Содержание

ГЛАВА 2

Среда программирования на Java

В этой главе...

- Установка комплекта Java Development Kit
- Выбор среды для разработки программ
- Использование инструментов командной строки
- Применение интегрированной среды разработки
- Описание утилиты JShell

Из этой главы вы узнаете, как устанавливать комплект инструментальных средств разработки Java Development Kit (JDK), а также компилировать и запускать на выполнение разнотипные программы: консольные программы, графические и веб-приложения. Мы будем пользоваться инструментальными средствами JDK, набирая команды в окне командной оболочки. Но многие программисты предпочитают удобства, предоставляемые интегрированной средой разработки (IDE). В этой главе будет показано, как пользоваться бесплатно доступной IDE для компиляции и выполнения программ, написанных на Java. Освоить IDE и пользоваться ими нетрудно, но они долго загружаются и предъявляют большие требования к вычислительным ресурсам компьютера, так что применять их для разработки небольших программ не имеет смысла. Овладев приемами, рассмотренными в этой главе, и выбрав подходящие инструментальные средства для разработки программ, вы можете перейти к главе 3, с которой, собственно, начинается изучение языка Java.

2.1. Установка комплекта Java Development Kit

Наиболее полные и современные версии комплекта Java Development Kit (JDK) от компании Oracle доступны для операционных систем Solaris, Linux, Mac OS X

и Windows. Версии, находящиеся на разных стадиях разработки для многих других платформ, лицензированы и поставляются производителями соответствующих платформ.

2.1.1. Загрузка комплекта JDK

Для загрузки комплекта Java Development Kit на свой компьютер вам нужно перейти на веб-страницу по адресу http://www.oracle.com/technetwork/java/javase/downloads/index.html, приложив немного усилий, чтобы разобраться в обозначениях и сокращениях и найти нужное программное обеспечение. И в этом вам помогут сведения, приведенные в табл. 2.1.

Таблица 2.1. Обозначения и сокращения программных средств Java

Наименование	Сокращение	Пояснение
Java Development Kit	JDK	Программное обеспечение для тех, кто желает писать программы на Java
Java Runtime Environment	JRE	Программное обеспечение для потребителей, желающих выполнять программы на Java
Standard Edition	SE	Платформа Java для применения в настольных системах и простых серверных приложениях
Enterprise Edition	EE	Платформа Java для сложных серверных приложений
Micro Edition	ME	Платформа Java для применения в мобильных устройствах
JavaFX	—	Альтернативный комплект инструментальных средств для построения GUI, входивший в дистрибутив Java SE от компании Oracle до версии Java 11
OpenJDK	—	Бесплатная реализация Java SE с открытым кодом, входящая в дистрибутив Java SE от компании Oracle
Java 2	J2	Устаревшее обозначение версий Java, выпущенных в 1998–2006 гг.
Software Development Kit	SDK	Устаревшее обозначение версий JDK, выпущенных в 1998–2006 гг.
Update	u	Обозначение, принятое в компании Oracle для выпусков с исправлениями ошибок вплоть до версии Java 8
NetBeans	—	Интегрированная среда разработки от компании Oracle

Сокращение JDK вам уже знакомо. Оно, как нетрудно догадаться, означает Java Development Kit, т.е. комплект инструментальных средств для разработки программ на Java. Некоторые трудности может вызвать тот факт, что в версиях 1.2–1.4 этот пакет называется Java SDK (Software Development Kit). Иногда вы встретите ссылки на старый термин. Вплоть до главы 10 упоминается также среда Java Runtime Environment (JRE), включающая в себя виртуальную машину, хотя и без компилятора. Но для разработчиков она не подходит, а предназначена для конечных пользователей программ на Java, которым компилятор ни к чему. Далее вам будет встречаться обозначение Java SE. Оно означает Java Standard Edition, т.е. стандартную редакцию Java, в отличие от редакций Java EE (Enterprise Edition) для предприятий и Java ME (Micro Edition) для встроенных устройств.

Иногда вам может встретиться обозначение Java 2, которое было введено в 1998 г., когда специалисты компании Sun Microsystems по маркетингу поняли, что очередной дробный номер выпуска никак не отражает глобальных отличий между JDK 1.2

и предыдущими версиями. Но поскольку такое мнение сформировалось уже после выхода в свет JDK, то было решено, что номер версии 1.2 останется за *комплектом разработки*. Последующие выпуски JDK имеют номера 1.3, 1.4 и 5.0. *Платформа* же была переименована с Java на Java 2. Таким образом, данный комплект разработки называется Java 2 Standard Edition Software Development Kit Version 5.0, или J2SE SDK 5.0.

Правда, в 2006 году нумерация версий была упрощена. Следующая версия Java Standard Edition получила название Java SE 6, а последовавшие за ней версии — Java SE 7 и Java SE 8. Но в то же время они получили "внутренние" номера 1.6.0, 1.7.0 и 1.8.0 соответственно. Эта незначительная путаница в обозначениях дошла и до версии Java SE 9, когда был сначала присвоен номер версии 9, а затем 9.0.1. (А почему не 9.0.0 для первоначальной версии? Еще любопытнее то обстоятельство, что в обозначении номера версии требуется опускать конечные нули, чтобы оставить недолговечный промежуток между главной версией и первым ее обновлением для системы безопасности.)

НА ЗАМЕТКУ! В остальной части этой книги сокращение SE опущено. Поэтому Java 9, по существу, означает Java SE 9.

До версии Java 9 существовали 32- и 64-разрядные версии комплекта Java Development Kit. Но теперь 32-разрядные версии в компании Oracle больше не выпускаются. Чтобы пользоваться современной версией комплекта JDK от компании Oracle, придется установить 64-разрядную версию соответствующей операционной системы.

В Linux необходимо сделать выбор между файлом RPM и архивным файлом с расширением **`.tar.gz`**. Предпочтение лучше отдать последнему, чтобы распаковать архив в любом удобном месте. Итак, выбирая подходящий комплект JDK, необходимо принять во внимание следующее.

- Для дальнейшей работы потребуется комплект JDK (Java SE Development Kit), а не JRE.
- В Linux лучше выбрать архивный файл с расширением **`.tar.gz`**.

Сделав выбор, примите условия лицензионного соглашения и загрузите файл с выбранным комплектом JDK.

НА ЗАМЕТКУ! Компания Oracle предлагает комплект, в который входит комплект инструментальных средств разработки Java Development Kit и интегрированная среда разработки NetBeans. Рекомендуется пока что воздержаться от всех подобных комплектов, установив только Java Development Kit. Если в дальнейшем вы решите воспользоваться NetBeans, загрузите эту IDE по адресу https://netbeans.org.

2.1.2. Установка комплекта JDK

После загрузки JDK нужно установить этот комплект и выяснить, где он был установлен, поскольку эти сведения понадобятся в дальнейшем. Ниже вкратце поясняется порядок установки JDK на разных платформах.

- Если вы работаете в Windows, запустите на выполнение программу установки. Вам будет предложено место для установки JDK. Рекомендуется не принимать предлагаемый каталог. По умолчанию это каталог `c:\Program Files\Java\jdk-11.0.x`. Удалите `Program Files` из пути для установки данного пакета.

- Если вы работаете в Mac OS, запустите на выполнение стандартный установщик. Он автоматически установит программное обеспечение в каталоге `/Library/Java/JavaVirtualMachines/jdk-11.0.x.jdk/Contents/Home`. Найдите этот каталог с помощью утилиты Finder.
- Если вы работаете в Linux, распакуйте архивный файл с расширением **`.tar.gz`**, например, в своем рабочем каталоге или же в каталоге `/opt`. А если вы устанавливаете комплект JDK из файла RPM, убедитесь в том, что он установлен в каталоге `/usr/java/jdk-11.0.x`.

В этой книге делаются ссылки на каталог `jdk`, содержащий комплект JDK. Так, если в тексте указана ссылка `jdk/bin`, она обозначает обращение к каталогу `/opt/jdk-11.0.4/bin` или `c:\Java\jdk-11.0.4\bin`.

После установки JDK вам нужно сделать еще один шаг: добавить имя каталога `jdk/bin` в список путей, по которым операционная система ищет исполняемые файлы. В различных системах это действие выполняется по-разному, как поясняется ниже.

- В Linux добавьте приведенную ниже строку в конце файла `~/.bashrc` или `~/.bash_profile`.

```
export PATH=jdk/bin:$PATH
```

Непременно укажите правильный путь к JDK, например `/opt/jdk-11.0.4`.

В ОС Windows 10 введите **`environment`** (окружение) в поле поиска диалогового окна Windows Settings (Параметры Windows) и установите флажок Edit environment variables for your account (Править переменные окружения для вашей учетной записи; рис. 2.1). В итоге должно появиться диалоговое окно Environment Variables (Переменные окружения), которое может быть скрыто за диалоговым окном Windows Settings. Если же вам не удастся найти его, попробуйте выполнить команду **`sysdm.cpl`** из диалогового окна Run (Выполнить), которое открывается нажатием клавиш <Windows+R>, а затем выберите вкладку Advanced (Дополнительно) и щелкните на кнопке Environment Variables. Найдите и выберите переменную `Path` из списка User Variables (Пользовательские переменные). Щелкните на кнопке Edit (Править), а затем на кнопке New (Создать) и введите элемент с каталогом *`jdk`*`\bin` (рис. 2.2). Сохраните сделанные установки. В итоге подсказка в любом вновь открытом окне командной строки будет начинаться с правильного пути.

Правильность сделанных установок можно проверить следующим образом. Откройте окно терминала, или командной строки, или оболочки. Как вы это сделаете, зависит от операционной системы. Введите следующую строку:

```
java -version
```

Нажмите клавишу <Enter>. На экране должно появиться следующее сообщение:

```
javac 9.0.4
```

Если вместо этого появится сообщение вроде `"java:command not found"` (`java:command` не найдено) или `"The name specified is nor recognized as an internal or external command, operable program or batch file"` (Указанное имя не распознано ни как внутренняя или внешняя команда, ни как действующая программа или командный файл), следует еще раз проверить, правильно ли выполнена установка и задан путь к JDK.

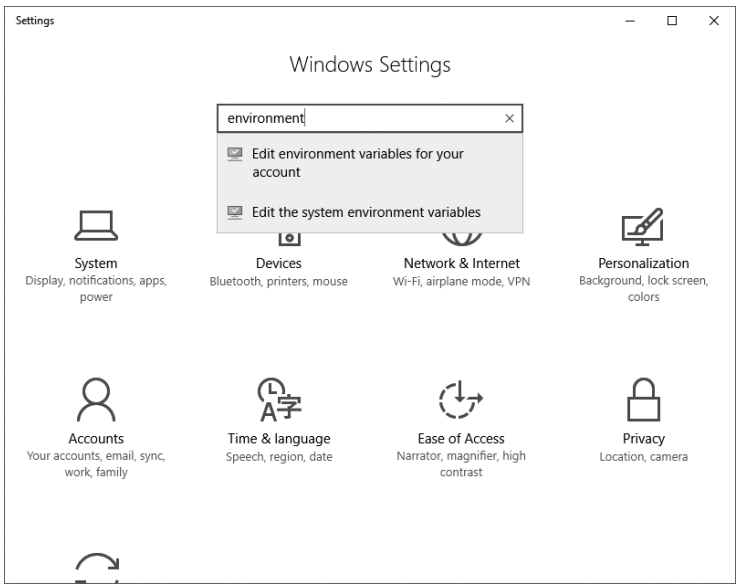

Рис. 2.1. Установка свойств системы в Windows 10

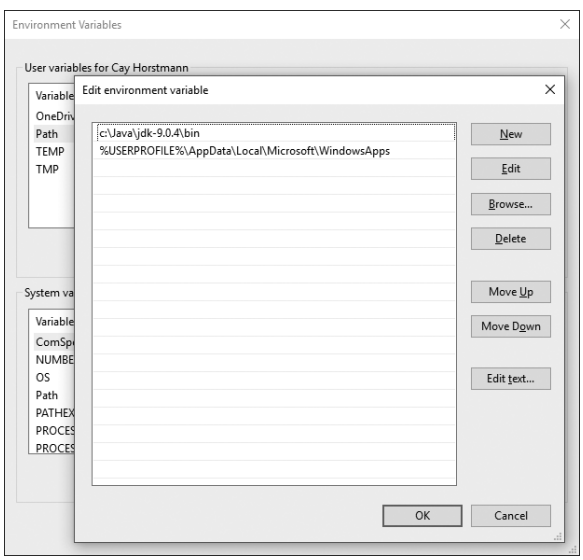

Рис. 2.2. Установка переменной окружения **`Path`** в Windows 10

2.1.3. Установка библиотек и документации

Исходные файлы библиотек поставляются в комплекте JDK в виде архива, хранящегося в файле `src.zip`. Распакуйте этот файл, чтобы получить доступ к исходному коду. С этой целью выполните следующие действия.

1. Убедитесь в том, что комплект JDK установлен, а имя каталога `jdk/bin` находится в списке путей к исполняемым файлам.

2. Создайте каталог `javasrc` в своем начальном каталоге. При желании можете сделать это в окне терминала, командной строки или оболочки, введя следующую команду:

```
mkdir javasrc
```

3. Найдите архивный файл `src.zip` в каталоге *jdk*`/lib`.
4. Распакуйте архивный файл `src.zip` в каталог `javasrc`. При желании можете сделать это в окне терминала, командной строки или оболочки, введя следующие команды:

```
cd javasrc
jar xvf jdk/src.zip
cd ..
```

СОВЕТ. Архивный файл **`src.zip`** содержит исходный код всех общедоступных библиотек. Чтобы получить дополнительный исходный код (компилятора, виртуальной машины, собственных методов и закрытых вспомогательных классов), посетите веб-страницу по адресу **`http://openjdk.java.net`**.

Документация содержится в отдельном от JDK архиве. Ее можно загрузить по адресу `http://www.oracle.com/technetwork/java/javase/downloads/index.html`. Для этого выполните следующие действия.

1. Загрузите архивный файл документации под названием `jdk-11.0.`*x*`_doc-all.zip`.
2. Распакуйте упомянутый выше архивный файл и переименуйте каталог `doc` на нечто более описательное вроде `javadoc`. При желании можете сделать это в окне терминала, командной строки или оболочки, введя следующие команды:

```
jar xvf Downloads/jdk-11.0.x_doc-all.zip
mv docs jdk-11-docs
```

3. Перейдите в окне своего браузера к странице `jdk-11-docs/index.html` и введите эту страницу в список закладок.

Кроме того, установите примеры программ к данной книге. Их можно загрузить по адресу `http://horstmann.com/corejava`. Примеры программ упакованы в архивный файл `corejava.zip`. Распакуйте их в свой начальный каталог. Они расположатся в каталоге `corejava`. При желании можете сделать это в окне терминала, командной строки или оболочки, введя следующую команду:

```
jar xvf Downloads/corejava.zip
```

2.2. Применение инструментов командной строки

Если у вас имеется опыт работы в IDE Microsoft Visual Studio, значит, вы уже знакомы со средой разработки, состоящей из встроенного текстового редактора, меню для компиляции и запуска программ, а также отладчика. В комплект JDK не входят средства, даже отдаленно напоминающие интегрированную среду разработки (IDE). Все команды выполняются из командной строки. И хотя такой подход к разработке программ на Java может показаться обременительным, мастерское владение им является весьма существенным навыком. Если вы устанавливаете платформу Java впервые,

вам придется найти и устранить выявленные неполадки, прежде чем устанавливать IDE. Но выполняя даже самые элементарные действия самостоятельно, вы получаете лучшее представление о внутреннем механизме работы IDE.

После того как вы освоите самые элементарные действия для компиляции и выполнения программ на Java, вам, скорее всего, потребуется IDE. Подробнее об этом речь пойдет в следующем разделе.

Выберем сначала самый трудный путь, вызывая компилятор и запуская программы на выполнение из командной строки.

1. Откройте окно командной оболочки.
2. Перейдите к каталогу `corejava/v1ch02/Welcome`. (Напомним, что каталог `corejava` был специально создан для хранения исходного кода примеров программ из данной книги, как пояснялось в разделе 2.1.3.)
3. Введите следующие команды:

```
javac Welcome.java
java Welcome
```

На экране должен появиться результат, приведенный на рис. 2.3.

Рис. 2.3. Компиляция и выполнение программы **`Welcome.java`**

Примите поздравления! Вы только что в первый раз скомпилировали и выполнили программу на Java.

Что же произошло? Служебная программа (иначе — утилита) **`javac`** — это компилятор Java. Она скомпилировала исходный код из файла `Welcome.java` и преобразовала его в байт-код, сохранив последний в файле `Welcome.class`. А утилита **`java`** запускает виртуальную машину Java. Она выполняет байт-код, который компилятор поместил в указанный файл с расширением **`.class`**.

Программа `Welcome` очень проста и лишь выводит сообщение на экран. Исходный код этой программы приведен в листинге 2.1, а о том, как она работает, речь пойдет в следующей главе.

Листинг 2.1. Исходный код из файла **Welcome.java**

```
1  /**
2   * Эта программа отображает приветствие автора книги
3   * @version 1.30 2014-02-27
4   * @author Cay Horstmann
5  */
6  public class Welcome
7  {
8    public static void main(String[] args)
9    {
10     String greeting = "Welcome to Core Java!";
11     System.out.println(greeting);
12     for (int i = 0; i < greeting.length(); i++)
13       System.out.print("=");
14       System.out.println();
16   }
17 }
```

В эпоху визуальных сред разработки программ многие программисты просто не умеют работать в режиме командной строки. Такое неумение чревато неприятными ошибками. Поэтому, работая в режиме командной строки, необходимо принимать во внимание следующее.

- Если вы вручную набираете код программы, внимательно следите за употреблением прописных и строчных букв. Так, в рассмотренном выше примере имя класса должно быть набрано как `Welcome`, а не `welcome` или `WELCOME`.
- Компилятор требует указывать *имя файла* (в данном случае `Welcome.java`). При запуске программы следует указывать *имя класса* (в данном случае `Welcome`) без расширения **`.java`** или **`.class`**.
- Если вы получите сообщение `"Bad command or file name"` (Неверная команда или имя файла) или упоминавшееся ранее сообщение `"javac:command not found"`, проверьте, правильно ли выполнена установка Java и верно ли указаны пути к исполняемым файлам.
- Если компилятор **`javac`** выдаст сообщение `"cannot read: Welcome.java"` (невозможно прочитать файл `Welcome.java`), следует проверить, имеется ли нужный файл в соответствующем каталоге.
- Если вы работаете в Linux, проверьте, правильно ли набраны прописные буквы в имени файла `Welcome.java`. А в Windows просматривайте содержимое каталогов по команде `dir`, а *не* средствами графического интерфейса Проводника Windows. Некоторые текстовые редакторы (в частности, Notepad) сохраняют текст в файлах с расширением **`.txt`**. Если вы пользуетесь таким редактором для редактирования содержимого файла `Welcome.java`, он сохранит его в файле `Welcome.java.txt`. По умолчанию Проводник Windows скрывает расширение **`.txt`**, поскольку оно предполагается по умолчанию. В этом случае следует переименовать файл, воспользовавшись командой **`ren`**, или повторно сохранить его, указав имя в кавычках, например `"Welcome.java"`.
- Если при запуске программы вы получаете сообщение об ошибке типа `java.lang.NoClassDefFoundError`, проверьте, правильно ли вы указали имя файла.
- Если вы получите сообщение касательно имени `welcome`, начинающегося со строчной буквы **`w`**, еще раз выполните команду `java Welcome`, написав это имя

с прописной буквы **W**. Не забывайте, что в Java учитывается регистр символов. Если же вы получите сообщение по поводу ввода имени `Welcome/java`, значит, вы случайно ввели команду `java Welcome.java`. Повторите команду `java Welcome`.

- Если вы указали имя `Welcome`, а виртуальная машина не в состоянии найти класс с этим именем, проверьте, не установлена ли каким-нибудь образом переменная окружения `CLASSPATH` в вашей системе. Эту переменную, как правило, не стоит устанавливать глобально, но некоторые неудачно написанные установщики программного обеспечения в Windows это делают. Последуйте той же самой процедуре, что и для установки переменной окружения `PATH`, но на этот раз удалите установку переменной окружения `CLASSPATH`.

СОВЕТ. Отличное учебное пособие имеется по адресу **https://docs.oracle.com/javase/tutorial/getStarted/cupojava**. В нем подробно описываются скрытые препятствия, которые нередко встречаются на пути начинающих программировать на Java.

НА ЗАМЕТКУ! В версии JDK 11 вместо команды **javac** с единственным исходным файлом можно пользоваться сценариями оболочки, начинающимися со строки **#!/path/to/java** с шебангом (т.е. со знаками решетки и восклицания в самом начале).

Программа `Welcome` не особенно впечатляет. Поэтому рассмотрим пример графического приложения. Это приложение представляет собой очень простую программу, которая загружает и выводит на экран изображение из файла. Как и прежде, скомпилируем и выполним это приложение в режиме командной строки.

1. Откройте окно терминала или командной оболочки.
2. Перейдите к каталогу `corejava/v1ch02/ImageViewer`.
3. Введите следующие команды:

```
javac ImageViewer.java
java ImageViewer
```

На экране появится новое окно приложения `ImageViewer`. Выберите команду меню File⇨Open (Файл⇨Открыть) и найдите файл изображения, чтобы открыть его. (В одном каталоге с данной программой находится несколько графических файлов.) Изображение появится в окне (рис. 2.4). Чтобы завершить выполнение программы, щелкните на кнопке Close (Закрыть) в строке заголовка текущего окна или выберите команду меню File⇨Exit (Файл⇨Выйти).

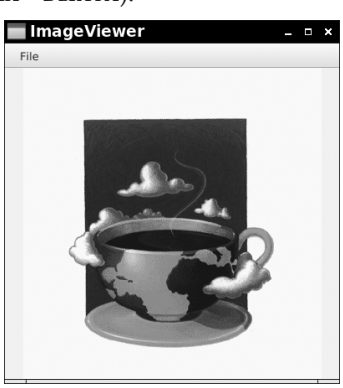

Рис. 2.4. Окно выполняющегося приложения **ImageViewer**

Бегло просмотрите исходный код данной программы, приведенный в листинге 2.2. Эта программа заметно длиннее, чем первая, но и она не слишком сложна, особенно если представить себе, сколько строк кода на языке C или C++ нужно было бы написать, чтобы получить такой же результат. Написанию приложений с графическим пользовательским интерфейсом (GUI), подобных данной программе, посвящена глава 10.

Листинг 2.2. Исходный код из файла **ImageViewer/ImageViewer.java**

```
1  import java.awt.*;
2  import java.io.*;
3  import javax.swing.*;
4
5  /**
6   * Программа для просмотра изображений.
7   * @version 1.31 2018-04-10
8   * @author Cay Horstmann
9  */
10 public class ImageViewer
11 {
12   public static void main(String[] args)
13   {
14     EventQueue.invokeLater(() -> {
15       var frame = new ImageViewerFrame();
16       frame.setTitle("ImageViewer");
17       frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
18       frame.setVisible(true);
19     });
20   }
21 }
22
23 /**
24  * Фрейм с текстовой меткой для вывода изображения.
25  */
26 class ImageViewerFrame extends JFrame
27 {
28   private static final int DEFAULT_WIDTH = 300;
29   private static final int DEFAULT_HEIGHT = 400;
30
31   public ImageViewerFrame()
32   {
33     setSize(DEFAULT_WIDTH, DEFAULT_HEIGHT);
34
35     // использовать метку для вывода изображений на экран
36     var label = new JLabel();
37     add(label);
38
39     // установить селектор файлов
40     var chooser = new JFileChooser();
41     chooser.setCurrentDirectory(new File("."));
42
43     // установить строку меню
44     var menuBar = new JMenuBar();
```

```
45      setJMenuBar(menuBar);
46
47      var menu = new JMenu("File");
48      menuBar.add(menu);
49
50      var openItem = new JMenuItem("Open");
51      menu.add(openItem);
52      openItem.addActionListener(event -> {
53        // отобразить диалоговое окно селектора файлов
54        int result = chooser.showOpenDialog(null);
55
56       // если файл выбран, задать его в качестве
57       // пиктограммы для метки
58       if (result == JFileChooser.APPROVE_OPTION)
59       {
60         String name = chooser.getSelectedFile().getPath();
61         label.setIcon(new ImageIcon(name));
62       }
63      });
64
65      var exitItem = new JMenuItem("Exit");
66      menu.add(exitItem);
67      exitItem.addActionListener(event -> System.exit(0));
68    }
69 }
```

2.3. Применение IDE

В предыдущем разделе было показано, каким образом программа на Java компилируется и выполняется из командной строки. И хотя это очень полезный навык, для повседневной работы следует выбрать интегрированную среду разработки (IDE). Такие среды стали настолько эффективными и удобными, что профессионально разрабатывать программное обеспечение без их помощи просто не имеет смысла. К числу наиболее предпочтительных относятся IDE Eclipse, NetBeans и IntelliJ IDEA. В этой главе будет показано, как приступить к работе с IDE Eclipse. Но вы вольны выбрать другую IDE для проработки материала данной книги.

Прежде всего загрузите IDE Eclipse по адресу `http://www.eclipse.org/downloads/`, где имеются версии IDE Eclipse для Linux, Mac OS X, Solaris и Windows. Выполните программу установки и выберите установочный набор Eclipse IDE for Java Developers (IDE Eclipse для разработчиков программ на Java).

Чтобы приступить к написанию программы на Java в IDE Eclipse, выполните следующие действия.

1. После запуска Eclipse выберите из меню команду File⇨New Project (Файл⇨Создать проект).
2. Выберите вариант Java Project (Проект Java) в диалоговом окне мастера проектов (рис. 2.5).
3. Щелкните на кнопке Next (Далее). Сбросьте флажок Use default location (Использовать место по умолчанию). Щелкните на кнопке Browse (Обзор) и перейдите к каталогу `corejava/v1ch02/Welcome` (рис. 2.6).

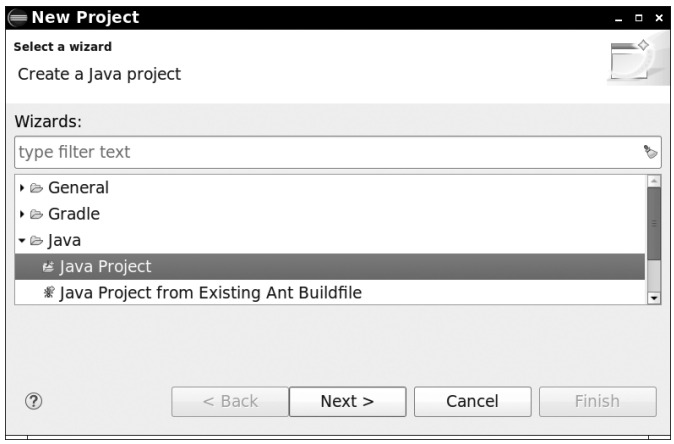

Рис. 2.5. *Диалоговое окно Eclipse для создания нового проекта*

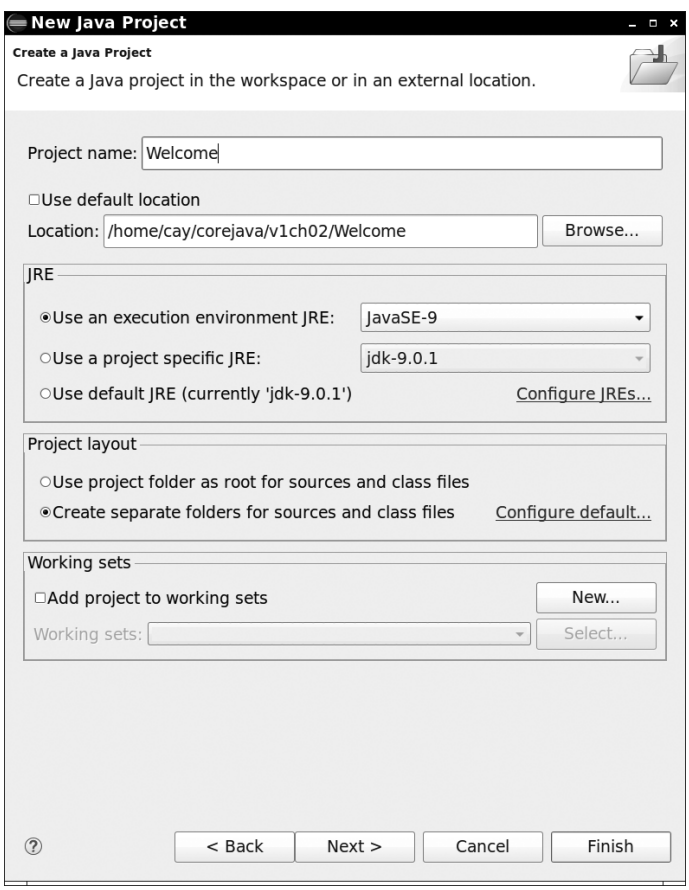

Рис. 2.6. Настройка проекта в Eclipse

4. Щелкните на кнопке Finish (Готово). В итоге будет создан новый проект.
5. Щелкайте по очереди на треугольных кнопках слева от имени проекта до тех пор, пока не найдете файл `Welcome.java`, а затем дважды щелкните на нем. В итоге появится окно с исходным кодом программы, как показано на рис. 2.7.

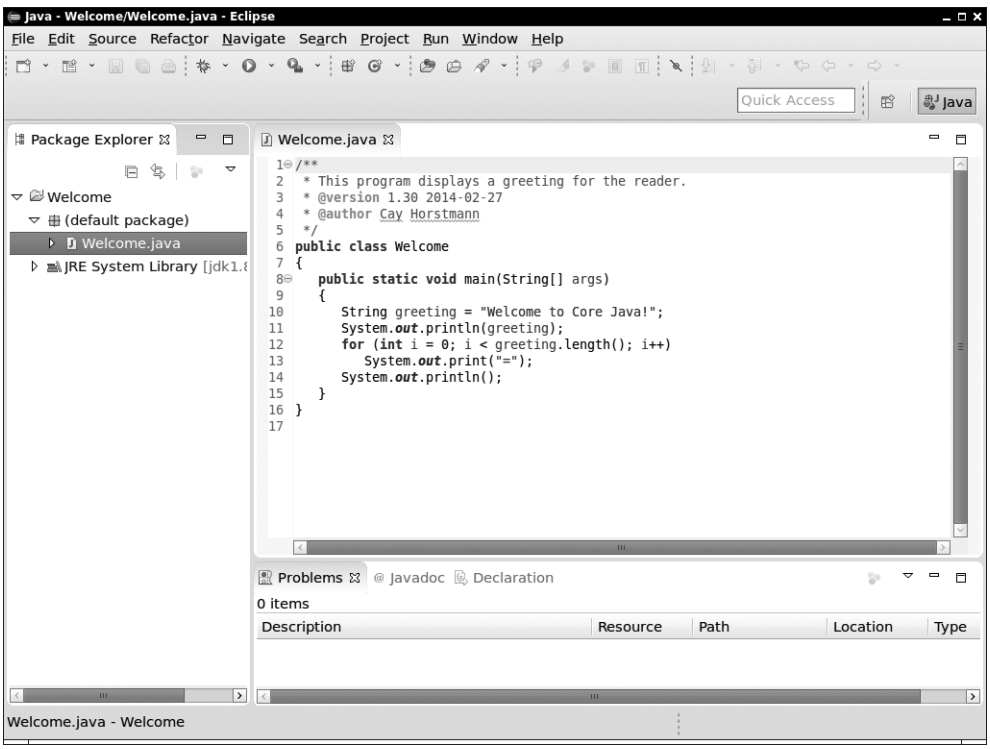

Рис. 2.7. Редактирование исходного кода в Eclipse

6. Щелкните правой кнопкой мыши на имени проекта (`Welcome`) на левой панели. Выберите в открывшемся контекстном меню команду Run⇨Run As⇨Java Application (Выполнить⇨Выполнить как⇨Приложение Java). В нижней части окна появится окно для вывода результатов выполнения программы.

Рассматриваемая здесь программа состоит из нескольких строк кода, и поэтому в ней вряд ли имеются ошибки или опечатки. Но для того, чтобы продемонстрировать порядок обработки ошибок, допустим, что в имени строчного типа данных `String` вместо прописной буквы набрана строчная:

```
string greeting = "Welcome to Core Java!";
```

Обратите внимание на волнистую линию под словом `string`. Перейдите на вкладку Problems (Ошибки) ниже исходного кода и щелкайте на треугольных кнопках до тех пор, пока не увидите сообщение об ошибке в связи с неверно указанным типом данных `string` (рис. 2.8). Щелкните на этом сообщении. Курсор автоматически перейдет на соответствующую строку кода в окне редактирования, где вы можете быстро исправить допущенную ошибку.

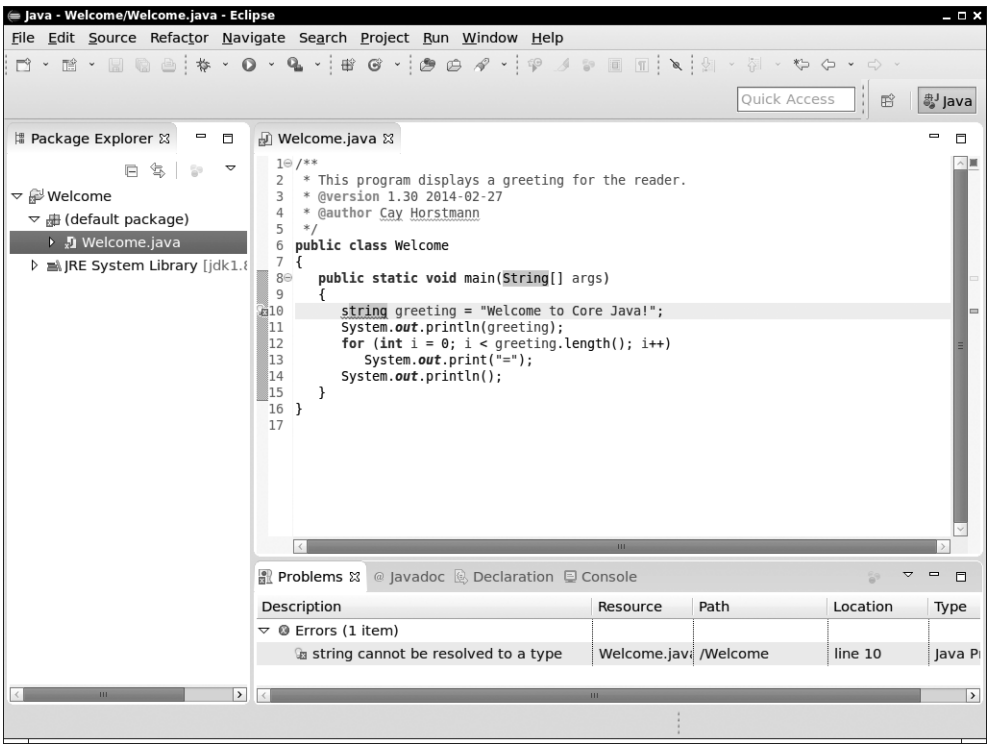

Рис. 2.8. Сообщение об ошибке, выводимое в окне Eclipse

СОВЕТ. Зачастую вывод сообщений об ошибках в Eclipse сопровождается пиктограммой с изображением лампочки. Щелкните на этой пиктограмме, чтобы получить список рекомендуемых способов исправления ошибки.

2.4. Утилита JShell

В предыдущем разделе было показано, как компилировать и выполнять прикладную программу на Java. В версии Java 9 внедрен еще один способ работы на платформе Java. *Для этой цели утилита* JShell предоставляет цикл "чтение–вычисление–вывод" (REPL). Вы вводите выражение на языке Java, а утилита JShell вычисляет его, выводит получаемый результат и ожидает от вас ввода следующего выражения. Чтобы запустить утилиту JShell на выполнение, достаточно ввести команду **`jshell`** в окне терминала, командной строки или оболочки (рис. 2.9).

Утилита JShell начинает свое выполнение с приветствия и последующей подсказки:

```
| Welcome to JShell -- Version 9.0.1
| For an introduction type: /help intro[1]

jshell>
```

[1] | Добро пожаловать в JShell -- версии 9.0.1
| Для ознакомления введите команду: /help intro

Теперь введите следующее выражение:

```
"Core Java".length()
```

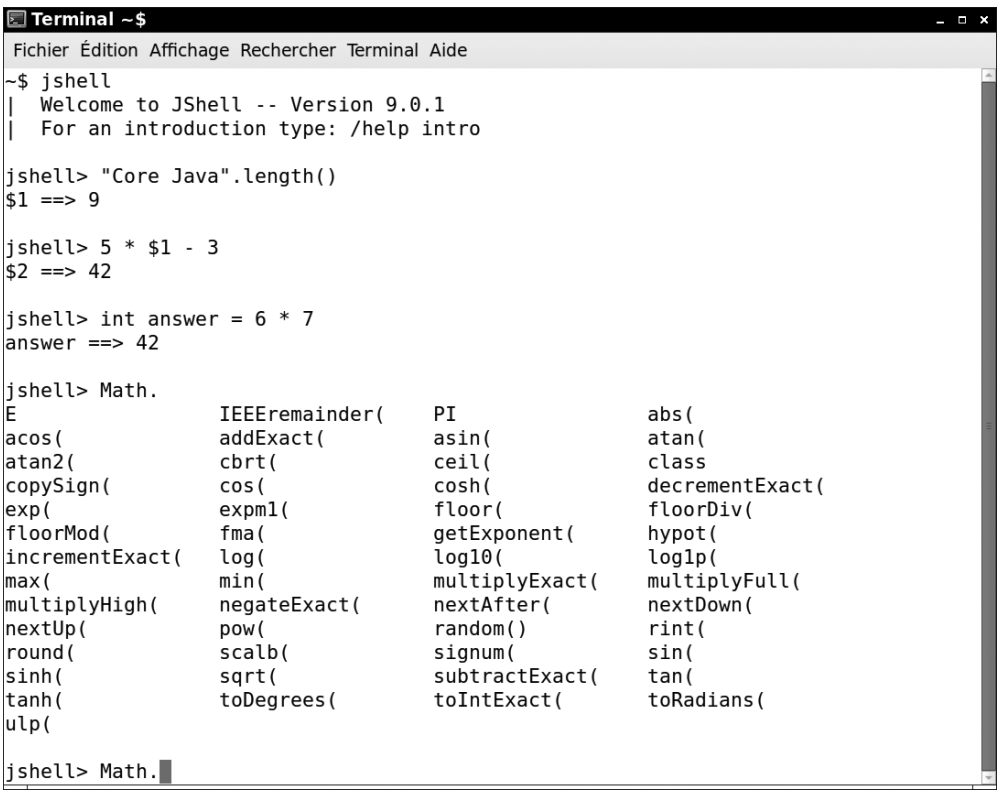

Рис. 2.9. Выполнение утилиты JShell

Утилита JShell ответит на это выводом получаемого результата. В данном случае это количество символов в строке `"Core Java"`, как показано ниже.

```
$1 ==> 9
```

Обратите внимание на то, что для получения такого результата вам *не* нужно вводить оператор `System.out.println`. Утилита JShell автоматически выведет значение каждого вводимого вами выражения.

В выводимом результате `$1` обозначает, что полученный результат доступен для последующих вычислений. Так, если ввести следующее выражение:

```
5 * $1 - 3
```

то в ответ будет получен такой результат:

```
$2 ==> 42
```

Если переменной требуется пользоваться неоднократно, ей можно присвоить более запоминающееся имя. Но для этого придется следовать правилам синтаксиса Java, указав как тип данных, так и имя переменных. (Более подробно синтаксис Java рассматривается в главе 3.) Например:

```
jshell> int answer = 6 * 7
answer ==> 42
```

Еще одним полезным средством является автозаполнение нажатием клавиши табуляции. Чтобы опробовать его, введите

```
Math.
```

и нажмите клавишу табуляции. В итоге будет выведен перечень всех методов, которые могут быть вызваны для переменной `generator`:

```
jshell> Math.
E                IEEEremainder(   PI               abs(
acos(            addExact(        asin(            atan(
atan2(           cbrt(            ceil(            class
copySign(        cos(             cosh(            decrementExact(
exp(             expm1(           floor(           floorDiv(
floorMod(        fma(             getExponent(     hypot(
incrementExact(  log(             log10(           log1p(
max(             min(             multiplyExact(   multiplyFull(
multiplyHigh(    negateExact(     nextAfter(       nextDown(
nextUp(          pow(             random()         rint(
round(           scalb(           signum(          sin(
sinh(            sqrt(            subtractExact(   tan(
tanh(            toDegrees(       toIntExact(      toRadians(
ulp(
```

А теперь введите **`l`** и снова нажмите клавишу табуляции. Имя метода будет автоматически дополнено до **`log`**, а в итоге получен более короткий список.

```
jshell> Math.log
log(     log10(      log1p(
```

Вызов метода можно далее заполнить вручную:

```
jshell> Math.log10(0.001)
$3 ==> -3.0
```

Чтобы повторить команду, нажимайте клавишу <↑> до тех пор, пока не появится строка, которую требуется выполнить снова или отредактировать. С помощью клавиш <←> и <→> можно переместить курсор и ввести или удалить символы. Отредактировав строку, нажмите клавишу <Enter>. Например, введите числовое значение **`0.001`** и замените его значением **`1000`**, а затем нажмите клавишу <Enter>:

```
jshell> Math.log10(1000)
$4 ==> 3.0
```

Утилита JShell упрощает и делает занимательным изучение языка Java и его библиотек, не прибегая к тяжеловесной среде разработки и не возясь с такими длинными операторами, как `public static void main`.

Итак, в этой главе были рассмотрены механизмы компиляции и запуска программ, написанных на Java. Теперь вы готовы перейти к главе 3, чтобы приступить непосредственно к изучению языка Java.