

# ОГЛАВЛЕНИЕ

Введение. Программирование на языке Java . . . . .	9
Особенности языка Java . . . . .	9
Программное обеспечение . . . . .	11
Среда разработки NetBeans . . . . .	18
Создание нового проекта . . . . .	19
Компиляция и запуск программы на выполнение . . . . .	22
Заккрытие проекта . . . . .	24
Открытие существующего проекта . . . . .	25
О книге . . . . .	26
Обратная связь с автором . . . . .	27
Глава 1. Приступаем к программированию . . . . .	28
Первая программа . . . . .	28
Создание программы . . . . .	28
Анализ программного кода . . . . .	30
Общие замечания . . . . .	34
Вариации на тему первой программы . . . . .	35
Вывод в консольное окно . . . . .	39
Окно с полем ввода . . . . .	40
Создание окна с полем ввода . . . . .	41
Анализ программного кода . . . . .	42
Управление видом окна с полем ввода . . . . .	43
Консольный ввод . . . . .	45
Резюме . . . . .	50
Глава 2. Базовые типы и основные операторы . . . . .	52
Переменные . . . . .	52
Базовые типы . . . . .	52
Объявление и инициализация переменных . . . . .	55
Считывание значения переменной . . . . .	60
Литералы и управляющие символы . . . . .	68
Приведение типов . . . . .	69
Основные операторы . . . . .	72
Арифметические операторы . . . . .	72
Операторы сравнения . . . . .	73
Логические операторы . . . . .	74
Побитовые операторы . . . . .	75
Тернарный оператор . . . . .	77
Оператор присваивания . . . . .	78
Сокращенные формы оператора присваивания . . . . .	79
Резюме . . . . .	79

---

Глава 3. Знакомство с классами и объектами . . . . .	81
Классы и объекты . . . . .	81
Описание класса с полями . . . . .	82
Создание объекта . . . . .	83
Использование объектов . . . . .	85
Класс с методами . . . . .	87
Методы и конструкторы . . . . .	92
Перегрузка методов . . . . .	92
Конструктор . . . . .	98
Статические и закрытые члены класса . . . . .	102
Статические поля и методы . . . . .	103
Закрытые и открытые члены класса . . . . .	106
Резюме . . . . .	110
Глава 4. Управляющие инструкции . . . . .	113
Условный оператор . . . . .	113
Синтаксис условного оператора . . . . .	113
Использование условного оператора . . . . .	115
Вложенные условные операторы . . . . .	123
Операторы цикла . . . . .	132
Оператор цикла while . . . . .	132
Оператор цикла do-while . . . . .	139
Оператор цикла for . . . . .	144
Сравнение операторов цикла . . . . .	146
Оператор выбора . . . . .	149
Резюме . . . . .	154
Глава 5. Массивы . . . . .	157
Одномерные массивы . . . . .	157
Создание одномерного массива . . . . .	157
Инициализация одномерного массива . . . . .	164
Оператор цикла for по коллекции . . . . .	171
Присваивание массивов . . . . .	174
Двумерные массивы . . . . .	177
Создание двумерного массива . . . . .	178
Инициализация двумерного массива . . . . .	182
Массив со строками разной длины . . . . .	185
Массивы и методы . . . . .	188
Резюме . . . . .	193
Глава 6. Наследование . . . . .	195
Реализация наследования . . . . .	195
Создание подкласса . . . . .	196
Конструктор подкласса . . . . .	203
Наследование и закрытые члены . . . . .	211
Наследование, пакеты и уровни доступа . . . . .	214

---

---

Переопределение методов . . . . .	220
Общие принципы переопределения методов . . . . .	221
Вызов разных версий метода . . . . .	223
Виртуальность методов и конструкторов . . . . .	227
Перегрузка и переопределение методов . . . . .	230
Метод toString() . . . . .	232
Объект подкласса и переменная суперкласса . . . . .	235
Резюме . . . . .	239
Глава 7. Абстрактные классы и интерфейсы . . . . .	241
Абстрактные классы и методы . . . . .	241
Интерфейсы . . . . .	249
Реализация интерфейса . . . . .	250
Интерфейсные переменные . . . . .	254
Методы с кодом по умолчанию . . . . .	257
Расширение интерфейсов . . . . .	262
Наследование классов и реализация интерфейсов . . . . .	267
Резюме . . . . .	270
Глава 8. Использование классов и объектов . . . . .	272
Методы и объекты . . . . .	272
Механизм передачи аргументов методам . . . . .	272
Передача аргументом объекта . . . . .	274
Объект как результат метода . . . . .	279
Объекты и наследование . . . . .	284
Фабрика объектов . . . . .	284
Конструктор создания копии . . . . .	287
Массивы и объекты . . . . .	291
Массив как поле . . . . .	292
Массив объектов . . . . .	295
Цепочка объектов . . . . .	298
Внутренние классы . . . . .	303
Анонимные классы . . . . .	307
Создание анонимного класса путем наследования абстрактного суперкласса . . . . .	308
Создание анонимного класса через реализацию интерфейса . . . . .	311
Резюме . . . . .	314
Глава 9. Обобщенные типы данных . . . . .	315
Знакомство с обобщенными классами . . . . .	315
Общие принципы использования обобщенных классов . . . . .	316
Пример создания обобщенного класса . . . . .	317
Обобщенный класс с несколькими параметрами . . . . .	320
Обобщенные методы . . . . .	323
Создание статического обобщенного метода . . . . .	323
Создание нестатического обобщенного метода . . . . .	326

---

Обобщенные классы и наследование . . . . .	328
Суперкласс на основе обобщенного класса . . . . .	329
Ограничение наследования для обобщенного типа . . . . .	332
Обобщенные интерфейсы . . . . .	337
Создание обобщенного класса на основе интерфейса . . . . .	337
Создание обычного класса на основе обобщенного интерфейса . . . . .	340
Обобщенные подстановки . . . . .	343
Знакомство с обобщенными подстановками . . . . .	343
Обобщенные подстановки с ограничениями . . . . .	347
Резюме . . . . .	351
Глава 10. Лямбда-выражения . . . . .	353
Знакомство с лямбда-выражениями . . . . .	353
Синтаксис лямбда-выражения . . . . .	353
Функциональные интерфейсы . . . . .	355
Альтернативный подход . . . . .	359
Несколько интерфейсов и ссылка на метод . . . . .	362
Ссылка на метод и конструктор . . . . .	365
Ссылка на метод объекта . . . . .	365
Ссылка на нестатический метод класса . . . . .	370
Ссылка на статический метод . . . . .	373
Ссылка на конструктор . . . . .	376
Ссылка на перегруженный метод . . . . .	378
Использование лямбда-выражений . . . . .	380
Передача лямбда-выражения аргументом методу . . . . .	381
Лямбда-выражение и результат метода . . . . .	385
Лямбда-выражение и поле объекта . . . . .	389
Резюме . . . . .	392
Глава 11. Обработка исключительных ситуаций . . . . .	393
Перехват и обработка ошибок . . . . .	393
Пример обработки исключения . . . . .	393
Принципы обработки исключений . . . . .	397
Вложенные try-catch блоки . . . . .	406
Использование объекта исключения . . . . .	412
Генерирование исключений . . . . .	414
Контролируемые и неконтролируемые исключения . . . . .	416
Создание пользовательских классов исключений . . . . .	423
Резюме . . . . .	426
Глава 12. Многопоточное программирование . . . . .	428
Знакомство с потоками . . . . .	428
Способы создания дочерних потоков . . . . .	430
Явная реализация интерфейса Runnable . . . . .	430
Создание потока с использованием анонимного класса . . . . .	434
Создание потока с использованием лямбда-выражения . . . . .	437
Наследование класса Thread . . . . .	439

---

Работа с потоками . . . . .	441
Главный поток . . . . .	441
Методы для работы с потоками . . . . .	443
Создание нескольких потоков . . . . .	444
Создание демон-потока . . . . .	449
Синхронизация потоков . . . . .	454
Резюме . . . . .	460
Глава 13. Приложения с графическим интерфейсом . . . . .	462
Принципы создания приложений с интерфейсом . . . . .	462
Создание окна . . . . .	464
Пустое окно . . . . .	464
Альтернативный способ создания окна . . . . .	467
Окно с кнопкой . . . . .	469
Явное использование объекта обработчика . . . . .	469
Принципы обработки событий . . . . .	474
Обработчик на основе анонимного класса . . . . .	477
Обработчик на основе лямбда-выражения . . . . .	480
Обработчик на основе объекта окна . . . . .	484
Создание класса для кнопки . . . . .	490
Резюме . . . . .	494
Глава 14. Обработка событий . . . . .	496
Классы компонентов и событий . . . . .	496
Классы графических компонентов . . . . .	496
Классы событий . . . . .	499
Использование текстового поля . . . . .	500
Считывание значения поля . . . . .	500
Использование общего обработчика . . . . .	506
Обработчик для поля . . . . .	514
Классы-адаптеры . . . . .	524
Основные классы-адаптеры . . . . .	524
Использование классов-адаптеров . . . . .	525
Резюме . . . . .	530
Глава 15. Графические компоненты . . . . .	531
Раскрывающийся список . . . . .	531
Список выбора . . . . .	541
Группа переключателей . . . . .	546
Опции и другие элементы . . . . .	552
Резюме . . . . .	572
Глава 16. Меню и панель инструментов . . . . .	573
Меню и панель инструментов . . . . .	573
Использование меню . . . . .	573
Панель инструментов . . . . .	574
Менеджеры компоновки и текстовая панель . . . . .	575
Менеджеры компоновки . . . . .	575
Текстовая панель . . . . .	576

---

Использование меню и панели инструментов . . . . .	577
Постановка задачи . . . . .	577
Анализ возможностей программы . . . . .	578
Программный код примера . . . . .	584
Анализ программного кода . . . . .	594
Резюме . . . . .	606
Глава 17. Апплеты . . . . .	607
Знакомство с апплетами . . . . .	607
Общие принципы реализации апплета . . . . .	607
Добавление апплета в веб-документ . . . . .	609
Программный код апплета . . . . .	612
Компиляция файла . . . . .	615
Настройки безопасности . . . . .	621
Апплеты и обработка событий . . . . .	622
Пример обработки событий в апплете . . . . .	623
Передача апплету параметров . . . . .	634
Апплет с элементами управления . . . . .	645
Резюме . . . . .	662
Глава 18. Файлы и аргументы командной строки . . . . .	664
Аргументы командной строки . . . . .	664
Работа с файлами . . . . .	671
Получение информации о файле . . . . .	672
Чтение из файла и запись в файл . . . . .	678
Средства выбора файлов . . . . .	688
Резюме . . . . .	696
Заключение. Еще немного о Java . . . . .	697
Предметный указатель . . . . .	698

# Введение

## ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ JAVA

*Это же вам не лезгинка, а твист!*

*Из к/ф «Кавказская пленница»*

Среди языков программирования Java — самый популярный и самый востребованный. Эта книга о том, как программировать на Java.

### Особенности языка Java

*Тот, кто нам мешает, тот нам и поможет.*

*Из к/ф «Кавказская пленница»*

История Java началась в 1990-х годах, когда группа инженеров компании Sun Microsystems в рамках проекта под названием Green приступила к разработке достаточно универсального, компактного и платформенно-независимого языка программирования Oak, предназначенного для использования в бытовых устройствах. В процессе реализации проекта изменились не только основные приоритеты, но и название языка программирования. Как бы то ни было, в 1995 году мир познакомился с языком программирования Java.



#### НА ЗАМЕТКУ

С появления первой версии Java было несколько обновлений платформы. На момент написания книги актуальной является версия Java 8. Именно она обсуждается в книге.

Язык Java, хоть и не без труда, но завоевал свое «место под солнцем». Сегодня Java прочно удерживает позиции самого востребованного языка программирования. Успеху языка способствовало бурное развитие интернет-технологий. Дело в том, что для Java-программ характерна

высокая степень универсальности и независимости от аппаратного обеспечения. Это важно при создании программ, ориентированных на работу в Сети, поскольку конечные пользователи используют различные операционные системы и оборудование. К тому же важную роль сыграла применимость Java для программирования всевозможных мобильных устройств. Поэтому нет ничего удивительного, что значительная доля коммерческих и свободно распространяемых программ написана на языке Java. Соответственно, спрос на программистов, работающих с языком Java, стабильно высок, а общие тенденции таковы, что он останется высоким и в ближайшее время.

Универсальность программ, написанных на языке Java, базируется на использовании *виртуальной машины*. Это такой специфический «посредник», под управлением которого выполняется байт-код, получаемый при компиляции программы. Здесь нужны пояснения.

После того как программа написана, она компилируется. Обычно в результате компиляции программы создается исполнительный файл с машинным кодом, который и выполняется, когда необходимо выполнить программу. Проще говоря, при компиляции команды, понятные для программиста, переводятся на «язык», понятный для компьютера. Если речь идет о программе, написанной на языке Java, то все происходит похожим образом, но с некоторыми особенностями. Самое важное, что в результате компиляции Java-программы получается не машинный код, а промежуточный *байт-код*. Это нечто среднее между машинным кодом и кодом программы. Если машинный код, как правило, выполняется под управлением операционной системы, то байт-код выполняется под управлением специальной программы, которая называется виртуальной машиной (или виртуальной Java-машиной). Понятно, что такую программу на компьютер предварительно следует установить.

Возникает вопрос: а в чем же выигрыш от использования виртуальной машины и как все описанное влияет на универсальность кодов? Выигрыш в том, что при написании кода можно абстрагироваться от особенностей операционной системы и аппаратного обеспечения, используемых конечным пользователем. Эти особенности учитываются — но учитываются на уровне виртуальной машины. Именно виртуальная машина при выполнении байт-кода «принимает в расчет» особенности операционной системы и аппаратного обеспечения компьютера, на котором выполняется программа.



**НА ЗАМЕТКУ**

---

Допустим, есть программа, написанная на языке C++. При ее компиляции получается машинный код, который для разных операционных систем будет разным. Если компилируется программа, написанная на Java, то получающийся в результате байт-код не зависит от операционной системы, которая установлена на компьютере, — он будет одним и тем же для разных операционных систем. Но вот виртуальная машина для каждой операционной системы своя. Разница в операционных системах «учитывается», когда на компьютер устанавливается виртуальная машина.

Описанный выше механизм, в общем и целом, обеспечивает высокую степень универсальности программ, написанных на Java. Особенно это заметно при создании программ с графическим интерфейсом.

**НА ЗАМЕТКУ**

---

Забегая вперед, отметим, что в плане создания приложений с графическим интерфейсом язык Java особенно хорош.

Есть еще один важный аспект, касающийся языка Java, на который сразу хочется обратить внимание. Язык Java — полностью *объектно-ориентированный* язык. Сказанное означает, что для написания даже самой маленькой и самой простой программы придется описать по меньшей мере один *класс*. Это автоматически создает некоторые трудности в освоении премудростей Java. Особенно сложно тем, кто не имеет опыта программирования. Ведь фактически сразу, с первых шагов, приходится знакомиться с концепцией *объектно-ориентированного программирования* (сокращенно ООП), которая, надо признать, не самая тривиальная. Но паниковать не стоит — мы найдем способ донести нужные сведения даже до самых неподготовленных читателей. Главное, чтобы было желание освоить язык Java.

## Программное обеспечение

*Будь проклят тот день, когда я сел за баранку  
этого пылесоса!*

*Из к/ф «Кавказская пленница»*

Если подойти к вопросу формально, то сам по себе язык Java — набор правил, в соответствии с которыми составляется программный код.

Но программы пишутся для того, чтобы они выполнялись. А раз так, то нам понадобится специальное программное обеспечение. Хорошая новость в том, что все необходимое программное обеспечение может быть получено совершенно свободно, просто, легально и бесплатно.

### **i**    **НА ЗАМЕТКУ**

---

Понятно, что есть и коммерческие приложения, предназначенные для написания программ в Java. Но для решения тех задач, которые мы ставим перед собой, стандартного свободно распространяемого программного обеспечения более чем достаточно.

Что же нам понадобится? В принципе, можно обойтись минимальными средствами в виде пакета приложений JDK (сокращение от *Java Development Kit* — средства разработки Java). В состав пакета JDK, кроме прочего, входит компилятор, всевозможные библиотеки, документация и исполнительная система JRE (сокращение от *Java Runtime Environment* — среда выполнения Java) — фактически виртуальная машина Java. Пакет приложений JDK распространяется бесплатно компанией Oracle (сайт компании [www.oracle.com](http://www.oracle.com)).

### **i**    **НА ЗАМЕТКУ**

---

В свое время разработчика Java, компанию Sun Microsystems, поглотила корпорация Oracle. Так что теперь поддержкой Java-технологий занимается именно она.

Ситуация такая, что без JDK нам не обойтись, но и ограничиваться только пакетом JDK не стоит. Если ограничиться только пакетом JDK, то программные коды придется набирать в текстовом редакторе, а компилировать программу придется «вручную» из командной строки. Поэтому желательно использовать *среду разработки* (сокращенно IDE от *Integrated Development Environment*).

Среда разработки содержит редактор кодов, отладчик, позволяющий в интерактивном режиме отслеживать код на наличие синтаксических ошибок, набор прочих утилит, позволяющих сделать процесс написания, тестирования и компиляции программ простым, удобным и где-то даже комфортным (насколько это вообще возможно). Проще говоря, среда разработки должна использоваться — тем более, если учесть, что имеются очень приличные бесплатно распространяемые среды разработки.

Мы остановим свой выбор на среде разработки NetBeans. Среда распространяется бесплатно, ее установочные файлы можно загрузить на сайте поддержки проекта [www.netbeans.org](http://www.netbeans.org).

Далее кратко рассмотрим, какое программное обеспечение и откуда следует загрузить перед тем, как мы непосредственно приступим к изучению языка программирования Java.

Задача наша простая:

- загрузить и установить пакет приложений JDK;
- после установки JDK следует загрузить и установить среду разработки NetBeans.

Действия по загрузке и установке программного обеспечения выполняются именно в том порядке, как они перечислены выше.



## ДЕТАЛИ

---

Среда разработки NetBeans в процессе работы с программными кодами обращается к системе JDK. Если систему JDK установить до установки NetBeans, то все настройки среды разработки, связанные с JDK, выполняются автоматически. Если систему JDK устанавливать после установки среды разработки NetBeans, то настройки среды разработки придется выполнять самостоятельно.

Итак, в первую очередь устанавливаем пакет JDK, для чего предварительно с сайта компании Oracle загружаем установочные файлы. На рис. В.1 показано окно браузера, открытое на странице [www.oracle.com](http://www.oracle.com).

В разделе загрузок (вкладка **Downloads**) следует найти ссылку на загрузку программного обеспечения для Java.



## ДЕТАЛИ

---

Существует несколько редакций, или дистрибутивов, Java. Например, платформа Java для создания программного обеспечения уровня больших корпораций называется Java Enterprise Edition (сокращенно Java EE). Стандартная редакция Java предназначена для создания пользовательских приложений и называется Java Standard Edition (сокращенно Java SE). Также существует редакция Java Micro Edition (сокращенно Java ME), используемая при создании приложений для всевозможных мобильных устройств. Мы будем использовать стандартную редакцию Java Standard Edition (или Java SE).

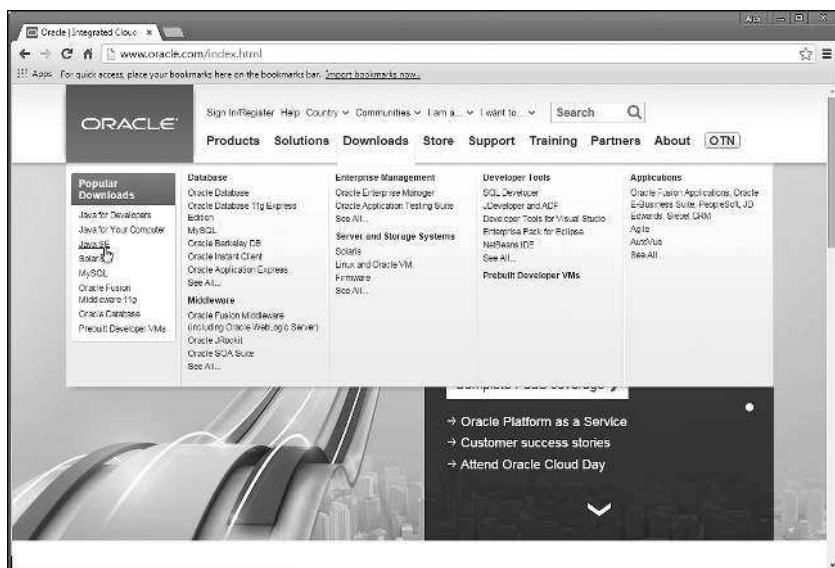


Рис. В.1. Окно браузера открыто на странице [www.oracle.com](http://www.oracle.com) корпорации Oracle

После щелчка по гиперссылке для загрузки программного обеспечения для работы с Java, переходим на еще одну страницу, с которой собственно и выполняется загрузка (рис. В.2).

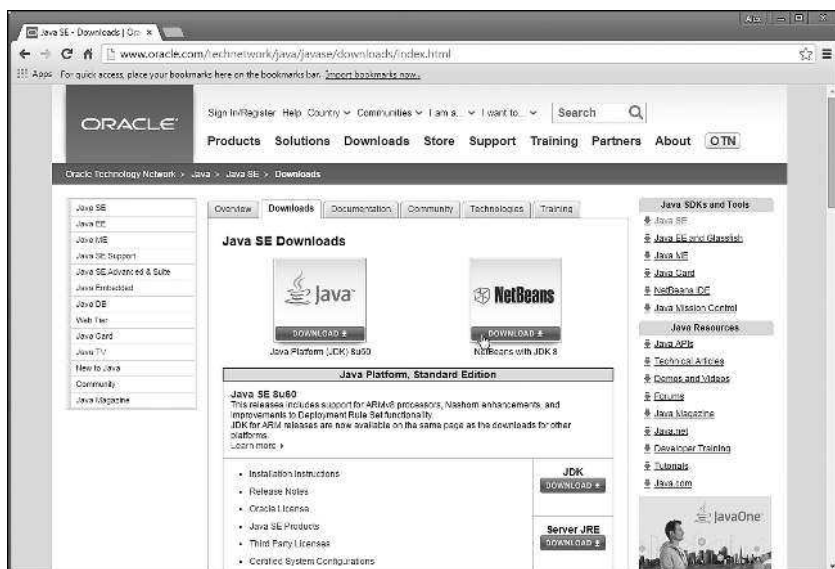
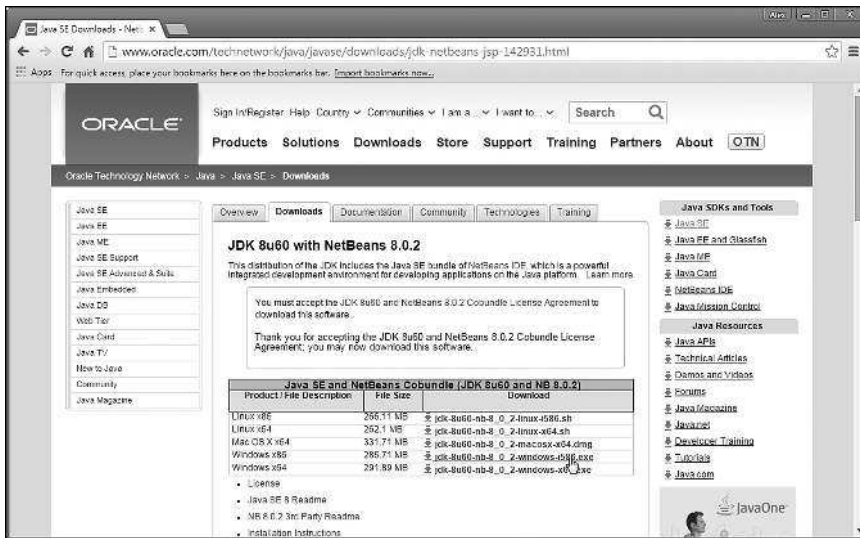


Рис. В.2. Окно браузера открыто на странице загрузки установочного файла пакета JDK и среды разработки NetBeans

В принципе здесь можно просто загрузить пакет JDK, но обычно предлагается еще и способ загрузки, при котором пакет JDK идет в комплекте со средой разработки NetBeans. Это, пожалуй, лучший вариант, который позволяет последовательно установить на компьютер JDK и NetBeans из одного установочного файла.

В процессе загрузки предлагается выбрать тип установочного файла в соответствии с используемой операционной системой. Ситуация проиллюстрирована на рис. В.3.



**Рис. В.3.** Выбор установочного файла в соответствии с используемой операционной системой



### НА ЗАМЕТКУ

Внешний вид сайтов, в том числе и сайт корпорации Oracle, время от времени меняется, поэтому не исключено, что для поиска страницы загрузки программного обеспечения придется проявить некоторую изобретательность.

Если со страницы корпорации Oracle загружается установочный файл сразу для JDK и NetBeans, то все, что остается — выполнить установку. Выполняется она просто, так что комментировать здесь особо нечего (совсем предлагаемыми в процессе установки настройками лучше согласиться). Если же по каким-то причинам загружается и устанавливается

только пакет JDK, то придется отдельно загрузить еще и установочный файл для среды разработки NetBeans. В этом случае переходим на страницу [www.netbeans.org](http://www.netbeans.org) проекта NetBeans, как показано на рис. В.4.



**Рис. В.4.** Страница [www.netbeans.org](http://www.netbeans.org) проекта NetBeans

Затем переходим к странице загрузки установочных файлов среды NetBeans, на которой следует выбрать версию среды для загрузки (рис. В.5).

Разные версии среды разработки отличаются, кроме языка интерфейса, поддерживаемыми технологиями (сюда включаются разные редакции платформы Java и еще несколько дополнительных языков программирования). Версия должна быть такой, чтобы в ней поддерживалась редакция Java SE. Если возможности аппаратного обеспечения позволяют, можно порекомендовать версию среды разработки с максимальным набором поддерживаемых технологий.

После выбора версии среды разработки NetBeans загружается установочный файл, после чего устанавливается среда разработки. На этом предварительная подготовка к написанию программ на Java завершается. Заметим лишь, что еще одна полезная страница находится по адресу [www.java.com](http://www.java.com). На рис. В.6 показано окно браузера, открытое на данной странице.

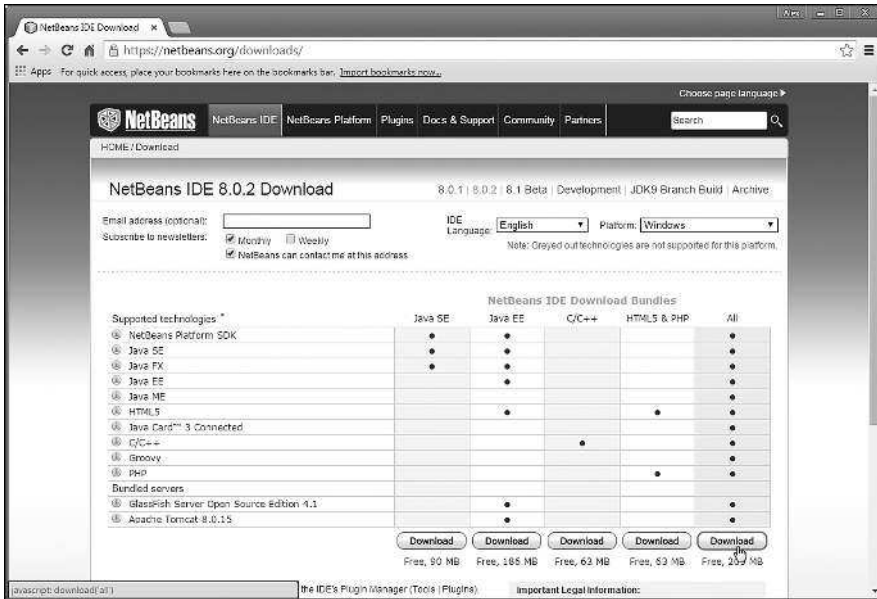


Рис. В.5. Выбор версии среды разработки NetBeans



Рис. В.6. Окно браузера открыто на странице www.java.com поддержки Java

На странице можно загружать обновления платформы Java, которые появляются достаточно часто.