Васильев А.Н.

ПРОГРАММИРОВАНИЕ НА JAVA

ДЛЯ НАЧИНАЮЩИХ

- Объектно-ориентированное программирование «с нуля»
- Средства разработки и подробные инструкции для начинающих
- От классов и объектов к созданию приложений с графическим интерфейсом
- Разбор программного кода и наглядные примеры
- Прекрасно подходит для самостоятельного обучения

РОССИЙСКИЙ КОМПЬЮТЕРНЫЙ БЕСТСЕЛЛЕР

УДК 004.43
ББК 32.973.2-018.1
В19

Васильев, Алексей Николаевич.

В19 Программирование на Java для начинающих / Алексей Васильев. – Москва : Издательство «Э», 2017. – 704 с. – (Российский компьютерный бестселлер).

ISBN 978-5-699-89475-8

Полный спектр сведений о языке Java с примерами и разбором задач от автора учебников-бестселлеров по языкам программирования Алексея Васильева. С помощью этой книги освоить язык Java сможет каждый желающий – от новичка до специалиста.

УДК 004.43
ББК 32.973.2-018.1

ISBN 978-5-699-89475-8

ОГЛАВЛЕНИЕ

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.

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).

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.

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.

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).

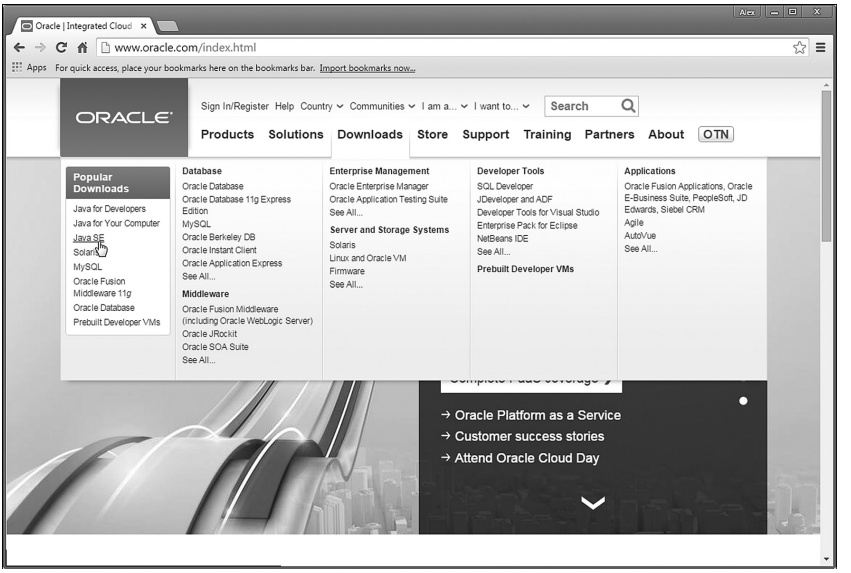

Рис. В.1. Окно браузера открыто на странице www.oracle.com корпорации Oracle

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.

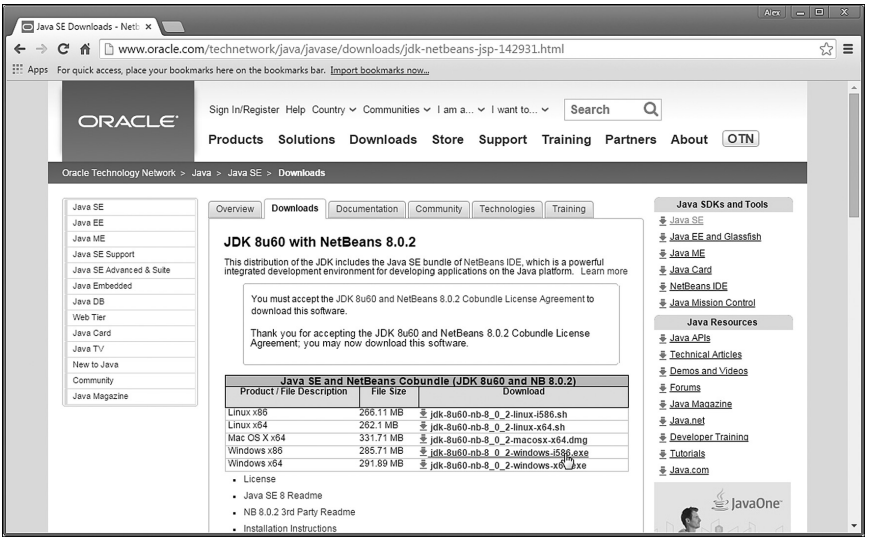

Рис. В.3. Выбор установочного файла в соответствии с используемой операционной системой

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 (со всеми 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 проекта NetBeans, как показано на рис. В.4.

Рис. В.4. Страница 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. На рис. В.6 показано окно браузера, открытое на данной странице.

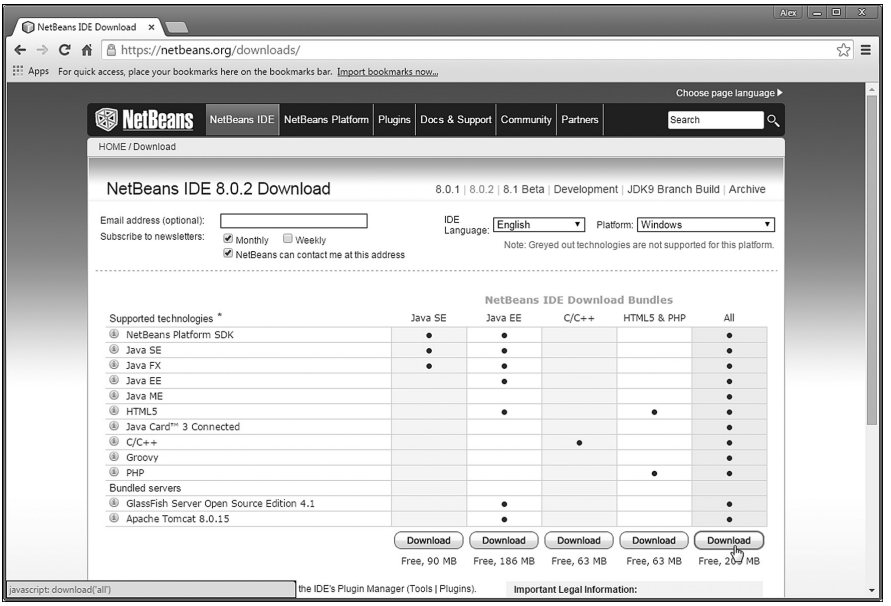

Рис. В.5. Выбор версии среды разработки NetBeans

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

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

Среда разработки NetBeans

Замечательная идея! Что ж она мне самому в голову не пришла?

Из к/ф «Ирония судьбы, или С легким паром!»

Все проекты (программы), рассматриваемые в книге, тестировались в среде разработки NetBeans. Читателю рекомендуется использовать эту же среду разработки (с другой стороны, не все рекомендации обязательны к исполнению).

В любом случае, уместно уделить хоть небольшое внимание приложению NetBeans.

На рис. В.7 показано, как выглядит пустое окно (без открытого в нем проекта) среды разработки NetBeans.

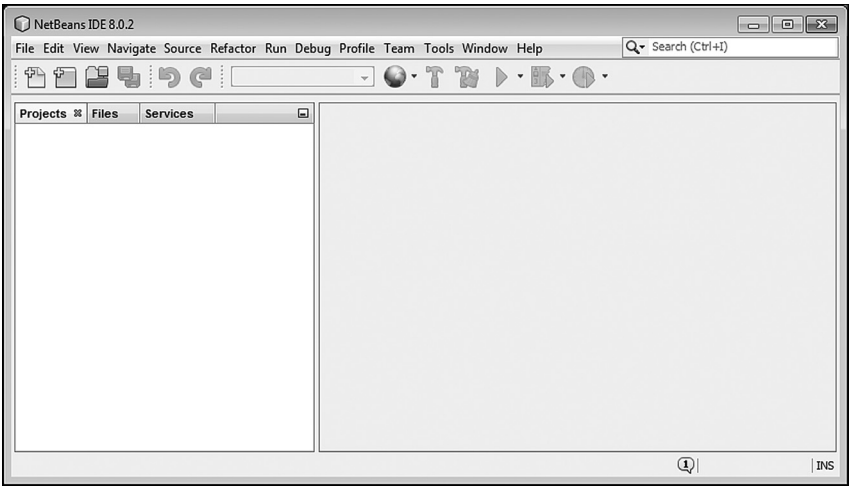

Рис. В.7. Внешний вид окна среды разработки NetBeans

Особенности работы с приложением NetBeans могут быть предметом отдельной книги. В наши планы это не входит. Мы рассмотрим лишь некоторые наиболее важные операции, которые читателю предстоит выполнять в процессе написания программных кодов.

Чтобы не отвлекаться в основной части книги на описание манипуляций с элементами интерфейса приложения NetBeans, сделаем это заблаговременно. А именно интерес представляют такие действия:

- создание нового проекта (выполняется при написании новой программы);
- компиляция и запуск на выполнение программы;
- закрытие открытого проекта (программы);
- открытие уже существующего проекта.

Теперь обо всем по порядку.

Создание нового проекта

Создание новой программы означает создание нового проекта. Чтобы создать новый проект, в меню **File** выбираем команду **New Project**, как показано на рис. В.8.

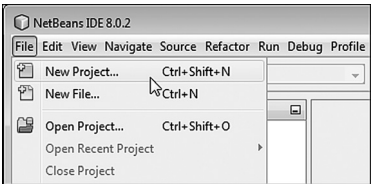

Рис. В.8. Создание проекта с помощью команды **New Project** из меню **File**

Альтернативный способ создания проекта — щелкнуть кнопку с желтой папкой и зеленым знаком «плюс» на панели инструментов, как показано на рис. В.9, или нажать комбинацию клавиш <Ctrl>+<Shift>+<N>.

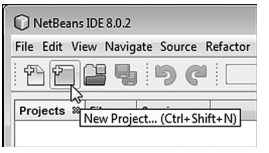

Рис. В.9. Для создания нового проекта выполняется щелчок на кнопке на панели инструментов

На следующем этапе открывается диалоговое окно **New Project**, в котором есть два раздела: **Categories** и **Projects** (рис. В.10).

В разделе **Categories** выбираем позицию **Java**, а в разделе **Projects** выбираем позицию **Java Application**, после чего щелкаем кнопку **Next**. В результате откроется диалоговое окно **New Java Application**, представленное на рис. В.11.

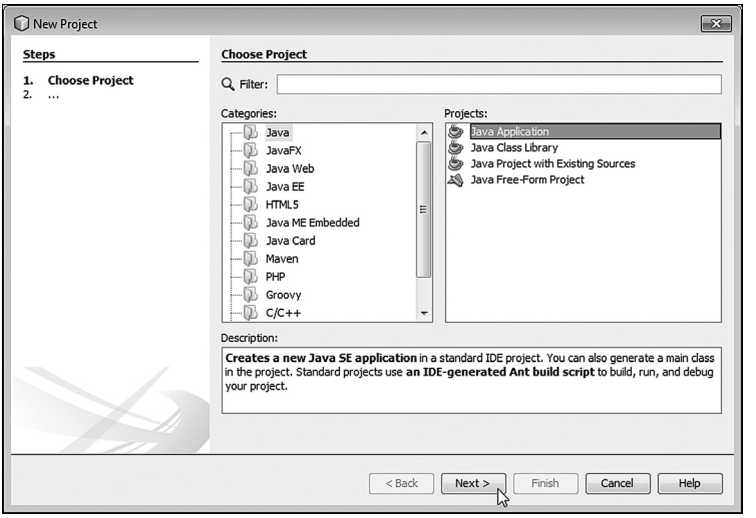

Рис. В.10. Выбор типа проекта в окне **New Project**

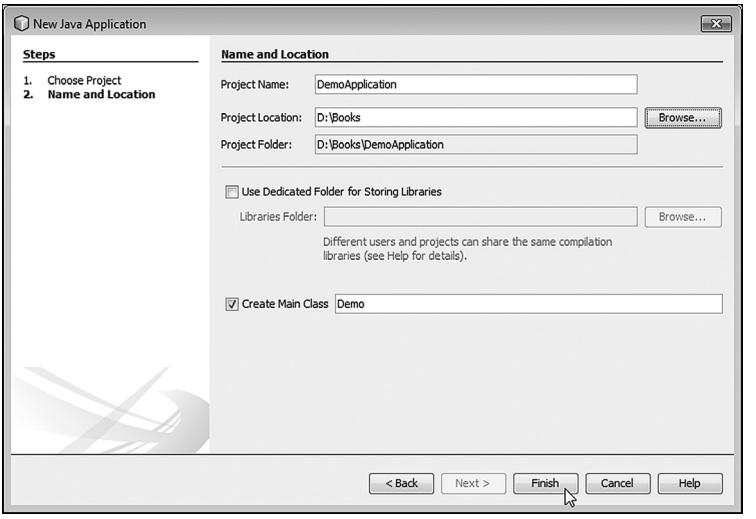

Рис. В.11. Определение основных параметров проекта в окне **New Java Application**

В окне **New Java Application** выполняются такие настройки приложения, как его название и место хранения файлов приложения. В частности, в поле **Project Location** указывается каталог, в котором хранятся файлы проекта. Для удобства выбора имеется кнопка **Browse**. Щелчок на кнопке приводит к отображению диалогового окна выбора каталога. В поле **Project Name** задается название проекта (здесь мы назвали

проект DemoApplication). Поле **Project Folder** заполняется автоматически, а вот поле **Create Main Class** лучше заполнить: поле активно при установленном флажке в опции слева от названия поля, а в самом поле вводится название для главного класса программы (в рассматриваемом примере главный класс называется Demo).

ДЕТАЛИ

Как отмечалось ранее, любая программа на языке Java содержит описание хотя бы одного класса. Если речь не идет об апплетах, то среди классов программы есть один, который называется главным классом программы. Класс содержит код главного метода, и этот код выполняется при выполнении программы. Проще говоря, главный класс программы содержит код, выполняемый при запуске программы. В поле **Create Main Class** указывается название для главного класса программы. Вообще, название этого класса задается непосредственно в программном коде, но для компиляции программы оно должно совпадать с именем, указанным для главного класса при создании проекта.

По умолчанию в поле **Create Main Class** предлагается текстовая строка в формате имя.имя (имя, точка, затем еще одно имя). Если оставить в поле такой «точечный» текст, то первое имя (до точки) является именем пакета, а второе имя (после точки) является именем главного класса. Если так, то код программы следует начинать с инструкции package, после которой указывается имя пакета (первое имя перед точкой). Например, если бы в поле **Create Main Class** было указано mypack.Demo, то код программы должен был бы начинаться инструкцией package mypack (заканчивается точкой с запятой). Мы будем обсуждать работу с пакетами, но немного позже. Пока же рекомендуется в поле **Create Main Class** просто указать имя главного класса.

После щелчка на кнопке **Finish** создание нового проекта завершено. На рис. В.12 показано окно среды разработки с открытым в нем вновь созданным проектом.

В правой части окна среды разработки отображается внутреннее окно редактора кодов с кодом главного класса (окно называется Demo.java). В новом проекте отображается шаблонный код, который на самом деле обычно удаляется, и вместо него вводится тот код, который нужен.

НА ЗАМЕТКУ

Если по каким-то причинам код главного класса не отображается, в левой верхней части окна среды разработки находим внутреннее

окно с названием **Projects**. В нем есть раскрывающийся пункт с названием приложения. Если раскрыть этот пункт, можно увидеть позицию с названием файла Demo.java для главного класса. Двойной щелчок на названии Demo.java приводит к отображению кода главного класса.

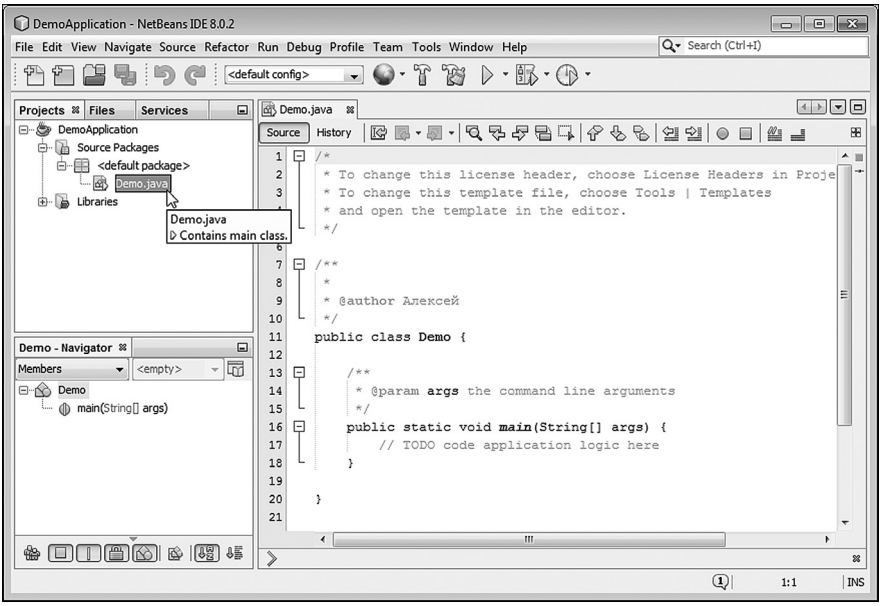

Рис. В.12. Окно среды разработки NetBeans с открытым новым проектом

Осталось набрать код программы, откомпилировать его и запустить на выполнение.

Компиляция и запуск программы на выполнение

Для тестирования функциональности приложения NetBeans в редакторе кодов вводим очень простой код, который есть в листинге В.1.

Листинг В.1. Программный код проекта DemoApplication

```
class Demo{
  public static void main(String[] args){
    System.out.println("Java & NetBeans");
  }
}
```

Как выглядит окно приложения NetBeans с введенным кодом в редакторе кодов, показано на рис. В.13.

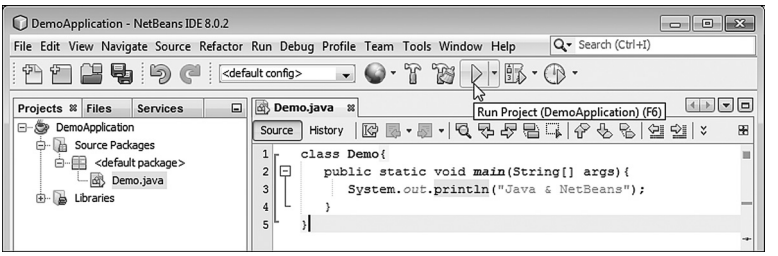

Рис. В.13. Компиляция программы и запуск ее на выполнение

ДЕТАЛИ

На данном этапе смысл команд из приведенного выше программного кода не так уж и важен. Анализировать коды мы будем в основной части книги. Пока же отметим, что инструкцией `class Demo` начинается описание класса. Само описание заключается в фигурные скобки { и }. В классе описывается главный метод программы, который называется `main()`. В теле метода выполняется всего одна команда `System.out.println("Java & NetBeans")`, которой в окне вывода отображается сообщение `Java & NetBeans`.

Для компиляции программы и запуска ее на выполнение щелкаем кнопку с зеленой стрелкой на панели инструментов (см. рис. В.13).

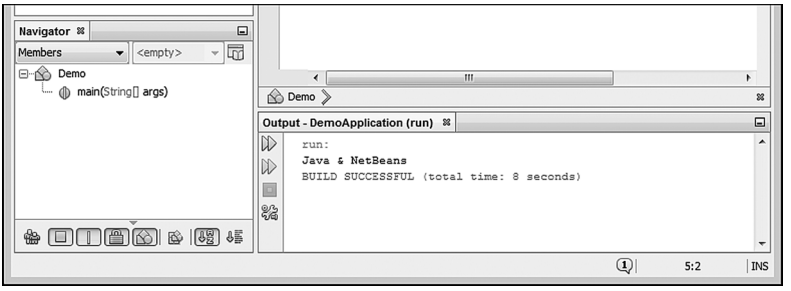

Рис. В.14. Результат выполнения программы отображается во внутреннем окне **Output**

Если компиляция пройдет удачно, в нижней части окна среды разработки во внутреннем окне **Output** появляется сообщение `Java & NetBeans`, как это показано на рис. В.14.

Проще говоря, результат выполнения программы такой:

Результат выполнения программы (из листинга В.1)

```
Java & NetBeans
```

Заметим, что откомпилировать и запустить программу на выполнение также можно с помощью команды **Run Project** из меню **Run** (рис. В.15) или нажав клавишу F6.

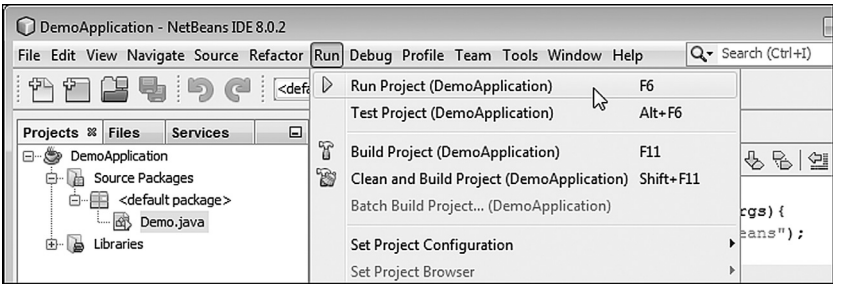

Рис. В.15. Альтернативный способ компиляции и запуска программы на выполнение с помощью команды **Run Project** из меню **Run**

ДЕТАЛИ

Файл с Java-программой имеет расширение .java. В общем случае программа, написанная на языке Java, может содержать несколько классов. При компиляции для каждого класса создается отдельный файл. Название таких файлов совпадает с названиями соответствующих классов, а расширения у файлов .class.

Закрытие проекта

Чтобы закрыть проект, достаточно его выделить во внутреннем окне **Projects**, щелкнуть правой кнопкой мыши и в открывшемся контекстном меню выбрать команду **Close**, как показано на рис. В.16.

Еще один способ состоит в том, чтобы при выделенном проекте (в окне **Projects**) в меню **File** выбрать команду **Close Project** (рис. В.17).

Если нужно закрыть все открытые проекты, можем воспользоваться командой **Close All Projects** (см. рис. В.17).

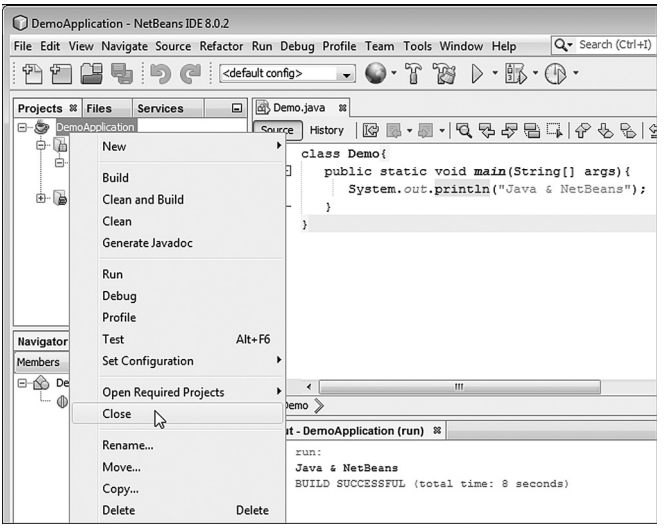

Рис. В.16. Выбор команды **Close** в контекстном меню проекта для его закрытия

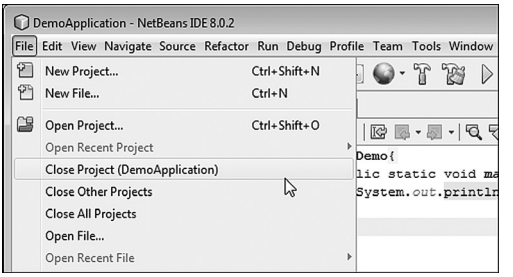

Рис. В.17. Закрытие проекта с помощью команды **Close Project** из меню **File**

Открытие существующего проекта

Иногда необходимо открыть проект, который был создан ранее. В таком случае полезным станет подменю **Open Recent Project** из меню **File**, которое содержит названия недавно открывавшихся проектов (рис. В.18).

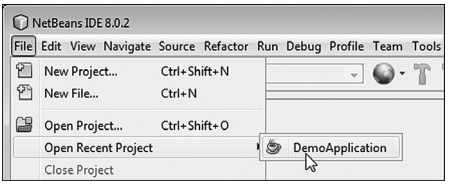

Рис. В.18. Выбор проекта для открытия в списке подменю **Open Recent Project** из меню **File**

Если в меню **File** выбрать команду **Open Project** (рис. В.19), откроется диалоговое окно, в котором выбирается папка, содержащая интересующий нас проект.

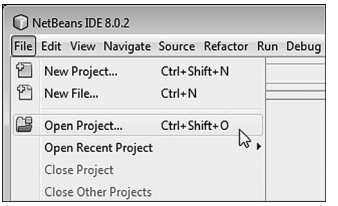

Рис. В.19. Открытие проекта с помощью команды **Open Project** из меню **File**

Наконец, можно воспользоваться специальной кнопкой на панели инструментов, как показано на рис. В.20.

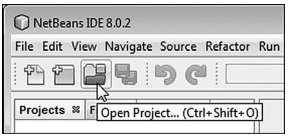

Рис. В.20. Для открытия проекта выполняется щелчок по кнопке на панели инструментов

Еще один способ открыть проект — нажать комбинацию клавиш Ctrl+Shift+O.

О книге

Бывают такие случаи, когда неплохо и соврать.

Из к/ф «Ирония судьбы, или С легким паром!»

Несколько слов хочется сказать собственно о книге. Как отмечалось в самом начале, книга о том, как программировать на Java. Мы начнем с самых простых вещей и постепенно рассмотрим практически все основные темы, которые так или иначе формируют парадигму программирования на Java. Далее приводится список некоторых тем, которые рассматриваются в книге:

- базовые приемы создания программ на Java;
- классы и объекты;

- перегрузка методов;
- лямбда-выражения;
- наследование и переопределение методов;
- использование интерфейсов;
- обработка исключительных ситуаций;
- многопоточное программирование;
- обобщенные классы;
- создание приложений с графическим интерфейсом;
- создание апплетов.

Список не является исчерпывающим, так что, помимо перечисленного выше, книга содержит обсуждение многих иных вопросов и механизмов.

Для удобства усвоения материала книга разбита на относительно небольшие главы. Каждая глава содержит примеры решения различных задач. В конце каждой главы есть краткое резюме, облегчающее усвоение материала главы. Хочется верить, что книга станет надежным помощником для всех, кто изучает язык программирования, в том числе и для тех, кто осваивает премудрости Java самостоятельно.

Обратная связь с автором

Если он явится еще раз, то подожжет дом.
Из к/ф «Ирония судьбы, или С легким паром!»

Автор книги — *Алексей Николаевич Васильев*, профессор кафедры теоретической физики Киевского национального университета имени Тараса Шевченко. Автор книг по программированию и математическому моделированию. Более подробную информацию можно найти на сайте www.vasilev.kiev.ua. Вопросы и замечания можно направлять по электронной почте alex@vasilev.kiev.ua.

Глава 1
ПРИСТУПАЕМ К ПРОГРАММИРОВАНИЮ

— Куда вы меня несете?
— Навстречу твоему счастью.
Из к/ф «Ирония судьбы, или С легким паром!»

Мы приступаем к изучению языка программирования Java. Первое, что мы сделаем, — напишем небольшую программу. И хотя мы еще практически ничего не узнали о языке Java, написать первую программу нам это не помешает.

Первая программа

— Ну ладно, куда ехать?
— Понятия не имею.
Из к/ф «Ирония судьбы, или С легким паром!»

Написание программы начинается с определения задачи, которая должна быть решена, или цели, которая должна быть достигнута при ее выполнении. Цель у нас простая: при выполнении программы должно появляться диалоговое окно с текстовым сообщением. Такая задача в Java решается исключительно просто. Понадобится всего несколько строчек кода.

Создание программы

Поскольку речь идет о первом проекте, мы поступим так: сначала создадим программу, посмотрим, как она выполняется, а уже затем проанализируем программный код.

Мы используем программный код, представленный в листинге 1.1.

Листинг 1.1. Программный код проекта ShowMeAWindowApplication

```
import javax.swing.JOptionPane;
class ShowMeAWindowDemo{
  public static void main(String[] args){
    JOptionPane.showMessageDialog(null,"Первая программа на Java!");
  }
}
```

Код совсем небольшой. Необходимо создать новый проект и ввести в окно редактора программный код из листинга 1.1. Как при этом может выглядеть окно среды разработки NetBeans с кодом программы, показано на рис. 1.1.

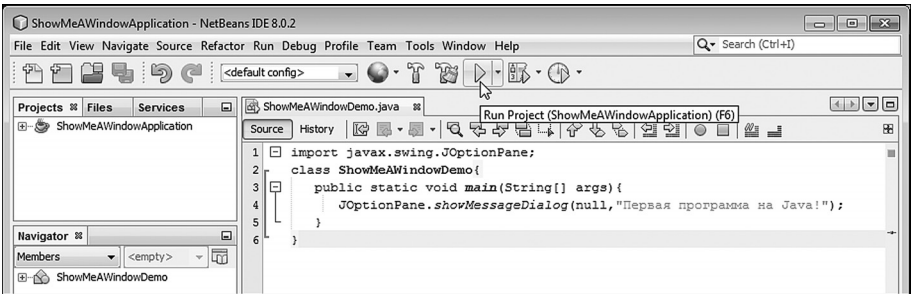

Рис. 1.1. Окно среды разработки NetBeans с кодом программы

ДЕТАЛИ

Напомним алгоритм создания нового приложения в среде разработки NetBeans. Для начала в меню **File** выбираем команду **New Project**. В окне **New Project** в разделе **Categories** выбираем пункт **Java**, а в разделе **Projects** выбираем пункт **Java Application**. В окне **New Java Application** в поле **Project Name** указываем название ShowMeAWindowApplication для имени проекта, в поле **Project Location** задаем место сохранения проекта, а в поле **Create Main Class** указываем название ShowMeAWindowDemo для главного класса (он же единственный), который создается в программе.

Для компиляции и запуска программы на выполнение щелкаем на панели инструментов по кнопке с зеленой стрелкой (см. рис. 1.1) или выбираем в меню **Run** команду **Run Project**. В результате начинает

выполняться программа и на экране появляется диалоговое окно, как на рис. 1.2.

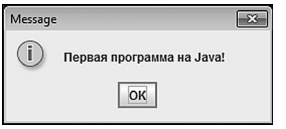

Рис. 1.2. В результате выполнения программы отображается диалоговое окно с сообщением

Окно называется **Message** (название окна отображается слева вверху в строке названия), содержит информационную пиктограмму (круг с буквой i внутри), текст Первая программа на Java!, под которым расположена кнопка **ОК**. При щелчке по кнопке **ОК** или системной пиктограмме (с крестиком) в правом верхнем углу окна оно закрывается, а программа прекращает выполнение.

Анализ программного кода

Теперь проанализируем код программы. Самая первая инструкция import javax.swing.JOptionPane необходима для использования в программе класса JOptionPane из библиотеки Swing.

НА ЗАМЕТКУ

Библиотека Swing содержит набор классов для разработки приложений с графическим интерфейсом. Является неотъемлемой частью платформы Java. С библиотекой Swing мы познакомимся поближе немного позже, когда будем рассматривать создание приложений с графическим интерфейсом.

Дело в том, что диалоговое окно в нашей программе отображается с помощью метода showMessageDialog(). Это статический метод класса JOptionPane — проще говоря, метод описан в данном классе, и говорить о методе showMessageDialog() вне контекста класса JOptionPane просто нет смысла.

ДЕТАЛИ

При вызове метода выполняется определенный блок программного кода или набор команд. В данном случае неважно даже, какие конкретно команды выполняются при вызове метода showMessageDialog(),