



Самоучитель

Ильдар Хабибуллин

Java

3-е издание



Принципы объектно-ориентированного программирования

Пакеты классов и интерфейсы Java SE 6

Библиотеки Swing и Java 2D

Апплеты, графика, звук и печать

Технологии Servlet, JSP, JSTL, JSF, XML

Около двухсот законченных программ

Ильдар Хабибуллин

Самоучитель
Java

3-е издание

Санкт-Петербург

«БХВ-Петербург»

2008

УДК 681.3.06
ББК 32.973.26-018.2
X12

Хабибуллин И. Ш.

X12 Самоучитель Java. — 3-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2008. — 768 с.: ил.

ISBN 978-5-9775-0191-0

Самоучитель написан на основе материала лекций, читаемых автором для студентов первого и второго курсов. 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. Третье издание содержит полное описание нововведений Java SE 6. Дано подробное изложение сервлетов, технологий JSP и библиотек тегов JSTL. Около двухсот законченных программ иллюстрируют приведенные приемы программирования. Подробные схемы и описания классов и методов Core Java API позволяют применять книгу и как настольный справочник по технологии Java.

Для широкого круга 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>Владимир Красовский</i>
Компьютерная верстка	<i>Натальи Смирновой</i>
Корректор	<i>Виктория Пиотровская</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 28.03.08.
Формат 70×100^{1/16}. Печать офсетная. Усл. печ. л. 61,92.
Тираж 3000 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию № 77.99.60.953.Д.002108.02.07 от 28.02.2007 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

Оглавление

ВВЕДЕНИЕ	1
Что такое Java?	2
Структура книги	3
Выполнение Java-программы	5
Что такое JDK?	7
Что такое JRE?	9
Как установить JDK?	9
Как использовать JDK?	10
Интегрированные среды Java	11
Особая позиция Microsoft	12
Java в Интернете	12
Литература по Java	13
Благодарности	14
ЧАСТЬ I. БАЗОВЫЕ КОНСТРУКЦИИ ЯЗЫКА JAVA.....	15
ГЛАВА 1. ВСТРОЕННЫЕ ТИПЫ ДАННЫХ, ОПЕРАЦИИ НАД НИМИ.....	17
Первая программа на Java	17
Комментарии	20
Аннотации	22
Константы	22
Целые	22
Действительные	23
Символы	23
Строки	24
Имена	25
Примитивные типы данных и операции	26
Логический тип	27
Логические операции	27
Упражнения	28
Целые типы	28
Операции над целыми типами	29
Арифметические операции	29
Приведение типов	30

Операции сравнения.....	32
Побитовые операции.....	32
Сдвиги.....	33
Упражнения	34
Вещественные типы.....	34
Операции присваивания	35
Упражнения	36
Условная операция.....	36
Упражнения	36
Выражения	36
Приоритет операций	37
Операторы.....	38
Блок.....	39
Операторы присваивания	39
Условный оператор	39
Упражнения	41
Операторы цикла.....	42
Оператор <i>continue</i> и метки.....	44
Оператор <i>break</i>	45
Упражнения	45
Оператор варианта	46
Массивы	47
Многомерные массивы	49
Заключение	51
Вопросы для самопроверки.....	51

ГЛАВА 2. ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ В JAVA 52

Парадигмы программирования.....	52
Принципы объектно-ориентированного программирования	55
Абстракция.....	55
Иерархия	58
Ответственность	59
Модульность	60
Принцип KISS.....	62
Упражнения	63
Как описать класс и подкласс?	63
Передача аргументов в метод	64
Перегрузка методов.....	66
Переопределение методов.....	67
Реализация полиморфизма в Java	67
Упражнения	69
Абстрактные методы и классы	69
Окончательные члены и классы	70
Класс <i>Object</i>	70
Конструкторы класса	71

Операция <i>new</i>	72
Упражнение	73
Статические члены класса.....	73
Класс <i>Complex</i>	75
Метод <i>main()</i>	78
Методы с переменным числом аргументов.....	79
Где видны переменные	80
Вложенные классы.....	81
Отношения "быть частью" и "являться"	86
Заключение	86
Вопросы для самопроверки.....	87

ГЛАВА 3. ПАКЕТЫ, ИНТЕРФЕЙСЫ И ПЕРЕЧИСЛЕНИЯ 88

Пакет и подпакет	89
Права доступа к членам класса.....	90
Размещение пакетов по файлам.....	92
Импорт классов и пакетов	94
Java-файлы	95
Интерфейсы	96
Перечисления	100
Объявление аннотаций	103
Design patterns	104
Схема проектирования MVC	105
Шаблон Singleton.....	106
Заключение	107
Вопросы для самопроверки.....	107

ЧАСТЬ II. ИСПОЛЬЗОВАНИЕ КЛАССОВ ИЗ JAVA DEVELOPMENT KIT 109

ГЛАВА 4. КЛАССЫ-ОБОЛОЧКИ И GENERICS 111

Числовые классы.....	112
Автоматическая упаковка и распаковка типов	114
Настраиваемые типы (generics).....	115
Шаблон типа (wildcard type).....	117
Настраиваемые методы	119
Класс <i>Boolean</i>	119
Класс <i>Character</i>	120
Класс <i>BigInteger</i>	123
Класс <i>BigDecimal</i>	125
Класс <i>Class</i>	129
Вопросы для самопроверки.....	131

ГЛАВА 5. РАБОТА СО СТРОКАМИ	132
Класс <i>String</i>	133
Как создать строку	133
Упражнение	138
Сцепление строк	138
Манипуляции строками	138
Как узнать длину строки	139
Как выбрать символы из строки	139
Как выбрать подстроку	140
Как разбить строку на подстроки	140
Как сравнить строки	141
Как найти символ в строке	142
Как найти подстроку	143
Как изменить регистр букв	143
Как заменить отдельный символ	144
Как заменить подстроку	144
Как убрать пробелы в начале и конце строки	144
Как преобразовать в строку данные другого типа	144
Упражнения	145
Класс <i>StringBuilder</i>	145
Конструкторы	145
Как добавить подстроку	146
Как вставить подстроку	146
Как удалить подстроку	147
Как удалить символ	147
Как заменить подстроку	147
Как перевернуть строку	148
Синтаксический разбор строки	148
Класс <i>StringTokenizer</i>	148
Заключение	149
Вопросы для самопроверки	149
ГЛАВА 6. КЛАССЫ-КОЛЛЕКЦИИ	150
Класс <i>Vector</i>	150
Как создать вектор	151
Как добавить элемент в вектор	151
Как заменить элемент	152
Как узнать размер вектора	152
Как обратиться к элементу вектора	152
Как узнать, есть ли элемент в векторе	152
Как узнать индекс элемента	152
Как удалить элементы	153
Класс <i>Stack</i>	154

Класс <i>Hashtable</i>	155
Как создать таблицу <i>Hashtable</i>	155
Как заполнить таблицу <i>Hashtable</i>	156
Как получить значение по ключу	156
Как узнать наличие ключа или значения	156
Как получить все элементы таблицы <i>Hashtable</i>	156
Как удалить элементы	157
Класс <i>Properties</i>	158
Интерфейс <i>Collection</i>	160
Интерфейс <i>List</i>	161
Интерфейс <i>Set</i>	162
Интерфейс <i>SortedSet</i>	162
Интерфейс <i>NavigableSet</i>	162
Интерфейс <i>Queue</i>	163
Интерфейс <i>BlockingQueue</i>	164
Интерфейс <i>Deque</i>	164
Интерфейс <i>BlockingDeque</i>	165
Интерфейс <i>Map</i>	165
Вложенный интерфейс <i>Map.Entry</i>	166
Интерфейс <i>SortedMap</i>	166
Интерфейс <i>NavigableMap</i>	167
Абстрактные классы-коллекции	168
Интерфейс <i>Iterator</i>	168
Интерфейс <i>ListIterator</i>	170
Классы, создающие списки	171
Двунаправленный список	171
Дек	172
Упражнение	172
Классы, создающие отображения	172
Связанные отображения	172
Упорядоченные отображения	173
Сравнение элементов коллекций	173
Упражнение	174
Классы, создающие множества	174
Связанные множества	175
Упорядоченные множества	175
Действия с коллекциями	176
Методы класса <i>Collections</i>	176
Упражнение	177
Заключение	177
Вопросы для самопроверки	177

ГЛАВА 7. КЛАССЫ-УТИЛИТЫ 178

Работа с массивами	178
Сортировка массива	178

Бинарный поиск в массиве	178
Заполнение массива	179
Копирование массива	179
Сравнение массивов.....	180
Представление массива строкой.....	180
Получение хеш-кода массива.....	181
Локальные установки.....	181
Работа с датами и временем.....	182
Часовой пояс и летнее время.....	183
Класс <i>Calendar</i>	183
Подкласс <i>GregorianCalendar</i>	184
Представление даты и времени	185
Получение случайных чисел.....	186
Копирование массивов	186
Взаимодействие с системой	187

ЧАСТЬ III. СОЗДАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ И АППЛЕТОВ 189

ГЛАВА 8. ПРИНЦИПЫ ПОСТРОЕНИЯ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА 191

Компонент и контейнер.....	193
Иерархия классов AWT	196
Окно библиотеки Swing.....	197
Использование системных приложений.....	198
System Tray.....	199
Splash Screen	199
Заключение	200
Вопросы для самопроверки.....	200

ГЛАВА 9. ГРАФИЧЕСКИЕ ПРИМИТИВЫ 201

Методы класса <i>Graphics</i>	201
Как задать цвет	201
Упражнение	203
Как нарисовать чертеж	203
Класс <i>Polygon</i>	204
Упражнение	205
Прочие методы класса <i>Graphics</i>	205
Как вывести текст.....	205
Как установить шрифт	206
Как задать шрифт	206
Класс <i>FontMetrics</i>	210
Упражнение	213
Возможности Java 2D	213

Преобразование координат	214
Класс <i>AffineTransform</i>	215
Упражнение	218
Рисование фигур средствами Java 2D	218
Класс <i>BasicStroke</i>	218
Класс <i>GeneralPath</i>	221
Классы <i>GradientPaint</i> и <i>TexturePaint</i>	222
Классы <i>LinearGradientPaint</i> и <i>RadialGradientPaint</i>	224
Вывод текста средствами Java 2D	225
Методы улучшения визуализации	230
Упражнение	231
Заключение	231
Вопросы для самопроверки.....	232
Глава 10. ОСНОВНЫЕ КОМПОНЕНТЫ AWT.....	233
Класс <i>Component</i>	233
Класс <i>Cursor</i>	235
Как создать свой курсор	235
Упражнение	236
События.....	237
Класс <i>Container</i>	237
События.....	238
Текстовая метка <i>Label</i>	238
События.....	238
Кнопка <i>Button</i>	238
События.....	239
Кнопка выбора <i>Checkbox</i>	239
События.....	239
Класс <i>CheckboxGroup</i>	239
Как создать группу радиокнопок.....	240
Раскрывающийся список <i>Choice</i>	241
События.....	242
Список <i>List</i>	242
События.....	243
Компоненты для ввода текста.....	244
Класс <i>TextComponent</i>	244
События	245
Строка ввода <i>TextField</i>	245
События	246
Поле ввода <i>TextArea</i>	246
События	246
Линейка прокрутки <i>Scrollbar</i>	248
События.....	249
Контейнер <i>Panel</i>	251
Контейнер <i>ScrollPane</i>	252

Контейнер <i>Window</i>	252
События.....	253
Контейнер <i>Frame</i>	253
События.....	254
Контейнер <i>Dialog</i>	255
События.....	257
Контейнер <i>FileDialog</i>	259
События.....	259
Создание собственных компонентов	260
Компонент <i>Canvas</i>	260
Создание "легкого" компонента	262
Упражнение	264
Создание меню	264
Всплывающее меню.....	270
Вопросы для самопроверки.....	273

ГЛАВА 11. ОФОРМЛЕНИЕ ГИП КОМПОНЕНТАМИ SWING 274

Состав библиотеки Swing.....	275
Основные компоненты Swing.....	277
Компонент <i>JComponent</i>	277
Схема MVC в компонентах Swing.....	278
Надпись <i>JLabel</i>	280
Кнопки.....	282
Кнопка <i>JButton</i>	284
Кнопка выбора <i>JToggleButton</i>	285
Кнопка выбора <i>JCheckBox</i>	286
Радиокнопка <i>JRadioButton</i>	286
Упражнение	288
Раскрывающийся список <i>JComboBox</i>	288
Список выбора <i>JList</i>	289
Визуализация элементов списков.....	291
Упражнение	293
Счетчик <i>JSpinner</i>	293
Полосы прокрутки <i>JScrollBar</i>	294
Ползунок <i>JSlider</i>	295
Упражнение	296
Индикатор <i>JProgressBar</i>	296
Дерево объектов <i>JTree</i>	297
Построение меню средствами Swing	301
Строка меню <i>JMenuBar</i>	301
Меню <i>JMenu</i>	301
Пункт меню <i>JMenuItem</i>	302
Пункт меню <i>JCheckBoxMenuItem</i>	303
Пункт меню <i>JRadioButtonMenuItem</i>	304
Всплывающее меню <i>JPopupMenu</i>	304

Панель выбора цвета <i>JColorChooser</i>	305
Упражнение	307
Окно выбора файла <i>JFileChooser</i>	307
Фильтр файлов <i>FileFilter</i>	307
Как получить выбранный файл	309
Дополнительный компонент	310
Замена изображений.....	311
Русификация Swing.....	312
Вопросы для самопроверки.....	313
ГЛАВА 12. ТЕКСТОВЫЕ КОМПОНЕНТЫ.....	314
Компонент <i>JTextComponent</i>	314
Модель данных — документ	314
Строка символов <i>Segment</i>	315
Запись текста в документ	316
Атрибуты текста	316
Удаление текста из документа	317
Фильтрация документа.....	317
Внесение структуры в документ	317
События в документе	318
Реализации документа.....	318
Установка модели данных	319
Вид.....	319
Контроллер — редактор текста.....	321
Курсор.....	321
Ограничение перемещения курсора	322
Реализации редактора	323
Раскладка клавиатуры.....	323
Печать текста документа	324
Поле ввода <i>JTextField</i>	325
Поле ввода пароля <i>JPasswordField</i>	327
Редактор объектов <i>JFormattedTextField</i>	327
Область ввода <i>JTextArea</i>	328
Текстовый редактор <i>JEditorPane</i>	329
Редактор <i>JTextPane</i>	330
Вопросы для самопроверки.....	331
ГЛАВА 13. ТАБЛИЦЫ	332
Класс <i>JTable</i>	332
Модель данных таблицы	334
Модель ячеек таблицы.....	334
Свойства столбца таблицы <i>TableColumn</i>	339
Модель столбцов таблицы.....	339
Заголовки столбцов таблицы <i>JTableHeader</i>	340

Модель выделения ячеек	341
Визуализация ячеек таблицы	342
Редактор ячеек таблицы	346
Сортировка строк таблицы.....	349
Фильтрация строк таблицы	350
Печать таблицы	352
Вопросы для самопроверки.....	352
ГЛАВА 14. РАЗМЕЩЕНИЕ КОМПОНЕНТОВ И КОНТЕЙНЕРЫ SWING.....	353
Менеджер <i>FlowLayout</i>	353
Менеджер <i>BorderLayout</i>	355
Менеджер <i>GridLayout</i>	357
Менеджер <i>CardLayout</i>	359
Менеджер <i>GridBagLayout</i>	360
Контейнеры Swing	362
Панель <i>JPanel</i>	362
Панель прокрутки <i>JScrollPane</i>	363
Двойная панель <i>JSplitPane</i>	365
Панель с вкладками <i>JTabbedPane</i>	367
Линейная панель <i>Box</i>	369
Менеджер размещения <i>BoxLayout</i>	369
Компоненты-заполнители	369
Менеджер размещения <i>SpringLayout</i>	371
Размеры <i>Spring</i>	372
Промежутки <i>Constraints</i>	373
Размещение компонентов.....	373
Панель инструментальных кнопок <i>JToolBar</i>	375
Интерфейс <i>Action</i>	377
Слоеная панель <i>JLayeredPane</i>	378
Корневая панель <i>JRootPane</i>	380
Окно <i>JWindow</i>	381
Диалоговое окно <i>JDialog</i>	382
Окно верхнего уровня <i>JFrame</i>	383
Внутреннее окно <i>JInternalFrame</i>	384
Рабочий стол <i>JDesktopPane</i>	386
Стандартные диалоги <i>JOptionPane</i>	387
Окно с индикатором <i>ProgressMonitor</i>	392
Заключение	393
Вопросы для самопроверки.....	393
ГЛАВА 15. ОБРАБОТКА СОБЫТИЙ.....	395
Самообработка событий.....	399
Обработка вложенным классом.....	400
Упражнение	401

Событие <i>ActionEvent</i>	402
Обработка действий мыши	402
Упражнение	405
Классы-адаптеры.....	406
Управление колесиком мыши.....	406
Обработка действий клавиатуры.....	408
Упражнение	409
Событие <i>TextEvent</i>	409
Событие изменения <i>ChangeEvent</i>	409
Обработка действий с окном	410
Событие <i>ComponentEvent</i>	411
Событие <i>ContainerEvent</i>	411
Событие <i>FocusEvent</i>	412
Событие <i>ItemEvent</i>	412
Событие <i>AdjustmentEvent</i>	412
Несколько слушателей одного источника	415
Диспетчеризация событий.....	416
Создание собственного события	418
Вопросы для самопроверки.....	419
ГЛАВА 16. ОФОРМЛЕНИЕ РАМОК.....	420
Пустая рамка <i>EmptyBorder</i>	422
Прямолинейная рамка <i>LineBorder</i>	422
Объемная рамка <i>BevelBorder</i>	423
Закругленная объемная рамка <i>SoftBevelBorder</i>	424
Врезанная рамка <i>EtchedBorder</i>	424
Рамка с изображением <i>MatteBorder</i>	424
Рамки с надписями <i>TitledBorder</i>	426
Сдвоенные рамки <i>CompoundBorder</i>	428
Создание собственных рамок	430
Вопросы для самопроверки.....	435
ГЛАВА 17. ИЗМЕНЕНИЕ ВНЕШНЕГО ВИДА КОМПОНЕНТА.....	436
Получение свойств L&F	438
Задание стандартного L&F.....	440
Дополнительные L&F	442
Смена всего L&F	442
Замена отдельных свойств L&F.....	445
Темы Java L&F.....	447
Вопросы для самопроверки.....	450
ГЛАВА 18. АППЛЕТЫ	451
Упражнения	457
Передача параметров в апплет.....	457

Параметры тега <code><applet></code>	460
Сведения об окружении апплета	461
Упражнение	461
Изображение и звук в апплетах	461
Слежение за процессом загрузки.....	462
Класс <i>MediaTracker</i>	463
Упражнения	465
Защита от апплета	465
Апплеты в библиотеке Swing.....	466
Апплет <i>JApplet</i>	467
Упражнение	468
Заключение	469
Вопросы для самопроверки.....	469
ГЛАВА 19. ПРОЧИЕ СВОЙСТВА SWING	470
Свойства экземпляра компонента	470
Прокрутка содержимого компонента.....	471
Передача фокуса ввода	471
Перенос данных Drag and Drop.....	476
Временная задержка <i>Timer</i>	478
ГЛАВА 20. ИЗОБРАЖЕНИЯ И ЗВУК.....	480
Модель "поставщик-потребитель"	480
Классы-фильтры.....	483
Как выделить фрагмент изображения	484
Как изменить цвет изображения.....	486
Как переставить пиксели изображения	486
Упражнения	488
Модель обработки прямым доступом	488
Преобразование изображения в Java 2D.....	491
Аффинное преобразование изображения	491
Изменение интенсивности изображения	494
Изменение составляющих цвета.....	495
Создание различных эффектов.....	496
Упражнения	497
Анимация	498
Улучшение изображения двойной буферизацией.....	500
Упражнения	504
Звук.....	504
Проигрывание звука в Java	505
Синтез и запись звука в Java 2	510
Упражнение	513
Вопросы для самопроверки.....	513

ЧАСТЬ IV. НЕОБХОДИМЫЕ КОНСТРУКЦИИ JAVA	515
ГЛАВА 21. ОБРАБОТКА ИСКЛЮЧИТЕЛЬНЫХ СИТУАЦИЙ	517
Блоки перехвата исключения.....	518
Упражнения	521
Часть заголовка метода <i>throws</i>	522
Оператор <i>throw</i>	524
Иерархия классов-исключений.....	525
Порядок обработки исключений	526
Упражнение	527
Создание собственных исключений.....	527
Заключение	528
Вопросы для самопроверки.....	528
ГЛАВА 22. ПОДПРОЦЕССЫ.....	530
Класс <i>Thread</i>	532
Синхронизация подпроцессов	536
Согласование работы нескольких подпроцессов.....	539
Приоритеты подпроцессов.....	543
Подпроцессы-демоны	543
Группы подпроцессов.....	544
Заключение	544
Вопросы для самопроверки.....	545
ГЛАВА 23. ПОТОКИ ВВОДА/ВЫВОДА И ПЕЧАТЬ.....	546
Консольный ввод/вывод.....	551
Форматированный вывод	554
Спецификации вывода целых чисел.....	555
Спецификации вывода вещественных чисел.....	556
Спецификация вывода символов	556
Спецификации вывода строк.....	556
Спецификации вывода логических значений	556
Спецификации вывода хеш-кода объекта.....	556
Спецификации вывода даты и времени	556
Класс <i>Console</i>	557
Упражнения	558
Файловый ввод/вывод	558
Получение свойств файла.....	560
Буферизованный ввод/вывод	562
Каналы буферизованного ввода/вывода	563
Упражнения	564
Поток простых типов Java.....	564
Кодировка UTF-8.....	565
Класс <i>DataOutputStream</i>	565

Прямой доступ к файлу	566
Упражнение	567
Каналы обмена информацией	567
Сериализация объектов	569
Печать в Java	572
Печать средствами Java 2D	574
Печать файла	578
Печать страниц с разными параметрами	580
Вопросы для самопроверки	582
ГЛАВА 24. СЕТЕВЫЕ СРЕДСТВА JAVA	583
Работа в WWW	586
Упражнения	589
Работа по протоколу TCP	590
Работа с проху-сервером	594
Упражнения	594
Работа по протоколу UDP	594
Упражнение	597
Вопросы для самопроверки	597
ЧАСТЬ V. WEB-ТЕХНОЛОГИИ JAVA	599
ГЛАВА 25. WEB-ИНСТРУМЕНТЫ JAVA	601
Архиватор <i>jar</i>	601
Создание архива	602
Файл описания MANIFEST.MF	604
Файл INDEX.LIST	605
Компоненты JavaBeans	606
Связь с базами данных через JDBC	607
Вопросы для самопроверки	612
ГЛАВА 26. СЕРВЛЕТЫ	613
Web-приложение	614
Интерфейс <i>Servlet</i>	615
Конфигурационный файл	615
Интерфейс <i>ServletConfig</i>	618
Контекст сервлета	620
Метод <i>Service</i>	621
Интерфейс <i>ServletRequest</i>	621
Интерфейс <i>ServletResponse</i>	623
Цикл работы сервлета	623
Класс <i>GenericServlet</i>	624

Работа по протоколу HTTP	625
Интерфейс <i>HttpServletRequest</i>	625
Интерфейс <i>HttpServletResponse</i>	626
Класс <i>HttpServlet</i>	627
Сеанс связи с сервлетом	634
Фильтры	637
Обращение к другим ресурсам	641
Вопросы для самопроверки	642
ГЛАВА 27. СТРАНИЦЫ JSP	643
Стандартные действия (теги) JSP	646
Язык записи выражений EL	649
Встроенные объекты JSP	650
Обращение к компоненту <i>JavaBean</i>	652
Выполнение апплета в браузере клиента	654
Передача управления	655
Пользовательские теги	655
Класс-обработчик пользовательского тега	657
Пользовательский тег с атрибутами	660
Пользовательский тег с телом	660
Обработка тела пользовательского тега	662
Обработка взаимодействующих тегов	664
Обработка исключений в пользовательских тегах	668
Обработка тегов средствами JSP	669
Стандартные библиотеки тегов JSTL	671
Библиотека <i>core</i>	671
Библиотека <i>xml</i>	675
Библиотека <i>fmt</i>	675
Библиотека <i>sql</i>	675
Библиотека <i>fn</i>	676
Frameworks	676
JavaServer Faces	677
Вопросы для самопроверки	681
ГЛАВА 28. СВЯЗЬ JAVA С ТЕХНОЛОГИЕЙ XML	683
Описание DTD	688
Пространства имен XML	690
Схема XML	692
Встроенные простые типы XSD	693
Вещественные числа	693
Целые числа	693
Строки символов	693
Дата и время	694

Двоичные типы	694
Прочие встроенные простые типы.....	694
Определение простых типов	694
Сужение	695
Список	696
Объединение	697
Описание элементов и их атрибутов	698
Определение сложных типов	698
Определение типа пустого элемента	699
Определение типа элемента с простым телом.....	699
Определение типа вложенных элементов.....	700
Определение типа со сложным телом	702
Пример: схема адресной книги.....	703
Безымянные типы.....	705
Пространства имен языка XSD.....	707
Включение файлов схемы в другую схему.....	709
Связь документа XML со своей схемой.....	711
Другие языки описания схем	711
Инструкции по обработке	712
Анализ документа XML	712
Анализ документов XML с помощью SAX2.....	713
Связывание данных XML с объектами Java.....	720
Объекты данных JDO	721
Анализ документов XML с помощью DOM API.....	722
Интерфейс <i>Node</i>	723
Интерфейс <i>Document</i>	725
Интерфейс <i>Element</i>	726
Другие DOM-парсеры.....	729
Преобразование дерева объектов в XML	729
Таблицы стилей XSL	731
Преобразование документа XML в HTML	734
Вопросы для самопроверки.....	735

СПИСОК ЛИТЕРАТУРЫ	737
--------------------------------	------------

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ	739
-----------------------------------	------------

Введение

Книга, которую вы держите в руках, возникла из курса лекций, читаемых автором в течение последних лет студентам младших курсов. 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 сгусток практического опыта, накопленного автором и его студентами с 1996 г.;
- содержит ответы на часто задаваемые вопросы, последних "компьютерщики" 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, тогда автор будет считать, что его скромный труд не пропал даром.

Прошло несколько лет с того дня, когда были написаны эти строки. Все случилось так, как я и написал. Разошлось два издания книги. 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 наводило 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. Со времени выхода первого издания Java прочно утвердилась на Web-серверах, проникла в сотовые телефоны и другие мобильные устройства. 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.

Все это привело к необходимости сделать третье издание, дополнив книгу новым материалом.

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 всюду 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, уже появились и другие языки, 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 под-процессы (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, тех ее разделов, 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 совершенствование 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, и собирается выпускать целую линейку все более мощных Java-процессоров. Есть уже и Java-процессоры 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.

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 компонуемых 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 Sun Microsystems — создатель 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). 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 добавления электронной подписи `javakey`;
- 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 постоянно развивает и обновляет JDK, каждый год появляются 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. J2SE 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 J2SE JDK 1.5.0 называется *jdk-1_5_0-windows-i586.exe*, а файл с документацией называется *jdk-1_5_0-doc.zip*.

Все это распаковывается в несколько сотен мегабайт дискового пространства. Данная версия требует процессор Pentium III и выше и не менее 64 Мбайт оперативной памяти.

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.

На время выхода этого издания книги версия J2SE 5.0 продолжала развиваться, вышло обновление JDK 1.5.0_13, но уже появилась новая, шестая версия.

В шестой версии, вышедшей в начале 2007 г., из названия технологии убрали цифру 2. Теперь пишут *Java Platform, Standard Edition 6*, сокращенно — Java SE 6 и JDK 6. Впрочем, во внутрифирменной документации остается прежнее обозначение, например третье обновление обозначается JDK 1.6.0_03.

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 устанавливается после Java SE JDK. Впрочем, на том же сайте есть полная версия архива, содержащая в себе Java SE JDK.

Кроме JDK компания Sun 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 Sun 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 Sun распространяет этот набор и отдельным файлом.

Как установить JDK?

Набор JDK упаковывается в самораспаковывающийся архив. Раздобыв каким-либо образом этот архив: скачав из Интернета, с сайта <http://java.sun.com/javase/downloads/> или какого-то другого адреса, получив компакт-диск, 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/jdk1.6.0_03`. Каталог и его название можно поменять, место и название установки не имеют значения.

После установки вы получите каталог с названием, например, `jdk1.6.0_03`, а в нем подкаталоги:

- `bin` с исполнимыми файлами;
- `db` с небольшой базой данных;
- `demo` с примерами программ;
- `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;
- `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/jdk1.6.0_03/bin. Некоторые программы, использующие Java, требуют определить и специальную переменную окружения JAVA_HOME, содержащую путь к каталогу установки JDK, например /usr/java/jdk1.6.0_03.

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 **MS-DOS Prompt** в Windows 95/98/ME или окна **Command Prompt** в Windows NT/2000/Server 2003/XP. 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 тех-

нологии 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 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.

Если работа из командной строки, столь милая сердцу "юникоидов", кажется вам несколько устаревшей, используйте для разработки интегрированную среду.

Интегрированные среды Java

Сразу же после создания Java, уже в 1996 г., появились интегрированные среды разработки программ IDE (Integrated Development Environment) для Java, и их число все время возрастает. Некоторые из них, такие как Eclipse, IntelliJ IDEA, NetBeans, являются просто интегрированными оболочками над JDK, вызывающими из одного окна текстовый редактор, компилятор и интерпретатор. Эти интегрированные среды требуют предварительной установки JDK. Другие содержат JDK в себе или имеют собственный компилятор, например JBuilder фирмы CodeGear или IBM Rational Application Developer. Их можно устанавливать, не имея под руками JDK. Надо заметить, что перечисленные продукты написаны полностью на Java.

Большинство интегрированных сред являются средствами визуального программирования и позволяют быстро создавать пользовательский интерфейс, т. е. относятся к классу средств RAD (Rapid Application Development).

Выбор какого-либо средства разработки диктуется, во-первых, возможностями вашего компьютера, ведь визуальные среды требуют больших ресурсов; во-вторых, личным вкусом; в-третьих, уже после некоторой практики, достоинствами компилятора, встроенного в программный продукт.