, представленными объектами LocalDate	94
Раздел 17.2. Дата и время	94
Раздел 17.3. Операции со значениями даты и времени	95
Раздел 17.4. Класс Instant	95
Раздел 17.5. Использование различных классов для работы с Date Time API	95
Раздел 17.6. Форматирование даты и времени	97
Раздел 17.7. Простые действия с датами	98

Глава 18. LocalTime	99
Раздел 18.1. Количество времени между двумя значениями LocalTime	99
Раздел 18.2. Введение.....	100
Раздел 18.3. Изменение времени.....	101
Раздел 18.4. Часовые пояса и разница во времени	101
Глава 19. Класс BigDecimal	101
Раздел 19.1. Сравнение больших чисел.....	102
Раздел 19.2. Использование значений BigDecimal вместо значений типа float.....	102
Раздел 19.3. Метод BigDecimal.valueOf()	103
Раздел 19.4. Математические операции с большими числами (BigDecimal).....	103
Раздел 19.5. Инициализация числа BigDecimals значениями ноль, один или десять	106
Раздел 19.6. Объекты BigDecimal являются неизменяемыми	106
Глава 20. Класс BigInteger	106
Раздел 20.1. Инициализация.....	107
Раздел 20.2. Примеры математических операций с числами BigInteger.....	108
Раздел 20.3. Сравнение чисел BigIntegers.....	109
Раздел 20.4. Бинарные логические операции над значениями BigInteger	110
Раздел 20.5. Генерация случайных BigIntegers	111
Глава 21. Класс NumberFormat	112
Раздел 21.1. Формат числа.....	112
Глава 22. Побитовые операции	113
Раздел 22.1. Проверка, установка, сброс и переключение отдельных битов. Использование значения типа long в качестве битовой маски	113
Раздел 22.2. Класс java.util.BitSet.....	113
Раздел 22.3. Как проверить, является ли значение степенью числа 2	114
Раздел 22.4. Знаковый и беззнаковый сдвиг	115
Раздел 22.5. Вычисление степени числа 2	116
Раздел 22.6. Упаковка/распаковка битовых групп	117
Глава 23. Массивы	117
Раздел 23.1. Создание и инициализация массивов	118
Раздел 23.2. Создание списка на основе массива	124
Раздел 23.3. Создание массива на основе коллекции	126
Раздел 23.4. Многомерные и зубчатые (ступенчатые) массивы	127
Раздел 23.5. Исключение ArrayIndexOutOfBoundsException	129
Раздел 23.6. Ковариантность массивов.....	129
Раздел 23.7. Преобразование массивов в поток	130
Раздел 23.8. Перебор элементов массива	131
Раздел 23.9. Преобразование массива в строку	133
Раздел 23.10. Сортировка массивов	134
Раздел 23.11. Получение длины массива	136
Раздел 23.12. Поиск элемента в массиве	136
Раздел 23.13. Как изменить размер массива?.....	138
Раздел 23.14. Преобразование массивов с примитивными типами данных в типы данных на основе классов-обертки и наоборот.....	139
Раздел 23.15. Удаление элемента из массива	139
Раздел 23.16. Сравнение массивов.....	140
Раздел 23.17. Копирование массивов	141
Раздел 23.18. Приведение типов массивов.....	142
Глава 24. Коллекции (Collections)	142
Раздел 24.1. Удаление элементов из списка в цикле	142
Раздел 24.2. Создание коллекций на основе имеющихся данных	145
Раздел 24.3. Объявление списка ArrayList и добавление объектов в него.....	147
Раздел 24.4. Перебор элементов в коллекциях	147
Раздел 24.5. Неизменяемые пустые коллекции	149
Раздел 24.6. Подколлекции.....	149
Раздел 24.7. Неизменяемые коллекции	150
Раздел 24.8. “Подводные камни”: исключения, связанные с попыткой внесения изменений одновременно от нескольких методов.....	151

Раздел 24.9. Удаление из списков совпадающих элементов с помощью итератора.....	151
Раздел 24.10. Объединение списков	152
Раздел 24.11. Создание собственной итерируемой структуры для возможности использования ее с итератором или с циклом for-each	153
Раздел 24.12. Коллекции и примитивные типы данных.....	155
Глава 25. Списки (Lists)	155
Раздел 25.1. Сортировка обобщенного списка	155
Раздел 25.2. Преобразование списка целых чисел в список строк	157
Раздел 25.3. Список классов, реализующих интерфейс List — плюсы и минусы.....	157
Раздел 25.4. Поиск общих элементов между двумя списками.....	159
Раздел 25.5. Замена элемента списка на месте.....	160
Раздел 25.6. Создание неизменяемого списка.....	160
Раздел 25.7. Перемещение объектов в списке.....	160
Раздел 25.8. Создание, добавление и удаление элементов в списке ArrayList	161
Раздел 25.9. Создание списка	162
Раздел 25.10. Операции доступа к элементу по его позиции	163
Раздел 25.11. Итерация элементов в списке	164
Раздел 25.12. Удаление из списка B элементов, присутствующих в списке A	165
Глава 26. Множества	165
Раздел 26.1. Инициализация.....	165
Раздел 26.2. Множества. Основные понятия	166
Раздел 26.3. Типы множеств и их использование	167
Раздел 26.4. Создание списка на основе множества	168
Раздел 26.5. Устранение дубликатов с помощью Set	168
Раздел 26.6. Объявление HashSet на основе исходных данных	169
Глава 27. Сравнение списка и множества	169
Раздел 27.1. Сравнение списка и множества	169
Глава 28. Словари (отображения)	170
Раздел 28.1. Эффективный перебор элементов словаря	170
Раздел 28.2. Использование интерфейса HashMap	173
Раздел 28.3. Использование методов Map по умолчанию, введенных в Java 8.....	174
Раздел 28.4. Перебор элементов словаря.....	176
Раздел 28.5. Слияние, объединение и композиция карт	177
Раздел 28.6. Как добавить несколько элементов	178
Раздел 28.7. Создание и инициализация словаря	180
Раздел 28.8. Проверка на наличие ключа в словаре.....	181
Раздел 28.9. Добавить элемент в словарь.....	182
Раздел 28.10. Удалить все элементы из словаря.....	182
Раздел 28.11. Использование в качестве ключа собственного объекта	182
Глава 29. LinkedHashMap	183
Раздел 29.1. Класс LinkedHashMap.....	183
Глава 30. WeakHashMap	184
Раздел 30.1. Концепции WeakHashMap.....	184
Глава 31. SortedMap	185
Раздел 31.1. Введение в SortedMap.....	185
Глава 32. TreeMap и TreeSet	186
Раздел 32.1. Создание TreeMap на основе простого типа Java	186
Раздел 32.2. Создание TreeSet на основе простого типа Java	186
Раздел 32.3. Создание TreeMap/TreeSet на основе собственного типа	187
Раздел 32.4. Безопасность потоков при использовании TreeMap и TreeSet	189
Глава 33. Очереди (Queues and Deques)	190
Раздел 33.1. Использование очереди с приоритетом PriorityQueue	190
Раздел 33.2. Двусторонняя очередь Deque	190
Раздел 33.3. Стеки	191
Раздел 33.4. BlockingQueue	193
Раздел 33.5. LinkedList как очередь FIFO	194
Раздел 33.6. Интерфейс Queue	194

Глава 34. Интерфейс Dequeue	195
Раздел 34.1. Добавление элементов в Deque	195
Раздел 34.2. Удаление элементов из Deque	195
Раздел 34.3. Извлечение элемента без удаления	195
Раздел 34.4. Перебор элементов в Deque	196
Глава 35. Перечисления (enum)	196
Раздел 35.1. Объявление перечисления и основные операции	196
Раздел 35.2. Перечисления с конструкторами	200
Раздел 35.3. Перечисления с абстрактными методами	201
Раздел 35.4. Реализация интерфейса	202
Раздел 35.5. Реализация паттерна Singleton с перечислением, в состав которого входит только один элемент	203
Раздел 35.6. Использование методов и статических блоков	203
Раздел 35.7. Перечисление, не содержащее элементов	204
Раздел 35.8. Использование enum в качестве параметра с ограничениями	205
Раздел 35.9. Документирование перечислений	205
Раздел 35.10. Использование в enum блока константы, специфичного для содержимого	206
Раздел 35.11. Получение значений констант в перечислении	208
Раздел 35.12. Enum и паттерн для полиморфизма	208
Раздел 35.13. Сравнение значений перечислений и определение вхождения константы в состав enum	209
Раздел 35.14. Поиск константы, входящей в перечисление, по ее имени	210
Раздел 35.15. Перечисления со свойствами (полями)	211
Раздел 35.16. Преобразование перечисления в строку	211
Раздел 35.17. Перечисления со статическими полями	212
Глава 36. Класс EnumMap	213
Раздел 36.1. Пример словаря EnumMap для применения в классе Book (собственный класс для работы с описанием книг)	213
Глава 37. Класс EnumSet	214
Раздел 37.1. Пример использования EnumSet	214
Глава 38. Перечисления, начинающиеся с цифры	214
Раздел 38.1. Перечисление с именем, расположенным в начале значения	215
Глава 39. Коллекция Hashtable	215
Раздел 39.1. Hashtable	215
Глава 40. Операторы	216
Раздел 40.1. Операторы инкремента/декремента (++/--)	216
Раздел 40.2. Условный оператор (? :)	217
Раздел 40.3. Побитовые и логические операторы (~, &, , ^)	218
Раздел 40.4. Оператор конкатенации строк (+)	219
Раздел 40.5. Арифметические операторы (+, -, *, /, %)	221
Раздел 40.6. Операторы сдвига (<<, >> и >>>)	224
Раздел 40.7. Оператор Instanceof	225
Раздел 40.8. Операторы присваивания (=, +=, -=, *=, /=, %=, <<=, >>=, >>>=, &=, = и ^=)	226
Раздел 40.9. Операторы “условное-и” и “условное-или” (&& и)	227
Раздел 40.10. Операторы отношения (<, <=, >, >=)	228
Раздел 40.11. Операторы равенства (==, !=)	229
Раздел 40.12. Оператор Lambda (->)	231
Глава 41. Конструкторы	231
Раздел 41.1. Конструктор по умолчанию	231
Раздел 41.2. Вызов родительского конструктора	232
Раздел 41.3. Конструктор с аргументами	234
Глава 42. Методы и конструктор класса Object	235
Раздел 42.1. Метод hashCode()	235
Раздел 42.2. Метод toString()	237
Раздел 42.3. Метод equals()	238
Раздел 42.4. Методы wait() и notify()	241
Раздел 42.5. Метод getClass()	243

Раздел 42.6. Метод clone()	244
Раздел 42.7. Конструктор объектов для класса Object.....	245
Раздел 42.8. Метод finalize()	246
Глава 43. Аннотации	247
Раздел 43.1. Идея аннотаций	247
Раздел 43.2. Определение аннотаций.....	247
Раздел 43.3. Проверка аннотаций в процессе выполнения с помощью API Reflection	250
Раздел 43.4. Встроенные аннотации.....	250
Раздел 43.5. Обработка аннотаций во время компиляции с помощью процессора аннотаций	253
Раздел 43.6. Повторяющиеся аннотации	257
Раздел 43.7. Наследуемые аннотации	258
Раздел 43.8. Получение значений аннотаций во время выполнения программы.....	259
Раздел 43.9. Аннотации для 'this' и параметров приемника.....	260
Раздел 43.10. Добавление нескольких значений аннотаций.....	261
Глава 44. Неизменяемый класс	262
Раздел 44.1. Пример без изменяемых ссылок.....	262
Раздел 44.2. В чем преимущество неизменяемости?.....	262
Раздел 44.3. Правила определения неизменяемых классов.....	262
Раздел 44.4. Пример с ссылками на изменяемый объект	263
Глава 45. Неизменяемые объекты	264
Раздел 45.1. Создание неизменяемой версии типа с помощью защитного копирования	264
Раздел 45.2. Рецепт создания неизменяемого класса.....	264
Раздел 45.3. Типичные недостатки проектирования, не позволяющие классу быть неизменяемым.....	265
Глава 46. Видимость (управление доступом к членам класса)	269
Раздел 46.1. Видимость объекта с модификатором доступа private	269
Раздел 46.2. Видимость объектов и переменных с модификатором доступа public	269
Раздел 46.3. Видимость объектов и переменных внутри пакета	270
Раздел 46.4. Видимость объектов и переменных с модификатором доступа protected.....	270
Раздел 46.5. Сводная информация о модификаторах доступа членов класса	271
Раздел 46.6. Члены интерфейса	271
Глава 47. Обобщенные типы данных (Generics)	272
Раздел 47.1. Создание обобщенного класса.....	272
Раздел 47.2. Критерии выбора между T, ? super T и ? extends T	275
Раздел 47.3. Алмаз (<>).....	276
Раздел 47.4. Объявление обобщенного метода.....	277
Раздел 47.5. Требование нескольких верхних границ ("extends A&B")	278
Раздел 47.6. Получение класса, удовлетворяющего обобщенному параметру во время выполнения программы	278
Раздел 47.7. Преимущества обобщенного класса и интерфейса	279
Раздел 47.8. Создание объектов обобщенного типа.....	279
Раздел 47.9. Создание обобщенного класса с ограничениями	280
Раздел 47.10. Обращение к объявленному обобщенному типу внутри его собственного объявления.....	281
Раздел 47.11. Привязка обобщенного параметра более чем к одному типу	283
Раздел 47.12. Использование обобщенного типа для автоматического приведения типов	284
Раздел 47.13. Использование оператора instanceof с обобщенными типами.....	284
Раздел 47.14. Различные способы реализации обобщенного интерфейса (или расширения обобщенного класса)	286
Глава 48. Классы и объекты	287
Раздел 48.1. Перегрузка методов	287
Раздел 48.2. Объяснение того, что такое перегрузка и переопределение методов	288
Раздел 48.3. Конструкторы.....	290
Раздел 48.4. Инициализация полей с модификатором доступа static final с помощью статического блока.....	291
Раздел 48.5. Основы построения объектов на базе классов и их использование	292

Раздел 48.6. Простейший возможный класс.....	294
Раздел 48.7. Сравнение обычного поля класса и поля с модификатором <code>static</code>	295
Глава 49. Локальный внутренний класс.....	296
Раздел 49.1. Локальный внутренний класс.....	296
Глава 50. Вложенные и внутренние классы.....	296
Раздел 50.1. Простой стек, реализованный на основе вложенного класса.....	296
Раздел 50.2. Статические и нестатические вложенные классы.....	297
Раздел 50.3. Модификаторы доступа для внутренних классов.....	299
Раздел 50.4. Анонимные внутренние классы.....	300
Раздел 50.5. Создание экземпляра нестатического внутреннего класса извне.....	301
Раздел 50.6. Внутренний класс в локальном методе.....	302
Раздел 50.7. Доступ к внешнему классу из нестатического внутреннего класса.....	302
Глава 51. Класс <code>java.util.Objects</code>.....	303
Раздел 51.1. Основное использование для проверки объекта на <code>null</code>	303
Раздел 51.2. Использование метода <code>Objects.nonNull()</code> в <code>stream api</code>	303
Глава 52. Метод по умолчанию (Default Methods).....	304
Раздел 52.1. Базовое использование метода по умолчанию.....	304
Раздел 52.2. Доступ к переопределенным методам по умолчанию из реализующего класса.....	305
Раздел 52.3. Зачем использовать методы по умолчанию?.....	305
Раздел 52.4. Доступ к другим методам интерфейса из метода по умолчанию.....	306
Раздел 52.5. Ошибка, вызванная совпадением имен методов по умолчанию при множественном наследовании.....	307
Раздел 52.6. Приоритет методов в классах, абстрактных классах и интерфейсах.....	307
Глава 53. Пакеты.....	308
Раздел 53.1. Использование пакетов для создания классов с одинаковыми именами.....	309
Раздел 53.2. Использование модификатора доступа <code>protected</code> для защиты содержимого пакета.....	309
Глава 54. Наследование.....	310
Раздел 54.1. Наследование.....	310
Раздел 54.2. Абстрактные классы.....	312
Раздел 54.3. Использование модификатора <code>final</code> для ограничения наследования и переопределения.....	313
Раздел 54.4. Принцип замещения Барбары Лисков.....	314
Раздел 54.5. Использование абстрактного класса и интерфейса: сравнение отношения “IS-A” и композиции “HAS-A”.....	315
Раздел 54.6. Наследование статических методов.....	318
Раздел 54.7. Программирование на интерфейсах.....	320
Раздел 54.8. Переопределение при наследовании.....	322
Раздел 54.9. Скрытие переменных.....	323
Раздел 54.10. Расширение и сужение ссылочных типов.....	323
Раздел 54.11. Наследование и статические методы.....	324
Глава 55. Ссылочные типы.....	325
Раздел 55.1. Различные типы ссылок.....	325
Глава 56. Консольный ввод/вывод.....	326
Раздел 56.1. Чтение пользовательского ввода с консоли.....	326
Раздел 56.2. Выравнивание строк в консоли.....	328
Раздел 56.3. Реализация базового поведения командной строки.....	329
Глава 57. Потoki (Streams).....	330
Раздел 57.1. Использование потоков.....	331
Раздел 57.2. Затратные потоки.....	333
Раздел 57.3. Создание словаря с частотами встречаемости.....	334
Раздел 57.4. Бесконечные потоки.....	335
Раздел 57.5. Сбор элементов потока в коллекцию.....	335
Раздел 57.6. Использование потоков для реализации математических функций.....	338
Раздел 57.7. Упаковка потоков с помощью метода <code>flatMap()</code>	338
Раздел 57.8. Параллельный поток.....	339
Раздел 57.9. Создание потока.....	340
Раздел 57.10. Получение статистики о числовых потоках.....	341

Раздел 57.11. Преобразование итератора в поток	342
Раздел 57.12. Использование IntStream для перебора элементов потока по индексу	342
Раздел 57.13. Объединение потоков.....	342
Раздел 57.14. Сокращение последовательностей элементов с помощью потоков	343
Раздел 57.15. Использование потоков Map.Entry для сохранения начальных значений после преобразования в словарь	344
Раздел 57.16. Преобразование потока IntStream в поток String.....	345
Раздел 57.17. Поиск первого элемента, соответствующего условию отбора элементов в поток	345
Раздел 57.18. Использование потоков и ссылок на методы для написания самодокументирующихся процессов.....	345
Раздел 57.19. Преобразование потока, содержащего объекты Optional, в поток обычных значений.....	347
Раздел 57.20. Получение фрагмента потока.....	347
Раздел 57.21. Создание словаря на основе потока	347
Раздел 57.22. Объединение элементов потока в одну строку	348
Раздел 57.23. Сортировка с помощью потока	349
Раздел 57.24. Потоки примитивов	350
Раздел 57.25. Категории операций с потоками.....	350
Раздел 57.26. Сбор элементов потока после обработки данных в массив	351
Раздел 57.27. Генерация случайных строк с помощью потоков	351
Глава 58. Классы InputStreams и OutputStreams	352
Раздел 58.1. Закрытие потоков.....	352
Раздел 58.2. Преобразование потока InputStream в строку.....	353
Раздел 58.3. Обертывание потоков ввода/вывода.....	353
Раздел 58.4. Пример использования класса DataInputStream	354
Раздел 58.5. Запись байтов в основной поток вывода OutputStream	355
Раздел 58.6. Копирование входного потока в выходной.....	355
Глава 59. Классы Readers и Writers	355
Раздел 59.1. Класс BufferedReader.....	355
Раздел 59.2. Пример использования класса StringWriter.....	357
Глава 60. Класс Preferences	357
Раздел 60.1. Использование настроек (preferences).....	357
Раздел 60.2. Добавление слушателей событий	358
Раздел 60.3. Получение подузлов Preferences.....	359
Раздел 60.4. Координация доступа к настройкам в нескольких экземплярах приложений.....	360
Раздел 60.5. Экспорт настроек	360
Раздел 60.6. Импорт настроек.....	361
Раздел 60.7. Удаление слушателей событий	362
Раздел 60.8. Получение значений настроек.....	363
Раздел 60.9. Установка значений настроек.....	363
Глава 61. Фабричные методы для коллекций	363
Раздел 61.1. Примеры фабричного метода List<E>	364
Раздел 61.2. Примеры фабричного метода Set<E>.....	364
Раздел 61.3. Примеры фабричного метода Map<K, V>.....	364
Глава 62. Альтернативные коллекции	364
Раздел 62.1. Мультикарта (multimap) в коллекциях Guava, Apache и Eclipse.....	364
Раздел 62.2. Apache HashBag, Guava HashMultiset и Eclipse HashBag	368
Раздел 62.3. Сравнение операций с коллекциями — создание коллекций	370
Глава 63. Коллекции с параллельным доступом (Concurrent Collections).....	375
Раздел 63.1. Потокобезопасные коллекции	375
Раздел 63.2. Операция “вставка” в коллекции ConcurrentHashMap.....	375
Раздел 63.3. Concurrent Collections	376
Глава 64. Выбор коллекций	377
Раздел 64.1. Блок-схема коллекций Java.....	377
Глава 65. Ключевое слово super	379
Раздел 65.1. Использование ключевого слова super на примерах.....	379

Глава 66. Сериализация	381
Раздел 66.1. Сериализация в Java	381
Раздел 66.2. Пользовательская сериализация.....	383
Раздел 66.3. Версионирование и уникальный идентификатор serialVersionUID.....	385
Раздел 66.4. Сериализация с помощью библиотеки Gson	388
Раздел 66.5. Пользовательская десериализация формата JSON с помощью библиотеки Jackson.....	388
Глава 67. Класс Optional	390
Раздел 67.1. Метод map	390
Раздел 67.2. Возврат значения по умолчанию, если Optional пуст	391
Раздел 67.3. Выбросить исключение, если нет значения.....	392
Раздел 67.4. Предоставление значения по умолчанию с помощью Supplier	392
Раздел 67.5. Фильтр.....	393
Раздел 67.6. Использование контейнеров Optional для примитивных типов данных.....	393
Раздел 67.7. Выполнять код только при наличии значения.....	394
Раздел 67.8. FlatMap	394
Глава 68. Ссылки на объекты	395
Раздел 68.1. Использование ссылок на объекты в качестве параметров для метода	395
Глава 69. Исключения и обработка исключений	397
Раздел 69.1. Перехват исключения с помощью оператора try-catch	397
Раздел 69.2. Оператор try-with-resources.....	399
Раздел 69.3. Разработка собственного класса исключений	402
Раздел 69.4. Обработка исключений InterruptedException.....	404
Раздел 69.5. Операторы возврата (return) в блоке try catch.....	405
Раздел 69.6. Введение.....	406
Раздел 69.7. Иерархия исключений Java — непроверяемые и проверяемые исключения.....	408
Раздел 69.8. Создание и чтение трассировки стека вызовов (stacktrace)	411
Раздел 69.9. Как выбрасывать исключения	414
Раздел 69.10. Дополнительные возможности при использовании исключений.....	416
Раздел 69.11. Блоки try-finally и try-catch-finally	418
Раздел 69.12. Ключевое слово ‘throws’ в объявлении метода.....	419
Глава 70. Календарь и его подклассы	420
Раздел 70.1. Создание объектов календаря.....	420
Раздел 70.2. Увеличение / уменьшение значений в полях класса Calendar.....	421
Раздел 70.3. Вычитание календарей.....	421
Раздел 70.4. Определение AM/PM	421
Глава 71. Использование ключевого слова static	422
Раздел 71.1. Ссылка на нестатический член из статического окружения.....	422
Раздел 71.2. Использование модификатора static для объявления констант.....	422
Глава 72. Класс Properties	423
Раздел 72.1. Загрузочные свойства.....	423
Раздел 72.2. Сохранение свойств в формате XML.....	424
Раздел 72.3. Осторожность при работе с файлами свойств: пробелы в конце строки.....	425
Глава 73. Лямбда-выражения	427
Раздел 73.1. Введение в лямбда-выражения Java.....	427
Раздел 73.2. Использование лямбда-выражений для сортировки коллекции	431
Раздел 73.3. Ссылки на методы	432
Раздел 73.4. Реализация нескольких интерфейсов.....	434
Раздел 73.5. Пример реализации слушателя с помощью лямбда-выражения.....	434
Раздел 73.6. Замыкания в Java с использованием лямбда-выражений	435
Раздел 73.7. Лямбда-выражения и использование памяти	436
Раздел 73.8. Использование лямбда-выражений с собственным функциональным интерфейсом.....	437
Раздел 73.9. От традиционного стиля к стилю с использованием лямбда-выражений.....	438
Раздел 73.10. Оператор “return” возвращает результат только из лямбда-выражения, а не из внешнего метода	439
Раздел 73.11. Лямбда-выражения и паттерн Execute-Around	440
Раздел 73.12. Использование лямбда-выражений и предикатов (условий) для получения из списка определенного значения (значений)	441

Глава 74. Основные управляющие конструкции	442
Раздел 74.1. Конструкция switch.....	442
Раздел 74.2. Цикл do..while.....	444
Раздел 74.3. Цикл For Each	444
Раздел 74.4. Оператор Continue в Java.....	445
Раздел 74.5. Операторы ветвления If / Else If / Else.....	445
Раздел 74.6. Циклы For	446
Раздел 74.7. Тернарный оператор	447
Раздел 74.8. Конструкция Try ... Catch ... Finally.....	447
Раздел 74.9. Оператор break.....	448
Раздел 74.10. Циклы While	448
Раздел 74.11. If / Else.....	449
Раздел 74.12. Вложенные операторы break / continue	449
Глава 75. Класс BufferedWriter	449
Раздел 75.1. Запись строки текста в файл.....	449
Глава 76. Новые способы чтения/записи файлов	450
Раздел 76.1. Создание путей	450
Раздел 76.2. Управление путями	450
Раздел 76.3. Получение информации о пути.....	451
Раздел 76.4. Получение информации о файловой системе.....	451
Раздел 76.5. Чтение файлов.....	452
Раздел 76.6. Запись файлов.....	452
Глава 77. Работа с файлами с помощью API Java I/O	453
Раздел 77.1. Переход от java.io.File к Java 7 NIO (java.nio.file.Path)	453
Раздел 77.2. Чтение изображения из файла.....	456
Раздел 77.3. Чтение/запись файлов с помощью потоков FileInputStream/FileOutputStream	456
Раздел 77.4. Чтение последовательности байтов и ее преобразование в массив byte[]	457
Раздел 77.5. Копирование файла с помощью интерфейса Channel	457
Раздел 77.6. Запись массива байтов byte[] в файл	458
Раздел 77.7. Использование потоков в сравнении с использованием API Writer/Reader	459
Раздел 77.8. Чтение файла с помощью класса Scanner	460
Раздел 77.9. Копирование файла с помощью потоков InputStream и OutputStream	461
Раздел 77.10. Чтение из двоичного файла	461
Раздел 77.11. Чтение файла с использованием канала (класс Channel) и буфера (класс Buffer).....	462
Раздел 77.12. Создание директории	463
Раздел 77.13. Блокировка или перенаправление стандартного потока ошибок	463
Раздел 77.14. Чтение всего файла за один раз.....	464
Раздел 77.15. Блокировка.....	464
Раздел 77.16. Чтение файла с помощью BufferedInputStream	465
Раздел 77.17. Итерация по каталогу с целью вывода на экран всех подкаталогов, которые в нем содержатся.....	466
Раздел 77.18. Запись файла с использованием канала (класс Channel) и буфера (класс Buffer)	466
Раздел 77.19. Запись файла с помощью PrintStream.....	466
Раздел 77.20. Обход файлов в каталоге с фильтрацией имен файлов по расширению.....	467
Раздел 77.21. Доступ к содержимому ZIP-файла	467
Глава 78. Класс Scanner.....	468
Раздел 78.1. Общий паттерн для решения задач, связанных с использованием ввода с помощью класса Scanner	468
Раздел 78.2. Использование собственных разделителей.....	470
Раздел 78.3. Чтение системного ввода с помощью объекта класса Scanner	471
Раздел 78.4. Чтение входного файла с помощью объекта класса Scanner	471
Раздел 78.5. Чтение всего содержимого файла в виде одной строки с помощью класса Scanner	472
Раздел 78.6. Аккуратность при закрытии объекта класса Scanner	472
Раздел 78.7. Чтение значения типа int из командной строки	472
Глава 79. Интерфейсы.....	473
Раздел 79.1. Реализация нескольких интерфейсов.....	473
Раздел 79.2. Объявление и реализация интерфейса	474

Раздел 79.3. Расширение (наследование) интерфейса	475
Раздел 79.4. Полезность интерфейсов	475
Раздел 79.5. Методы по умолчанию	477
Раздел 79.6. Модификаторы в интерфейсах	479
Раздел 79.7. Использование интерфейсов с обобщенными типами данных (дженериками)	480
Раздел 79.8. Усиление параметров ограничения типа	482
Раздел 79.9. Реализация интерфейсов в абстрактном классе	483
Глава 80. Регулярные выражения	484
Раздел 80.1. Использование захватывающих групп	484
Раздел 80.2. Использование регулярного выражения с собственным поведением путем компиляции шаблона с флагами	485
Раздел 80.3. Управляющие символы	485
Раздел 80.4. Проверка на неравенство заданной строке	486
Раздел 80.5. Сравнение с литералом регулярного выражения	486
Раздел 80.6. Использование в регулярных выражениях символа обратного слепа	487
Глава 81. Интерфейсы Comparable и Comparator	488
Раздел 81.1. Сортировка списка с помощью Comparable<T> или Comparator<T>	488
Раздел 81.2. Методы compareTo и compare	491
Раздел 81.3. Естественная сортировка (на основе использования compareTo) и явная сортировка (на основе использования compare)	492
Раздел 81.4. Создание компаратора (Comparator), использующего метод сравнения	493
Раздел 81.5. Сортировка элементов словаря	494
Глава 82. Операции с плавающей запятой в Java	494
Раздел 82.1. Сравнение чисел с плавающей точкой	494
Раздел 82.2. Понятия переполнения (Overflow) и недостижимости (Underflow)	497
Раздел 82.3. Форматирование значений с плавающей точкой	497
Раздел 82.4. Строгое соблюдение спецификации IEEE	498
Глава 83. Валюта и деньги	499
Раздел 83.1. Добавление пользовательской валюты	499
Глава 84. Клонирование объектов	500
Раздел 84.1. Клонирование, выполняющее глубокое копирование	500
Раздел 84.2. Клонирование с использованием фабричного копирования	501
Раздел 84.3. Клонирование с использованием конструктора копирования	501
Раздел 84.4. Клонирование с помощью реализации интерфейса Cloneable	502
Раздел 84.5. Клонирование с выполнением поверхностной копии	502
Глава 85. Рекурсия	503
Раздел 85.1. Основная идея рекурсии	503
Раздел 85.2. Проблемы, связанные с глубокой рекурсией в Java	504
Раздел 85.3. Типы рекурсии	505
Раздел 85.4. Вычисление N-го числа ряда Фибоначчи	506
Раздел 85.5. Исключение StackOverflowError и реализация рекурсии в виде цикла	506
Раздел 85.6. Вычисление N-ой степени числа	509
Раздел 85.7. Обход древовидной структуры данных с помощью рекурсии	509
Раздел 85.8. Как записать строку в обратном порядке с помощью рекурсии	510
Раздел 85.9. Вычисление суммы целых чисел от 1 до N	510
Глава 86. Преобразование объектов в строки и наоборот	511
Раздел 86.1. Преобразование типа String в другие типы данных	511
Раздел 86.2. Преобразование в/из последовательности байтов	512
Раздел 86.3. Кодирование / декодирование последовательностей в кодировке Base64	513
Раздел 86.4. Преобразование других типов данных в объект String	514
Раздел 86.5. Получение объекта String на основе потока InputStream	514
Глава 87. Генерация случайных чисел	515
Раздел 87.1. Псевдослучайные числа	515
Раздел 87.2. Псевдослучайные числа из определенного диапазона	515
Раздел 87.3. Генерация криптографически безопасных псевдослучайных чисел	516
Раздел 87.4. Генерация случайных чисел с заданным начальным числом	516
Раздел 87.5. Создание набора случайных чисел без дубликатов	517
Раздел 87.6. Генерация случайных чисел с использованием пакета apache-common lang3	518

Глава 88. Паттерн проектирования Синглтон (Singleton)	519
Раздел 88.1. Синглтон на основе перечисления Enum	519
Раздел 88.2. Синглтон без использования перечисления Enum (нетерпеливая инициализация)	519
Раздел 88.3. Потокобезопасная “ленивая” инициализация с помощью класса-держателя (holder-класса) Реализация синглтона по способу, предложенному Биллом Пью (Bill Pugh)	520
Раздел 88.4. Потокобезопасный синглтон, в котором реализована блокировка с двойной проверкой	520
Раздел 88.5. Расширение синглтона (наследование синглтона)	521
Глава 89. Автоупаковка	524
Раздел 89.1. Взаимозаменяемое использование типов int и Integer	524
Раздел 89.2. Автоупаковка может привести к возникновению исключения NullPointerException ..	525
Раздел 89.3. Использование значений типа Boolean в операторе if	526
Раздел 89.4. Различные случаи, когда типы Integer и int могут взаимозаменять друг друга	526
Раздел 89.5. Перерасход памяти и вычислительных ресурсов при автоупаковке	527
Глава 90. Двумерная графика в Java	528
Раздел 90.1. Пример 1: Рисование и заливка прямоугольника с помощью Java	528
Раздел 90.2. Пример 2: Рисование и заливка цветом овала	530
Глава 91. JAXB	531
Раздел 91.1. Чтение XML файла (unmarshalling)	531
Раздел 91.2. Запись XML файла (маршаллинг объекта)	532
Раздел 91.3. Ручная настройка в XML сопоставления “поле/свойство”	532
Раздел 91.4. Привязка пространства имен XML к сериализуемому классу Java	533
Раздел 91.5. Использование адаптера XmlAdapter для генерации нужного формата xml	534
Раздел 91.6. Использование XmlAdapter для обрезки строки	535
Раздел 91.7. Автоматическая настройка XML-сопоставления поля/свойства (@XmlAccessorType)	536
Раздел 91.8. Определение экземпляра XmlAdapter для (повторного) использования существующих данных	537
Глава 92. Класс — Отражение (Reflection) в Java	540
Раздел 92.1. Метод getClass() класса Object	540
Глава 93. Сетевое взаимодействие	541
Раздел 93.1. Базовая связь клиента и сервера с использованием сокета	541
Раздел 93.2. Основы взаимодействия клиента и сервера с использованием UDP (Datagram)	543
Раздел 93.3. Загрузка хранилищ TrustStore и KeyStore из потока InputStream	544
Раздел 93.4. Пример для сокета — чтение веб-страницы с помощью простого сокета	545
Раздел 93.5. Временное отключение проверки сертификата SSL (в целях тестирования)	546
Раздел 93.6. Загрузка файла с помощью канала	547
Раздел 93.7. Многоадресная рассылка (Multicasting)	547
Глава 94. Пакет NIO и сетевое взаимодействие	550
Раздел 94.1. Использование селектора для ожидания событий (пример с OP_CONNECT)	550
Глава 95. Класс HttpURLConnection	551
Раздел 95.1. Получение тела ответа из URL в виде строки	551
Раздел 95.2. Метод POST (отправка данных)	552
Раздел 95.3. Удаление ресурса	553
Раздел 95.4. Проверка наличия ресурса	554
Глава 96. Прикладной программный интерфейс JAX-WS	554
Раздел 96.1. Базовая аутентификация	554
Глава 97. Движок JavaScript Nashorn	555
Раздел 97.1. Запуск на выполнение файла JavaScript	555
Раздел 97.2. Перехват стандартного вывода сценария	555
Раздел 97.3. Привет Nashorn	556
Раздел 97.4. Вычисление арифметических выражений, содержащихся в строках	556
Раздел 97.5. Определение глобальных переменных	557
Раздел 97.6. Определить и получить значение глобальных переменных	557
Раздел 97.7. Использование объектов Java в языке JavaScript на движке Nashorn	557
Раздел 97.8. Реализация интерфейса на основе сценария	558

Глава 98. Java Native Interface	559
Раздел 98.1. Вызов функций языка C++ из Java	559
Раздел 98.2. Вызов методов Java из C++ (обратный вызов).....	561
Раздел 98.3. Загрузка нативных библиотек	562
Глава 99. Функциональные интерфейсы	563
Раздел 99.1. Список стандартных функциональных интерфейсов библиотеки Java Runtime Library (по сигнатурам).....	563
Глава 100. Текущий интерфейс	564
Раздел 100.1. Текущий стиль программирования	564
Раздел 100.2. Истина заключается в Fluent Testing Framework	566
Глава 101. Программный интерфейс вызова удаленных методов (RMI)	566
Раздел 101.1. Обратный вызов: вызов методов “клиента”	566
Раздел 101.2. Пример простой реализации клиента и сервера с использованием RMI	571
Раздел 101.3. Клиент-сервер: вызов методов для одной JVM из другой	573
Глава 102. Интерфейсы Iterator и Iterable	575
Раздел 102.1. Удаление элементов с помощью итератора.....	575
Раздел 102.2. Создание собственного интерфейса Iterable.....	576
Раздел 102.3. Использование Iterable в цикле for	577
Раздел 102.4. Использование итератора напрямую.....	577
Глава 103. API Reflection	578
Раздел 103.1. Динамические прокси	578
Раздел 103.2. Введение.....	579
Раздел 103.3. Хакинг с использованием Reflection	581
Раздел 103.4. Неправильное использование API Reflection для изменения переменных с модификатором доступа private и final	583
Раздел 103.5. Получение значений и установка полей	584
Раздел 103.6. Вызов конструктора	585
Раздел 103.7. Вызов конструктора вложенного класса.....	586
Раздел 103.8. Вызов метода	586
Раздел 103.9. Получение класса по его полному имени.....	587
Раздел 103.10. Получение констант перечисления.....	587
Раздел 103.11. Вызов перегруженных конструкторов с использованием рефлексии.....	588
Глава 104. ByteBuffer	589
Раздел 104.1. Базовое использование — использование в виде DirectByteBuffer	589
Раздел 104.2. Базовое использование — создание ByteBuffer	590
Раздел 104.3. Базовое использование — запись данных в буфер	590
Глава 105. Апплеты	591
Раздел 105.1. Минимальный апплет.....	591
Раздел 105.2. Создание графического интерфейса пользователя.....	591
Раздел 105.3. Как открыть ссылку с помощью апплета	592
Раздел 105.4. Загрузка изображений, аудио и других ресурсов	593
Глава 106. Выражения	594
Раздел 106.1. Приоритет операторов	594
Раздел 106.2. Выражения (основы)	595
Раздел 106.3. Порядок вычисления выражения.....	597
Раздел 106.4. Константные выражения	597
Глава 107. JSON в Java	599
Раздел 107.1. Использование библиотеки Jackson Object Mapper.....	599
Раздел 107.2. Преобразовать JSON в объект Object (библиотека Gson).....	600
Раздел 107.3. JSONObject.NULL	600
Раздел 107.4. JSON Builder — цепочка методов	601
Раздел 107.5. Преобразование объекта в JSON (библиотека Gson)	601
Раздел 107.6. Обход элементов JSON.....	601
Раздел 107.7. Методы optXXX и getXXX	602
Раздел 107.8. Извлечение одного элемента из JSON	602
Раздел 107.9. Преобразование массива JSONArray в Java List (библиотека Gson)	602

Раздел 107.10. Преобразование данных в формат JSON.....	603
Раздел 107.11. Декодирование данных из JSON	604
Глава 108. Парсинг XML с использованием API JAXP.....	604
Раздел 108.1. Парсинг документа с использованием StAX API	604
Раздел 108.2. Парсинг и навигация по документу с использованием DOM API	606
Глава 109. Запросы XPath для документов XML	607
Раздел 109.1. Парсинг нескольких запросов XPath в одном документе XML	607
Раздел 109.2. Повторный парсинг результатов одного и того же запроса XPath в XML.....	608
Раздел 109.3. Получение списка узлов NodeList из документа XML	609
Глава 110. ХОМ — объектная модель XML	609
Раздел 110.1. Чтение XML-файла	609
Раздел 110.2. Запись в файл XML.....	612
Глава 111. Полиморфизм.....	615
Раздел 111.1. Переопределение методов (overriding)	615
Раздел 111.2. Перегрузка методов (overloading).....	616
Раздел 111.3. Полиморфизм и различные типы переопределения	617
Раздел 111.4. Виртуальные функции.....	621
Раздел 111.5. Изменение поведения путем добавления классов, не внося изменения в существующий код	622
Глава 112. Инкапсуляция.....	623
Раздел 112.1. Инкапсуляция для поддержания стабильности	623
Раздел 112.2. Инкапсуляция для уменьшения связности.....	624
Глава 113. Java-агенты	625
Раздел 113.1. Модификация классов с помощью агентов	625
Раздел 113.2. Добавление агента во время выполнения программы	626
Раздел 113.3. Настройка базового агента	627
Глава 114. Изменяющийся список аргументов Varargs (Variable Argument).....	627
Раздел 114.1. Работа с параметрами Varargs.....	627
Раздел 114.2. Объявление параметра Varargs	628
Глава 115. Ведение журнала (java.util.logging).....	628
Раздел 115.1. Логирование сложных сообщений (эффективно)	628
Раздел 115.2. Использование логирования по умолчанию	630
Раздел 115.3. Уровни логирования.....	631
Глава 116. log4j / log4j2.....	632
Раздел 116.1. Создание файла с настройками для работы с БД	632
Раздел 116.2. Как получить Log4j	632
Раздел 116.3. Настройка файла свойств.....	634
Раздел 116.4. Основной файл конфигурации log4j2.xml.....	634
Раздел 116.5. Как использовать Log4j в Java-коде	635
Раздел 116.6. Переход с log4j 1.x на 2.x.....	635
Раздел 116.7. Фильтр вывода журнала по уровню (log4j 1.x).....	636
Глава 117. Официальный стандарт Oracle для кода, написанного на языке Java.....	637
Раздел 117.1. Условные обозначения.....	637
Раздел 117.2. Структура класса.....	638
Раздел 117.3. Аннотации.....	639
Раздел 117.4. Объявления с ключевым словом import.....	640
Раздел 117.5. Скобки.....	641
Раздел 117.6. Избыточные группирующие круглые скобки.....	641
Раздел 117.7. Модификаторы	642
Раздел 117.8. Отступы	642
Раздел 117.9. Литералы	643
Раздел 117.10. Объявление пакета.....	643
Раздел 117.11. Лямбда-выражения.....	643
Раздел 117.12. Исходные файлы Java	644
Раздел 117.13. Перенос операторов.....	644
Раздел 117.14. Разнесение по нескольким строкам объявлений методов	645
Раздел 117.15. Перенос выражений.....	645