

Java 7



- Принципы объектно-ориентированного программирования
- Пакеты классов и интерфейсы Java SE 7
- Библиотеки Swing и Java 2D, NIO2
- Апплеты, графика, звук и печать
- Технологии Servlet, JSP, JSTL, JSF, XML
- Около 200 законченных программ

**Наиболее
полное
руководство**

В ПОДЛИННИКЕ®

Ильдар Хабибуллин

Java 7

Санкт-Петербург
«БХВ-Петербург»
2012

УДК 681.3.06
ББК 32.973.26-018.2
X12

Хабибуллин И. Ш.

X12 Java 7. — СПб.: БХВ-Петербург, 2012. — 768 с.: ил. — (В подлиннике)
ISBN 978-5-9775-0735-6

Рассмотрено все необходимое для разработки, компиляции, отладки и запуска приложений Java. Изложены практические приемы использования как традиционных, так и новейших конструкций объектно-ориентированного языка Java, графической библиотеки классов Swing, расширенной библиотеки Java 2D, работа со звуком, печать, способы русификации программ. Приведено полное описание нововведений Java SE 7: двоичная запись чисел, строковые варианты разветвлений, "ромбовидный оператор", NIO2, новые средства многопоточности и др. Дано подробное изложение последней версии сервлетов, технологии JSP и библиотек тегов JSTL. Около двухсот законченных программ иллюстрируют рассмотренные приемы программирования. Приведена подробная справочная информация о классах и методах Core Java API.

Для программистов

УДК 681.3.06
ББК 32.973.26-018.2

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Игорь Шишигин</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Екатерина Капалыгина</i>
Компьютерная верстка	<i>Ольги Сергиенко</i>
Корректор	<i>Зинаида Дмитриева</i>
Дизайн серии	<i>Инны Тачиной</i>
Оформление обложки	<i>Елены Беляевой</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 31.08.11.
Формат 70×100¹/₁₆. Печать офсетная. Усл. печ. л. 61,92.
Тираж 1800 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.005770.05.09 от 26.05.2009 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0735-6

© Хабибуллин И. Ш., 2011
© Оформление, издательство "БХВ-Петербург", 2011

Оглавление

Введение.....	19
Что такое Java?	20
Структура книги.....	21
Выполнение Java-программы	24
Что такое JDK?	25
Что такое JRE?	27
Как установить JDK?	27
Как использовать JDK?	28
Интегрированные среды Java.....	30
Особая позиция Microsoft.....	30
Java в Интернете	31
Литература по Java.....	32
Благодарности	33
ЧАСТЬ I. БАЗОВЫЕ КОНСТРУКЦИИ ЯЗЫКА JAVA	35
Глава 1. Встроенные типы данных, операции над ними	37
Первая программа на Java	37
Комментарии	40
Аннотации	42
Константы.....	42
Целые	42
Действительные.....	43
Символы.....	43
Строки	44
Имена	45
Примитивные типы данных и операции	45
Логический тип	47
Логические операции.....	47
Упражнения	48
Целые типы.....	48
Операции над целыми типами	49
Арифметические операции	49
Приведение типов	50

Операции сравнения	52
Побитовые операции	52
Сдвиги	53
Упражнения	54
Вещественные типы	54
Операции присваивания	55
Упражнения	56
Условная операция	56
Упражнения	56
Выражения	56
Приоритет операций	57
Операторы	58
Блок	59
Операторы присваивания	59
Условный оператор	59
Упражнения	61
Операторы цикла	62
Оператор <i>continue</i> и метки	64
Оператор <i>break</i>	65
Упражнения	65
Оператор варианта	65
Массивы	67
Многомерные массивы	69
Заключение	71
Вопросы для самопроверки	71
Глава 2. Объектно-ориентированное программирование в Java	73
Парадигмы программирования	73
Принципы объектно-ориентированного программирования	76
Абстракция	76
Иерархия	79
Ответственность	80
Модульность	81
Принцип KISS	83
Упражнения	84
Как описать класс и подкласс?	84
Передача аргументов в метод	86
Перегрузка методов	87
Переопределение методов	88
Реализация полиморфизма в Java	89
Упражнения	90
Абстрактные методы и классы	90
Окончательные члены и классы	91
Класс <i>Object</i>	92
Конструкторы класса	93
Операция <i>new</i>	94
Упражнение	94
Статические члены класса	94
Класс <i>Complex</i>	96

Метод <i>main()</i>	99
Методы с переменным числом аргументов.....	100
Где видны переменные.....	101
Вложенные классы.....	103
Отношения "быть частью" и "являться".....	107
Заключение.....	108
Вопросы для самопроверки.....	108
Глава 3. Пакеты, интерфейсы и перечисления.....	109
Пакет и подпакет.....	110
Права доступа к членам класса.....	111
Размещение пакетов по файлам.....	113
Импорт классов и пакетов.....	115
Java-файлы.....	116
Интерфейсы.....	117
Перечисления.....	121
Объявление аннотаций.....	124
Design patterns.....	126
Схема проектирования MVC.....	126
Шаблон Singleton.....	127
Заключение.....	129
Вопросы для самопроверки.....	129
ЧАСТЬ II. ИСПОЛЬЗОВАНИЕ КЛАССОВ ИЗ JAVA API.....	131
Глава 4. Классы-оболочки и generics.....	133
Числовые классы.....	134
Автоматическая упаковка и распаковка типов.....	136
Настраиваемые типы (generics).....	137
Шаблон типа (wildcard type).....	140
Настраиваемые методы.....	141
Класс <i>Boolean</i>	142
Класс <i>Character</i>	143
Класс <i>BigInteger</i>	146
Класс <i>BigDecimal</i>	148
Класс <i>Class</i>	152
Вопросы для самопроверки.....	155
Глава 5. Работа со строками.....	156
Класс <i>String</i>	157
Как создать строку.....	157
Упражнение.....	162
Сцепление строк.....	162
Как узнать длину строки.....	162
Как выбрать символы из строки.....	163
Как выбрать подстроку.....	163
Как разбить строку на подстроки.....	164
Как сравнить строки.....	164
Как найти символ в строке.....	166

Как найти подстроку	167
Как изменить регистр букв.....	167
Как заменить отдельный символ	168
Как заменить подстроку	168
Как убрать пробелы в начале и конце строки.....	168
Как преобразовать в строку данные другого типа	168
Упражнения	169
Класс <i>StringBuilder</i>	169
Конструкторы	169
Как добавить подстроку	170
Как вставить подстроку	170
Как удалить подстроку	171
Как удалить символ.....	171
Как заменить подстроку	171
Как перевернуть строку.....	171
Синтаксический разбор строки.....	172
Класс <i>StringTokenizer</i>	172
Заключение	173
Вопросы для самопроверки	173
Глава 6. Классы-коллекции.....	174
Класс <i>Vector</i>	174
Как создать вектор	175
Как добавить элемент в вектор	175
Как заменить элемент	176
Как узнать размер вектора.....	176
Как обратиться к элементу вектора	176
Как узнать, есть ли элемент в векторе.....	176
Как узнать индекс элемента	177
Как удалить элементы.....	177
Класс <i>Stack</i>	178
Класс <i>Hashtable</i>	179
Как создать таблицу <i>Hashtable</i>	180
Как заполнить таблицу <i>Hashtable</i>	180
Как получить значение по ключу	180
Как узнать наличие ключа или значения	181
Как получить все элементы таблицы <i>Hashtable</i>	181
Как удалить элементы.....	181
Класс <i>Properties</i>	182
Интерфейс <i>Collection</i>	185
Интерфейс <i>List</i>	185
Интерфейс <i>Set</i>	186
Интерфейс <i>SortedSet</i>	186
Интерфейс <i>NavigableSet</i>	187
Интерфейс <i>Queue</i>	188
Интерфейс <i>BlockingQueue</i>	188
Интерфейс <i>Deque</i>	188
Интерфейс <i>BlockingDeque</i>	189

Интерфейс <i>Map</i>	190
Вложенный интерфейс <i>Map.Entry</i>	191
Интерфейс <i>SortedMap</i>	191
Интерфейс <i>NavigableMap</i>	191
Абстрактные классы-коллекции	192
Интерфейс <i>Iterator</i>	193
Интерфейс <i>ListIterator</i>	194
Классы, создающие списки	195
Двунаправленный список	196
Дек	196
Упражнение	197
Классы, создающие отображения	197
Связанные отображения	197
Упорядоченные отображения	197
Сравнение элементов коллекций	198
Упражнение	199
Классы, создающие множества	199
Связанные множества	199
Упорядоченные множества	200
Действия с коллекциями	200
Методы класса <i>Collections</i>	200
Упражнение	201
Заключение	202
Вопросы для самопроверки	202
Глава 7. Классы-утилиты	203
Работа с массивами	203
Сортировка массива	203
Бинарный поиск в массиве	203
Заполнение массива	204
Копирование массива	204
Сравнение массивов	205
Представление массива строкой	205
Получение хеш-кода массива	206
Локальные установки	206
Работа с датами и временем	208
Часовой пояс и летнее время	208
Класс <i>Calendar</i>	209
Подкласс <i>GregorianCalendar</i>	209
Представление даты и времени	210
Получение случайных чисел	211
Копирование массивов	211
Взаимодействие с системой	212

ЧАСТЬ III. СОЗДАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ И АППЛЕТОВ

Глава 8. Принципы построения графического интерфейса	215
Компонент и контейнер	217
Иерархия классов AWT	220

Окно библиотеки Swing	221
Использование системных приложений	222
System Tray	223
Splash Screen	224
Заключение	224
Вопросы для самопроверки	224
Глава 9. Графические примитивы	226
Методы класса <i>Graphics</i>	226
Как задать цвет	226
Упражнение	228
Как нарисовать чертеж	228
Класс <i>Polygon</i>	229
Упражнение	230
Прочие методы класса <i>Graphics</i>	230
Как вывести текст	231
Как установить шрифт	231
Как задать шрифт	231
Класс <i>FontMetrics</i>	235
Упражнение	238
Возможности Java 2D	238
Преобразование координат	240
Класс <i>AffineTransform</i>	240
Упражнение	243
Рисование фигур средствами Java 2D	243
Класс <i>BasicStroke</i>	243
Класс <i>GeneralPath</i>	246
Классы <i>GradientPaint</i> и <i>TexturePaint</i>	247
Классы <i>LinearGradientPaint</i> и <i>RadialGradientPaint</i>	249
Вывод текста средствами Java 2D	250
Методы улучшения визуализации	254
Упражнение	256
Заклучение	256
Вопросы для самопроверки	256
Глава 10. Основные компоненты AWT	257
Класс <i>Component</i>	257
Класс <i>Cursor</i>	259
Как создать свой курсор	259
Упражнение	260
События	260
Класс <i>Container</i>	261
События	262
Текстовая метка <i>Label</i>	262
События	262
Кнопка <i>Button</i>	262
События	263
Кнопка выбора <i>Checkbox</i>	263
События	263

Класс <i>CheckboxGroup</i>	263
Как создать группу радиокнопок	264
Раскрывающийся список <i>Choice</i>	265
События	266
Список <i>List</i>	266
События	267
Компоненты для ввода текста	268
Класс <i>TextComponent</i>	268
События	269
Строка ввода <i>TextField</i>	269
События	269
Поле ввода <i>TextArea</i>	269
События	270
Линейка прокрутки <i>Scrollbar</i>	272
События	272
Контейнер <i>Panel</i>	274
Контейнер <i>ScrollPane</i>	275
Контейнер <i>Window</i>	276
События	276
Контейнер <i>Frame</i>	277
События	277
Контейнер <i>Dialog</i>	279
События	280
Контейнер <i>FileDialog</i>	282
События	282
Создание собственных компонентов	283
Компонент <i>Canvas</i>	283
Создание "легкого" компонента	285
Упражнение	287
Создание меню	287
Всплывающее меню	292
Вопросы для самопроверки	295
Глава 11. Оформление GUI компонентами Swing	296
Состав библиотеки Swing	297
Основные компоненты Swing	299
Компонент <i>JComponent</i>	299
Схема MVC в компонентах Swing	300
Надпись <i>JLabel</i>	302
Кнопки	304
Кнопка <i>JButton</i>	306
Кнопка выбора <i>JToggleButton</i>	306
Кнопка выбора <i>JCheckBox</i>	308
Радиокнопка <i>JRadioButton</i>	308
Упражнение	309
Раскрывающийся список <i>JComboBox</i>	310
Список выбора <i>JList</i>	311
Визуализация элементов списков	312

Упражнение	314
Счетчик <i>JSpinner</i>	314
Полосы прокрутки <i>JScrollBar</i>	316
Ползунок <i>JSlider</i>	316
Упражнение	318
Индикатор <i>JProgressBar</i>	318
Дерево объектов <i>JTree</i>	318
Построение меню средствами Swing	322
Строка меню <i>JMenuBar</i>	322
Меню <i>JMenu</i>	323
Пункт меню <i>JMenuItem</i>	323
Всплывающее меню <i>JPopupMenu</i>	325
Панель выбора цвета <i>JColorChooser</i>	326
Упражнение	328
Окно выбора файла <i>JFileChooser</i>	328
Фильтр файлов <i>FileFilter</i>	328
Как получить выбранный файл	330
Дополнительный компонент	330
Замена изображений	331
Русификация Swing	333
Вопросы для самопроверки	333
Глава 12. Текстовые компоненты	334
Компонент <i>JTextComponent</i>	334
Модель данных — документ	334
Строка символов <i>Segment</i>	335
Запись текста в документ	336
Атрибуты текста	336
Удаление текста из документа	337
Фильтрация документа	337
Внесение структуры в документ	337
События в документе	338
Реализации документа	338
Установка модели данных	339
Вид	339
Контроллер — редактор текста	341
Курсор	341
Ограничение перемещения курсора	342
Реализации редактора	343
Раскладка клавиатуры	343
Печать текста документа	344
Поле ввода <i>JTextField</i>	344
Поле ввода пароля <i>JPasswordField</i>	347
Редактор объектов <i>JFormattedTextField</i>	347
Область ввода <i>JTextArea</i>	348
Текстовый редактор <i>JEditorPane</i>	349
Редактор <i>JTextPane</i>	350
Вопросы для самопроверки	350

Глава 13. Таблицы	351
Класс <i>JTable</i>	351
Модель данных таблицы	353
Модель ячеек таблицы.....	353
Свойства столбца таблицы <i>TableColumn</i>	358
Модель столбцов таблицы.....	358
Заголовки столбцов таблицы <i>JTableHeader</i>	358
Модель выделения ячеек	360
Визуализация ячеек таблицы	361
Редактор ячеек таблицы	364
Сортировка строк таблицы.....	367
Фильтрация строк таблицы	369
Печать таблицы	370
Вопросы для самопроверки	371
Глава 14. Размещение компонентов и контейнеры Swing.....	372
Менеджер <i>FlowLayout</i>	372
Менеджер <i>BorderLayout</i>	374
Менеджер <i>GridLayout</i>	376
Менеджер <i>CardLayout</i>	377
Менеджер <i>GridBagLayout</i>	379
Контейнеры Swing	381
Панель <i>JPanel</i>	381
Панель прокрутки <i>JScrollPane</i>	382
Двойная панель <i>JSplitPane</i>	384
Панель с вкладками <i>JTabbedPane</i>	385
Линейная панель <i>Box</i>	387
Менеджер размещения <i>BoxLayout</i>	387
Компоненты-заполнители	388
Менеджер размещения <i>SpringLayout</i>	389
Размеры <i>Spring</i>	390
Промежутки <i>Constraints</i>	391
Размещение компонентов.....	392
Панель инструментальных кнопок <i>JToolBar</i>	393
Интерфейс <i>Action</i>	395
Слоеная панель <i>JLayeredPane</i>	396
Корневая панель <i>JRootPane</i>	397
Окно <i>JWindow</i>	399
Диалоговое окно <i>JDialog</i>	400
Окно верхнего уровня <i>JFrame</i>	401
Внутреннее окно <i>JInternalFrame</i>	402
Рабочий стол <i>JDesktopPane</i>	404
Стандартные диалоги <i>JOptionPane</i>	405
Окно с индикатором <i>ProgressMonitor</i>	409
Заключение.....	410
Вопросы для самопроверки	411
Глава 15. Обработка событий	412
Самообработка событий.....	416
Обработка вложенным классом.....	417
Упражнение	418

Событие <i>ActionEvent</i>	418
Обработка действий мыши	419
Упражнение	422
Классы-адаптеры.....	422
Управление колесиком мыши.....	423
Обработка действий клавиатуры	424
Упражнение	425
Событие <i>TextEvent</i>	425
Событие изменения <i>ChangeEvent</i>	426
Обработка действий с окном	426
Событие <i>ComponentEvent</i>	427
Событие <i>ContainerEvent</i>	428
Событие <i>FocusEvent</i>	428
Событие <i>ItemEvent</i>	428
Событие <i>AdjustmentEvent</i>	429
Несколько слушателей одного источника	431
Диспетчеризация событий	432
Создание собственного события.....	434
Вопросы для самопроверки	435
Глава 16. Оформление рамок	436
Пустая рамка <i>EmptyBorder</i>	438
Прямолинейная рамка <i>LineBorder</i>	438
Объемная рамка <i>BevelBorder</i>	439
Закругленная объемная рамка <i>SoftBevelBorder</i>	439
Врезанная рамка <i>EtchedBorder</i>	440
Рамка с изображением <i>MatteBorder</i>	440
Рамки с надписями <i>TitledBorder</i>	441
Сдвоенные рамки <i>CompoundBorder</i>	444
Создание собственных рамок	445
Вопросы для самопроверки	450
Глава 17. Изменение внешнего вида компонента	451
Получение свойств L&F	453
Задание стандартного L&F.....	455
Дополнительные L&F	457
Смена всего L&F.....	457
Замена отдельных свойств L&F.....	459
Темы Java L&F	462
Вопросы для самопроверки	465
Глава 18. Апплеты	466
Упражнения	472
Передача параметров в апплет.....	472
Атрибуты тега <i><applet></i>	475
Сведения об окружении апплета	476
Упражнение	477
Изображение и звук в апплетах	477
Слежение за процессом загрузки.....	477
Класс <i>MediaTracker</i>	478

Упражнения	480
Защита от апплета	480
Апплеты в библиотеке Swing	481
Апплет <i>JApplet</i>	482
Упражнение	483
Заключение	484
Вопросы для самопроверки	484
Глава 19. Прочие свойства Swing	485
Свойства экземпляра компонента	485
Прокрутка содержимого компонента	486
Передача фокуса ввода	486
Перенос данных Drag and Drop	491
Временная задержка <i>Timer</i>	492
Глава 20. Изображения и звук	494
Модель "поставщик-потребитель"	494
Классы-фильтры	497
Как выделить фрагмент изображения	498
Как изменить цвет изображения	499
Как переставить пиксели изображения	500
Упражнения	501
Модель обработки прямым доступом	501
Преобразование изображения в Java 2D	504
Аффинное преобразование изображения	504
Изменение интенсивности изображения	507
Изменение составляющих цвета	508
Создание различных эффектов	509
Упражнения	510
Анимация	510
Улучшение изображения двойной буферизацией	512
Упражнения	516
Звук	516
Проигрывание звука в Java	517
Синтез и запись звука в Java	522
Упражнение	524
Вопросы для самопроверки	525

ЧАСТЬ IV. НЕОБХОДИМЫЕ КОНСТРУКЦИИ JAVA

Глава 21. Обработка исключительных ситуаций	529
Блоки перехвата исключения	530
Упражнения	533
Часть заголовка метода <i>throws</i>	533
Оператор <i>throw</i>	536
Обработка нескольких типов исключений с помощью иерархии	536
Иерархия классов-исключений	537
Порядок обработки исключений	538
Упражнение	538

Обработка нескольких типов исключений с помощью перечисления	539
Создание собственных исключений	539
Заключение	541
Вопросы для самопроверки	541
Глава 22. Подпроцессы	542
Класс <i>Thread</i>	545
Синхронизация подпроцессов	550
Согласование работы нескольких подпроцессов	552
Приоритеты подпроцессов	557
Подпроцессы-демоны	558
Группы подпроцессов	559
Заключение	559
Вопросы для самопроверки	559
Глава 23. Потоки ввода/вывода и печать	560
Консольный ввод/вывод	565
Форматированный вывод	568
Спецификации вывода целых чисел	569
Спецификации вывода вещественных чисел	570
Спецификация вывода символов	570
Спецификации вывода строк	570
Спецификации вывода логических значений	570
Спецификации вывода хеш-кода объекта	570
Спецификации вывода даты и времени	570
Класс <i>Console</i>	571
Упражнения	572
Файловый ввод/вывод	572
Получение свойств файла	574
Работа с файлом средствами NIO2	576
Буферизованный ввод/вывод	578
Каналы буферизованного ввода/вывода	579
Упражнения	581
Поток простых типов Java	582
Кодировка UTF-8	582
Класс <i>DataOutputStream</i>	582
Прямой доступ к файлу	584
Упражнение	585
Каналы обмена информацией	585
Сериализация объектов	587
Печать в Java	590
Печать средствами Java 2D	592
Печать файла	596
Печать страниц с разными параметрами	598
Вопросы для самопроверки	599
Глава 24. Сетевые средства Java	601
Работа в WWW	604
Упражнения	607

Работа по протоколу TCP	608
Работа с проху-сервером	611
Упражнения	612
Работа по протоколу UDP	612
Упражнение	614
Вопросы для самопроверки	614
ЧАСТЬ V. WEB-ТЕХНОЛОГИИ JAVA	617
Глава 25. Web-инструменты Java.....	619
Архиватор <i>jar</i>	619
Создание архива	620
Файл описания MANIFEST.MF	622
Файл INDEX.LIST.....	623
Компоненты JavaBeans.....	624
Связь с базами данных через JDBC.....	625
Вопросы для самопроверки	629
Глава 26. Сервлеты	631
Web-приложение.....	632
Интерфейс <i>Servlet</i>	633
Конфигурационный файл	634
Интерфейс <i>ServletConfig</i>	637
Контекст сервлета	639
Метод <i>Service</i>	639
Интерфейс <i>ServletRequest</i>	640
Интерфейс <i>ServletResponse</i>	641
Цикл работы сервлета.....	641
Класс <i>GenericServlet</i>	642
Работа по протоколу HTTP	643
Интерфейс <i>HttpServletRequest</i>	643
Интерфейс <i>HttpServletResponse</i>	645
Класс <i>HttpServlet</i>	646
Аннотации сервлета	646
Пример сервлета класса <i>HttpServlet</i>	647
Сеанс связи с сервлетом	652
Фильтры	655
Обращение к другим ресурсам	660
Асинхронное выполнение запросов	661
Вопросы для самопроверки	664
Глава 27. Страницы JSP.....	665
Стандартные действия (теги) JSP	668
Язык записи выражений EL	671
Встроенные объекты JSP.....	672
Обращение к компоненту <i>JavaBean</i>	674
Выполнение апплета в браузере клиента	675
Передача управления	676

Пользовательские теги	677
Класс-обработчик пользовательского тега	679
Пользовательский тег с атрибутами	681
Пользовательский тег с телом	682
Обработка тела пользовательского тега	684
Обработка взаимодействующих тегов	686
Обработка исключений в пользовательских тегах	690
Обработка тегов средствами JSP	690
Стандартные библиотеки тегов JSTL	692
Библиотека core	693
Библиотека xml	696
Библиотека fmt	696
Библиотека sql	697
Библиотека fn	697
Frameworks	697
JavaServer Faces	698
Вопросы для самопроверки	703
Глава 28. Связь Java с технологией XML	704
Описание DTD	709
Пространства имен XML	711
Схема XML	713
Встроенные простые типы XSD	714
Вещественные числа	714
Целые числа	714
Строки символов	714
Дата и время	715
Двоичные типы	715
Прочие встроенные простые типы	715
Определение простых типов	716
Сужение	716
Список	717
Объединение	718
Описание элементов и их атрибутов	719
Определение сложных типов	719
Определение типа пустого элемента	720
Определение типа элемента с простым телом	720
Определение типа вложенных элементов	721
Определение типа со сложным телом	723
Пример: схема адресной книги	724
Безымянные типы	726
Пространства имен языка XSD	728
Включение файлов схемы в другую схему	730
Связь документа XML со своей схемой	731
Другие языки описания схем	732
Инструкции по обработке	732
Анализ документа XML	733
Анализ документов XML с помощью SAX2	734

Анализ документов XML с помощью StAX	741
Связывание данных XML с объектами Java	743
Объекты данных JDO	744
Анализ документов XML с помощью DOM API.....	745
Интерфейс <i>Node</i>	746
Интерфейс <i>Document</i>	747
Интерфейс <i>Element</i>	748
Другие DOM-парсеры.....	751
Преобразование дерева объектов в XML.....	752
Таблицы стилей XSL	754
Преобразование документа XML в HTML	756
Вопросы для самопроверки	757
Список литературы	758
Предметный указатель	760

Введение

Книга, которую вы держите в руках, возникла из курса лекций, читаемых автором студентам младших курсов уже более десяти лет. Подобные книги рождаются после того, как студенты в очередной раз зададут вопрос, который лектор уже несколько раз разъяснял в разных вариациях. Возникает желание отослать их к... какой-нибудь литературе. Пересмотрев еще раз несколько десятков книг, использованных при подготовке лекций, порывшись в библиотеке и на прилавках книжных магазинов, лектор с удивлением обнаруживает, что не может предложить студентам ничего подходящего. Остается сесть за стол и написать книгу самому. Такое происхождение книги накладывает на нее определенные особенности. Она:

- ❑ представляет собой ступок практического опыта, накопленного автором и его студентами с 1996 г.;
- ❑ содержит ответы на часто задаваемые вопросы, последних "компьютерчики" называют *FAQ* (Frequently Asked Questions);
- ❑ написана кратко и сжато, как конспект лекций, в ней нет лишних слов (за исключением, может быть, тех, что вы только что прочитали);
- ❑ рассчитана на читателей, стремящихся быстро и всерьез ознакомиться с новинками компьютерных технологий;
- ❑ содержит много примеров применения конструкций Java, которые можно использовать как фрагменты больших производственных разработок в качестве "How to...?";
- ❑ включает материал, являющийся обязательной частью подготовки специалиста по информационным технологиям;
- ❑ не предполагает знание какого-либо языка программирования, а для знатоков — выделяет особенности языка Java среди других языков;
- ❑ предлагает обсуждение вопросов русификации Java.

Прочитав эту книгу, вы вступите в ряды программистов на Java — разработчиков передовой технологии начала XXI века.

Если спустя несколько месяцев эта книга будет валяться на вашем столе с растрепанными страницами, залитыми кофе и засыпанными пеплом, с массой закладок и загнутых углов, а вы начнете сетовать на то, что книга недостаточно полна и слишком проста и ее содержание тривиально и широко известно, а примеры банальны, тогда автор будет считать, что его скромный труд не пропал даром.

Пошел второй десяток лет с того дня, когда были написаны эти строки. Все случилось так, как я и написал. Разошлись три издания книги "Самоучитель Java". Я видел много ее экземпляров в самом разном состоянии. Читатели высказали мне множество нелицеприятных соображений по поводу содержания книги, обнаруженных ошибок и опечаток. Студенты на зачетах и экзаменах пересказывали мне целые куски книги, что тоже навело на размышления по поводу ее содержания и стиля изложения. У меня накопилось много дополнительного материала, который так и просился в книгу.

Технология Java развивается очень быстро. Сначала предназначавшаяся для небольших сетевых приложений, Java прочно утвердилась на Web-серверах, проникла в сотовые телефоны, планшеты и другие мобильные устройства. Популярная операционная система Android базируется на Java. Теперь Java — обязательная часть Web-программирования.

Развивается и сам язык. В него вводятся новые конструкции, появляются новые библиотеки классов. Графическая библиотека Swing стала частью стандартной поставки Java. В стандартную поставку теперь включены и средства работы с документами XML. Вышла уже седьмая версия Java.

Все это привело к необходимости сделать новое издание, дополнив книгу новым материалом и исправив, увы, неизбежные опечатки.

Ну что же, начнем!

Что такое Java?

Это остров Ява в Малайском архипелаге, территория Индонезии. Это сорт кофе, который любят пить создатели Java (произносится "джава", с ударением на первом слоге). А если серьезно, то ответить на этот вопрос трудно, потому что границы Java, и без того размытые, все время расширяются.

Сначала Java (официальный день рождения технологии Java — 23 мая 1995 г.) предназначалась для программирования бытовых электронных устройств, таких как сотовые телефоны и другие мобильные устройства.

Потом Java стала применяться для программирования браузеров — появились *апплеты*.

Затем оказалось, что на Java можно создавать полноценные приложения. Их графические элементы стали оформлять в виде компонентов — появились *JavaBeans*, с которыми Java вошла в мир распределенных систем и промежуточного программного обеспечения, тесно связавшись с технологией CORBA.

Остался один шаг до программирования серверов — этот шаг был сделан — появились *сервлеты* (servlets), страницы *JSP* (JavaServer Pages) и *EJB* (Enterprise JavaBeans). Серверы должны взаимодействовать с базами данных — появились драйверы *JDBC*. Взаимодействие оказалось удачным, и многие системы управления базами данных и даже операционные системы включили Java в свое ядро, например Oracle, Linux, MacOS X, AIX. Что еще не охвачено? Назовите и через полгода услышите, что Java уже вовсю применяется и там. Из-за этой размытости самого понятия его описывают таким же размытым словом — *технология*.

Такое быстрое и широкое распространение технологии Java не в последнюю очередь связано с тем, что она использует новый, специально созданный язык программирования, который так и называется — язык Java. Этот язык создан на базе языков Smalltalk,

Pascal, C++ и др., вобрав их лучшие, по мнению создателей, черты и отбросив худшие. На этот счет есть разные мнения, но бесспорно, что язык получился удобным для изучения, написанные на нем программы легко читаются и отлаживаются: первую программу можно написать уже через час после начала изучения языка. Язык Java становится языком обучения объектно-ориентированному программированию, так же как язык Pascal был языком обучения структурному программированию. Недаром на Java уже написано огромное количество программ, библиотек классов, а собственный апплет не написал только уж совсем ленивый.

Для полноты картины следует сказать, что создавать приложения для технологии Java можно не только на языке Java, есть и другие языки: Clojure, Scala, Jython, есть даже компиляторы с языков Pascal и C++, но лучше все-таки использовать язык Java: на нем все аспекты технологии излагаются проще и удобнее.

Язык Java часто используется для описания различных приемов объектно-ориентированного программирования, так же как для записи алгоритмов применялся вначале язык Algol, а затем язык Pascal.

Ясно, что всю технологию Java нельзя изложить в одной книге, полное описание ее возможностей составит целую библиотеку. Эта книга посвящена только языку Java. Прочитав ее, вы сможете создавать Java-приложения любой сложности, свободно разбираться в литературе и листингах программ, продолжать изучение аспектов технологии Java по специальной литературе и по исходным кодам свободно распространяемых программных продуктов.

Язык Java тоже очень бурно развивается, некоторые его методы объявляются устаревшими (deprecated), появляются новые конструкции, увеличивается встроенная библиотека классов, но есть устоявшееся ядро языка, сохраняется его дух и стиль. Вот это-то устоявшееся и излагается в книге.

Структура книги

Книга состоит из пяти частей.

Часть I содержит три главы, в которых рассматриваются базовые понятия языка. По прочтении ее вы сможете свободно разбираться в понятиях объектно-ориентированного программирования и их реализации на языке Java, создавать свои объектно-ориентированные программы, рассчитанные на консольный ввод/вывод.

В *главе 1* описываются типы исходных данных, операции с ними, выражения, массивы, операторы управления потоком информации, приводятся примеры записи часто встречающихся алгоритмов на Java. После знакомства с этой главой вы сможете писать программы на Java, реализующие любые вычислительные алгоритмы, встречающиеся в вашей практике.

В *главе 2* вводятся основные понятия объектно-ориентированного программирования: объект и метод, абстракция, инкапсуляция, наследование, полиморфизм, контракты методов и их поручения друг другу. Эта глава призвана привить вам "объектный" взгляд на реализацию сложных проектов, после ее прочтения вы научитесь описывать проект как совокупность взаимодействующих объектов. Здесь же предлагается реализация всех этих

понятий на языке Java. Тут вы, наконец, поймете, что же такое эти объекты и как они взаимодействуют.

В *главе 3* определяются пакеты классов и интерфейсы, ограничения доступа к классам и методам, на примерах подробно разбираются правила их использования. Объясняется структура встроенной библиотеки классов Java API.

В *части II* рассматриваются пакеты основных классов, составляющих неотъемлемую часть Java, разбираются приемы работы с ними и приводятся примеры практического использования основных классов. Здесь вы увидите, как идеи объектно-ориентированного программирования реализуются на практике в сложных производственных библиотеках классов. После изучения этой части вы сможете реализовывать наиболее часто встречающиеся ситуации объектно-ориентированного программирования с помощью стандартных классов.

Глава 4 прослеживает иерархию стандартных классов и интерфейсов Java, на этом примере показано, как в профессиональных системах программирования реализуются концепции абстракции, инкапсуляции и наследования.

В *главе 5* подробно излагаются приемы работы со строками символов, которые, как и всё в Java, являются объектами, приводятся примеры синтаксического анализа текстов, обсуждаются вопросы русификации.

В *главе 6* показано, как в языке Java реализованы коллекции, позволяющие работать с совокупностями объектов и создавать сложные структуры данных.

Глава 7 описывает различные классы-утилиты, полезные во многих ситуациях при работе с датами, случайными числами, словарями и другими необходимыми элементами программ.

В *части III* объясняется создание графического интерфейса пользователя (ГИП) с помощью стандартной библиотеки классов AWT (Abstract Window Toolkit) с компонентами Swing и даны многочисленные примеры построения интерфейса. Подробно разбирается принятый в Java метод обработки событий, основанный на идее делегирования. Здесь же появляются апплеты как программы Java, работающие в окне браузера. Подробно обсуждается система безопасности выполнения апплетов. После прочтения третьей части вы сможете создавать с помощью Swing полноценные приложения под графические платформы MS Windows, X Window System и др., а также программировать браузеры.

Глава 8 описывает иерархию классов библиотеки AWT, которую необходимо четко себе представлять для создания удобного интерфейса. Здесь же рассматривается библиотека графических компонентов Swing, ставшая стандартной наряду с AWT.

В *главе 9* демонстрируются приемы рисования с помощью графических примитивов, способы задания цвета и использование шрифтов, а также решается вопрос русификации приложений Java.

В *главе 10* обсуждается понятие графического компонента, рассматриваются готовые компоненты AWT и их применение, а также создание собственных компонентов AWT.

В *главе 11* рассматриваются графические компоненты общего назначения, относящиеся к библиотеке Swing.

В *главе 12* рассматриваются текстовые графические компоненты библиотеки Swing.

В *главе 13* подробно обсуждаются возможности создания таблиц средствами Swing.

В *главе 14* показано, какие способы размещения компонентов в графическом контейнере имеются в AWT и Swing и как их применять в разных ситуациях.

В *главе 15* вводятся способы реагирования компонентов на сигналы от клавиатуры и мыши, а именно модель делегирования, принятая в Java.

В *главе 16* описывается создание рамок, окружающих графические компоненты Swing.

В *главе 17* обсуждается интересная способность Swing изменять свой внешний вид, сливаясь с окружающей графической средой или, наоборот, выделяясь из нее.

В *главе 18*, наконец-то, появляются апплеты — Java-программы, предназначенные для выполнения в окне браузера, и обсуждаются особенности их создания.

В *главе 19* собраны сведения о библиотеке Swing, не вошедшие в предыдущие главы.

В *главе 20* рассматривается работа с изображениями и звуком средствами AWT.

В *части IV* изучаются конструкции языка Java, не связанные общей темой. Некоторые из них необходимы для создания надежных программ, учитывающих все нештатные ситуации, другие позволяют реализовывать сложное взаимодействие объектов. Здесь же рассматривается передача потоков данных от одной программы Java к другой. Внимательное изучение четвертой части позволит вам дополнить свои разработки гибкими средствами управления выполнением приложения, создавать сложные клиент-серверные системы.

Глава 21 описывает встроенные в Java средства обработки исключительных ситуаций, возникающих во время выполнения готовой программы.

Глава 22 рассказывает об интересном свойстве языка Java — способности создавать подпроцессы (threads) и управлять их взаимодействием прямо из программы.

В *главе 23* обсуждается концепция потока данных и ее реализация в Java для организации ввода/вывода на внешние устройства.

Глава 24 рассматривает сетевые средства языка Java, позволяющие скрыть все сложности протоколов Интернета и максимально облегчить написание клиент-серверных и распределенных приложений.

Часть V книги посвящена Web-технологии Java, точнее, тем ее разделам, которые касаются программирования серверов.

В *главе 25* описываются те аспекты технологии Java, которые необходимы для Web-программирования: архиватор JAR, компоненты JavaBeans, драйверы соединения с базами данных JDBC.

Глава 26 посвящена основному средству программирования серверов — сервлетам.

В *главе 27* разбираются страницы JSP, значительно облегчающие оформление ответов на запросы Web-клиентов.

Наконец, в *главе 28* рассматривается вездесущая технология XML и инструменты Java для обработки документов XML.

Выполнение Java-программы

Как вы знаете, программа, написанная на одном из языков высокого уровня, к которым относится и язык Java, так называемый *исходный модуль* ("исходник", или "сырец" на жаргоне от английского *source*), не может быть сразу же выполнена. Ее сначала надо скомпилировать, т. е. перевести в последовательность машинных команд — *объектный модуль*. Но и он, как правило, не может быть сразу же выполнен: объектный модуль надо еще скомпоновать с библиотеками использованных в модуле функций и разрешить перекрестные ссылки между секциями объектного модуля, получив в результате *загрузочный модуль* — полностью готовую к выполнению программу.

Исходный модуль, написанный на Java, не может избежать этих процедур, но здесь проявляется главная особенность технологии Java — программа компилируется сразу в машинные команды, но не команды какого-то конкретного процессора, а в команды так называемой виртуальной машины Java (Java Virtual Machine, JVM). *Виртуальная машина Java* — это совокупность команд вместе с системой их выполнения. Для специалистов скажем, что виртуальная машина Java полностью стековая, так что не требуется сложная адресация ячеек памяти и большое количество регистров. Поэтому команды JVM короткие, большинство из них имеет длину 1 байт, отчего команды JVM называют *байт-кодами* (bytecodes), хотя имеются команды длиной 2 и 3 байта. Согласно статистическим исследованиям средняя длина команды составляет 1,8 байта. Полное описание команд и всей архитектуры JVM содержится в *спецификации виртуальной машины Java* (Virtual Machine Specification, VMS). Ознакомьтесь с этой спецификацией, если вы хотите в точности узнать, как работает виртуальная машина Java.

Другая особенность Java — все стандартные функции, вызываемые в программе, подключаются к ней только на этапе выполнения, а не включаются в байт-коды. Как говорят специалисты, происходит *динамическая компоновка* (dynamic binding). Это тоже сильно уменьшает объем скомпилированной программы.

Итак, на первом этапе программа, написанная на языке Java, переводится компилятором в байт-коды. Эта компиляция не зависит от типа какого-либо конкретного процессора и архитектуры конкретного компьютера. Она может быть выполнена один раз сразу же после написания программы, программу не надо перекомпилировать под разные платформы. Байт-коды записываются в одном или нескольких файлах, могут храниться во внешней памяти или передаваться по сети. Это особенно удобно благодаря небольшому размеру файлов с байт-кодами. Затем полученные в результате компиляции байт-коды можно выполнять на любом компьютере, имеющем систему, реализующую JVM. При этом не важен ни тип процессора, ни архитектура компьютера. Так реализуется принцип Java "Write once, run anywhere" — "Написано однажды, выполняется где угодно".

Интерпретация байт-кодов и динамическая компоновка значительно замедляют выполнение программ. Это не имеет значения в тех ситуациях, когда байт-коды передаются по сети, сеть все равно медленнее любой интерпретации, но в других ситуациях требуется мощный и быстрый компьютер. Поэтому постоянно идет усовершенствование интерпретаторов в сторону увеличения скорости интерпретации. Разработаны *JIT-компиляторы* (Just-In-Time), запоминающие уже интерпретированные участки кода в машинных командах процессора и просто выполняющие эти участки при повторном обращении, например в циклах. Это значительно увеличивает скорость повторяющихся вычислений. Корпорация Sun Microsystems разработала целую технологию HotSpot и включает ее

в свою виртуальную машину Java. Но, конечно, наибольшую скорость может дать только специализированный процессор.

Компания Sun Microsystems выпустила микропроцессоры picoJava, работающие на системе команд JVM. Есть Java-процессоры и других фирм. Эти процессоры непосредственно выполняют байт-коды. Но при выполнении программ Java на других процессорах требуется еще интерпретация команд JVM в команды конкретного процессора, а значит, нужна программа-интерпретатор, причем для каждого типа процессоров и для каждой архитектуры компьютера следует написать свой интерпретатор.

Эта задача уже решена практически для всех компьютерных платформ. На них реализованы виртуальные машины Java, а для наиболее распространенных платформ имеется несколько реализаций JVM разных фирм. Все больше операционных систем и систем управления базами данных включают реализацию JVM в свое ядро. Создана и специальная операционная система JavaOS, применяемая в электронных устройствах. В большинство браузеров встроена виртуальная машина Java для выполнения апплетов. Операционная система Android содержит виртуальную машину Java, называемую Dalvik, которая работает на ядре Linux.

Программы, приведенные в этой книге, выполнялись в операционных средах программирования MS Windows 2000/XP/Server 2003, Red Hat Linux, Fedora Core Linux, SUSE Linux без перекомпиляции. Это видно по рисункам, приведенным во многих главах книги. Они "сняты" с экранов графических оболочек разных операционных систем.

Внимательный читатель уже заметил, что кроме реализации JVM для выполнения байт-кодов на компьютере еще нужно иметь набор функций, вызываемых из байт-кодов и динамически компонующихся с байт-кодами. Этот набор оформляется в виде библиотеки классов Java, состоящей из одного или нескольких *пакетов*. Каждая функция может быть записана байт-кодами, но, поскольку она будет храниться на конкретном компьютере, ее можно записать прямо в системе команд этого компьютера, избегнув тем самым интерпретации байт-кодов. Такие функции, написанные чаще всего на языке C/C++ и скомпилированные под определенную платформу, называют "*родными*" *методами* (native methods). Применение "родных" методов ускоряет выполнение программы.

Корпорация Oracle, купившая фирму Sun Microsystems — создателя технологии Java, — бесплатно распространяет набор необходимых программных инструментов для полного цикла работы с этим языком программирования: компиляции, интерпретации, отладки, включающий и богатую библиотеку классов. Называется этот набор JDK (Java Development Kit). Он весь содержится в одном файле. Есть наборы инструментальных программ и других фирм. Например, большой популярностью пользуется JDK корпорации IBM.

Что такое JDK?

Набор программ и классов JDK содержит:

- компилятор из исходного текста в байт-коды `javac`;
- интерпретатор `java`, содержащий реализацию JVM;
- облегченный интерпретатор `jre` (в последних версиях отсутствует);

- программу просмотра апплетов `appletviewer`, заменяющую браузер;
- отладчик `jdb`;
- дизассемблер `javap`;
- программу архивации и сжатия `jar`;
- программу сбора и генерирования документации `javadoc`;
- программу генерации заголовочных файлов языка C для создания "родных" методов `javah`;
- программу генерации электронных ключей `keytool`;
- программу `native2ascii`, преобразующую бинарные файлы в текстовые;
- программы `rmic` и `rmiregistry` для работы с удаленными объектами;
- программу `serialver`, определяющую номер версии класса;
- библиотеки и заголовочные файлы "родных" методов;
- библиотеку классов Java API (Application Programming Interface).

В прежние версии JDK включались и отладочные варианты исполнимых программ: `javac_g`, `java_g` и т. д.

Компания Sun Microsystems активно развивала и обновляла JDK, почти каждый год выходили новые версии.

В 1996 г. была выпущена первая версия — JDK 1.0, которая модифицировалась до версии с номером 1.0.2. В этой версии библиотека классов Java API содержала 8 пакетов. Весь набор JDK 1.0.2 поставлялся в упакованном виде в одном файле размером около 5 Мбайт, а после распаковки занимал на диске около 8 Мбайт.

В 1997 г. появилась версия JDK 1.1, последняя ее модификация, 1.1.8, выпущена в 1998 г. В этой версии было 23 пакета классов, занимала она 8,5 Мбайт в упакованном виде и около 30 Мбайт — в распакованном.

В первых версиях JDK все пакеты библиотеки Java API были упакованы в один архивный файл `classes.zip` и вызывались непосредственно из этого архива, его не нужно было распаковывать.

Затем набор инструментальных средств JDK был сильно переработан.

Версия JDK 1.2 вышла в декабре 1998 г. и содержала уже 57 пакетов классов. В архивном виде это файл размером почти 20 Мбайт и еще отдельный файл размером более 17 Мбайт с упакованной документацией. Полная версия располагается на 130 Мбайт дискового пространства, из них около 80 Мбайт занимает документация.

Начиная с этой версии, все продукты технологии Java собственного производства компания Sun стала называть *Java 2 Platform, Standard Edition*, сокращенно J2SE, а в литературе утвердилось название Java 2. Кроме 57 пакетов классов, обязательных на любой платформе и получивших название *Core API*, в Java 2 JDK 1.2 входят еще дополнительные пакеты классов, называемые Standard Extension API.

В версии J2SE JDK 1.5.0, вышедшей в конце 2004 г., было уже под сотню пакетов, составляющих Core API (Application Programming Interface). В упакованном виде — это файл размером около 46 Мбайт и необязательный файл с упакованной документацией такого же размера. В это же время произошло очередное переименование технологии

Java: из версии убрали первую цифру и стали писать *Java 2 Platform, Standard Edition 5.0*, сокращенно J2SE 5.0 и JDK 5.0, хотя во внутрифирменной документации сохраняется название JDK 1.5.0.

Последнее обновление J2SE 5.0, JDK 1.5.0_22, было выпущено 3 ноября 2009 года.

В шестой версии, вышедшей в начале 2007 г., из названия технологии убрали цифру 2 и стали писать *Java Platform, Standard Edition 6*, сокращенно — Java SE 6 и JDK 6. Впрочем, во внутрифирменной документации остается прежнее обозначение, например последнее на момент написания книги обновление обозначается JDK 1.6.0_26.

Летом 2011 года появилась седьмая версия Java SE 7 и распространяется JDK 1.7.0, описанию которой посвящена эта книга.

Java SE JDK создается для каждой платформы: MS Windows, Solaris, Linux, отдельно, а документация написана на языке HTML и одинакова на всех платформах. Поэтому она записана в отдельном файле. Например, для MS Windows файл с Java SE JDK 1.7.0 называется `jdk-7-windows-i586.exe` с добавлением номера обновления, а файл с документацией называется `jdk-7-fcs-bin-b147-apidocs-27_jun_2011.zip`.

Эти файлы можно совершенно свободно скачать со страницы <http://www.oracle.com/technetwork/java/javase/downloads/index.html>.

Для создания Web-программ в *части V* книги вам потребуется еще набор пакетов *Java Platform, Enterprise Edition* (Java EE). Так же как Java SE, он поставляется одним самораспаковывающимся архивом, в который входит SDK (Software Development Kit), Java EE API и сервер приложений. Архив можно скопировать с того же сайта. Набор Java EE SDK — это дополнение к Java SE и поэтому устанавливается после Java SE JDK. Впрочем, на том же сайте есть полная версия архива, содержащая в себе и Java EE SDK, и Java SE JDK.

Java EE входит в состав серверов приложений, поэтому если вы установили JBoss, GlassFish или другой сервер приложений, то у вас уже есть набор классов Java EE.

Кроме JDK компания Oracle отдельно распространяет еще и набор JRE (Java Runtime Environment).

Что такое JRE?

Набор программ и пакетов классов JRE содержит все необходимое для выполнения байт-кодов, в том числе интерпретатор `java` (в прежних версиях — облегченный интерпретатор `jre`) и библиотеку классов. Это часть JDK, не содержащая компиляторы, отладчики и другие средства разработки. Именно Oracle JRE или его аналог, созданный другими фирмами, присутствует в тех браузерах, которые умеют выполнять программы на Java, в операционных системах и системах управления базами данных.

Хотя JRE входит в состав JDK, корпорация Oracle распространяет этот набор и отдельным файлом.

Как установить JDK?

Напомню, что набор JDK упаковывается в самораспаковывающийся архив. Раздобыв каким-либо образом этот архив: скачав из Интернета, с сайта <http://www.oracle.com/>

technetwork/java/javase/downloads/index.html или какого-то другого адреса, вам остается только запустить файл с архивом на выполнение. Откроется окно установки, в котором среди всего прочего вам будет предложено выбрать каталог (directory) установки, например, /usr/java/jdk1.7.0. Каталог и его название можно поменять, место и название установки не имеют значения.

После установки вы получите каталог с названием, например, jdk1.7.0, а в нем подкаталоги:

- bin с исполнимыми файлами;
- db с небольшой базой данных;
- demo с примерами программ, присутствует не во всех версиях JDK;
- docs с документацией, если вы ее установили в этот каталог;
- include с заголовочными файлами "родных" методов;
- jre с набором JRE;
- lib с библиотеками классов и файлами свойств;
- sample с примерами программ, присутствует не во всех версиях JDK;
- src с исходными текстами программ JDK, получаемый после распаковки файла src.zip.

Да-да! Набор JDK содержит исходные тексты большинства своих программ, написанные на Java. Это очень удобно. Вы всегда можете в точности узнать, как работает тот или иной метод обработки информации из JDK, посмотрев исходный код данного метода. Это очень полезно и для изучения Java на "живых", работающих примерах.

ПРЕДУПРЕЖДЕНИЕ

Не следует распаковывать zip- и jar-архивы, кроме архива исходных текстов src.zip.

После установки надо дополнить значение системной переменной PATH, добавив в нее путь к каталогу bin, например /usr/java/jdk1.7.0/bin. Некоторые программы, использующие Java, требуют определить и специальную переменную окружения JAVA_HOME, содержащую путь к каталогу установки JDK, например /usr/java/jdk1.7.0.

Проверить правильность установки Java, а заодно и посмотреть ее версию можно, набрав в командной строке

```
java -version
```

Как использовать JDK?

Несмотря на то что набор JDK предназначен для создания программ, работающих в графических средах, таких как MS Windows или X Window System, он ориентирован на выполнение из командной строки окна **Command Prompt** в MS Windows. В системах UNIX, Linux, BSD можно работать и в текстовом режиме, и в окне **Xterm**.

Написать программу на Java можно в любом текстовом редакторе, например Notepad, WordPad в MS Windows, редакторах vi, emacs в UNIX. Надо только сохранить файл в текстовом, а не графическом формате и дать ему расширение java. Пусть, для примера, именем файла будет MyProgram.java, а сам файл сохранен в текущем каталоге.

После создания этого файла из командной строки вызывается компилятор `javac` и ему передается исходный файл как параметр:

```
javac MyProgram.java
```

Компилятор создает в том же каталоге по одному файлу на каждый класс, описанный в программе, называя каждый файл именем класса с расширением `class`. Допустим, в нашем примере имеется только один класс, названный `MyProgram`, тогда получаем файл с именем `MyProgram.class`, содержащий байт-коды.

Компилятор молчалив — если компиляция прошла успешно, он ничего не сообщит, на экране появится только приглашение операционной системы. Если же компилятор заметит ошибки, то он выведет на экран сообщения о них. Большое достоинство компилятора JDK в том, что он "отлавливает" много ошибок и выдает подробные и понятные сообщения.

Далее из командной строки вызывается интерпретатор байт-кодов `java`, которому передается файл с байт-кодами, причем его имя записывается без расширения (смысл этого вы узнаете позднее):

```
java MyProgram
```

На экране появится вывод результатов работы программы или сообщения об ошибках времени выполнения.

Работая в графических оболочках операционных систем, мы привыкли вызывать программу на исполнение двойным щелчком мыши по имени исполнимого файла (в MS Windows у имени исполнимого файла стандартное расширение `exe`) или щелчком по его ярлыку. В технологии Java тоже есть такая возможность. Надо только упаковать `class`-файлы с байт-кодами в архив специального вида JAR. Как это сделать, рассказано в *главе 25*. При установке JDK на MS Windows для файлов с расширением `jar` автоматически создается ассоциация с интерпретатором `java`, который будет вызван при двойном щелчке мыши на `jar`-архиве.

Кроме того, можно написать командный файл (файл с расширением `bat` в MS Windows или Shell-файл командной оболочки в UNIX), записав в нем строку вызова интерпретатора `java` со всеми нужными параметрами.

Еще один способ запустить Java-программу средствами операционной системы — написать загрузчик (launcher) виртуальной машины Java. Так и сделано в стандартной поставке JDK: исполнимый файл `java.exe` содержит программу, написанную на языке C, которая запускает виртуальную машину Java и передает ей на исполнение класс Java с методом `main()`. Исходный текст этой программы есть среди исходных текстов Java в каталоге `src/launcher`. Им можно воспользоваться для написания своего загрузчика. Есть много программ, облегчающих написание загрузчика, например программа Java Launcher фирмы SyncEdit, <http://www.syncedit.com/software/javalauncher/>, или Advanced Installer for Java фирмы CapHyon, <http://www.advancedinstaller.com/>.

Наконец, существуют компиляторы исходного текста, написанного на языке Java, непосредственно в исполнимый файл операционной системы, с которой вы работаете. Их общее название AOT (Ahead-Of-Time) compiler. Например, у знаменитого компилятора GCC (GNU Compiler Collection) есть вход с именем `Gcj`, с помощью которого можно сделать компиляцию как в байт-коды, так и в исполнимый файл, а также перекомпиляцию байт-кодов в исполнимый файл.