Chapter 1

An introduction to Java

Objectives

Applied

1. Given a NetBeans project that contains the source code for a Java application, use NetBeans to open the project, view and compile the source code, and run the application.
2. Given the source code for a Java application, use NetBeans to create a project; enter edit, and compile the source code; and run the application.

Objectives (cont.)

Knowledge

1. Describe how Java compares with C++ and C# based on these features: syntax, platform independence, speed, and memory management.
2. Name and describe two types of desktop applications that you can create with Java.
3. Describe how Java compiles and interprets code.
4. Explain how the use of bytecode lets Java achieve platform independence.
5. Describe the benefits of using a Java IDE like NetBeans or Eclipse.
6. Explain why you don’t need to compile the source code for an application before you use NetBeans to run the application.

Java timeline

Year	Month	Release
1996	January	JDK 1.0
1997	February	JDK 1.1
1998	December	SDK 1.2
1999	August	Java 2 Platform, Standard Edition (J2SE)
	December	Java 2 Platform, Enterprise Edition (J2EE)
2000	May	J2SE with SDK 1.3
2002	February	J2SE with SDK 1.4
2004	September	J2SE 5.0 with JDK 1.5
2006	December	Java SE 6 with JDK 1.6
2011	July	Java SE 7 with JDK 1.7
2014	March	Java SE 8 with JDK 1.8
2017	July	Java SE 9 with JDK 1.9

Java editions

- Java SE (Standard Edition)
- Java EE (Enterprise Edition)
- Java ME (Micro Edition)

Operating systems that support Java

- Windows
- Mac OS X
- Linux
- Most versions of UNIX
- Most other modern operating systems

A note about Android

- The Android operating system doesn’t support Java in the same way as most operating systems. However, you can use all Java 7 language features and some Java 8 features to write the code for Android apps.

Java compared to C++

Feature	Description
Syntax	Java syntax is similar to C++ syntax.
Platforms	Compiled Java code can run on any platform that has a Java runtime environment. C++ code must be compiled once for each type of system that it is going to be run on.
Speed	C++ runs faster than Java in some contexts, but Java runs faster in other contexts.
Memory	Java handles most memory operations automatically, but C++ programmers must write code that manages memory.

Java compared to C#

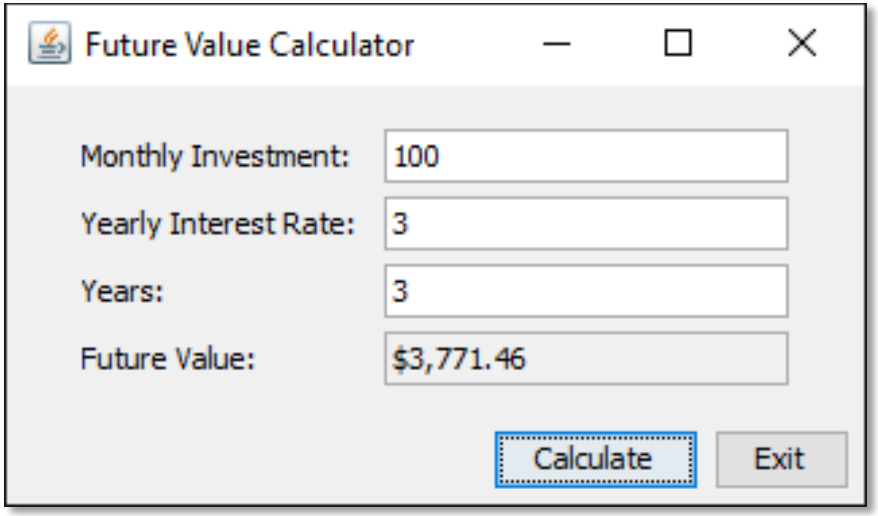
Feature	Description
Syntax	Java syntax is similar to C# syntax.
Platforms	Like Java, compiled C# code can run on any platform that has a runtime environment for it.
Speed	Java runs faster than C# in most contexts.
Memory	Like Java, C# handles most memory operations automatically.

A console application

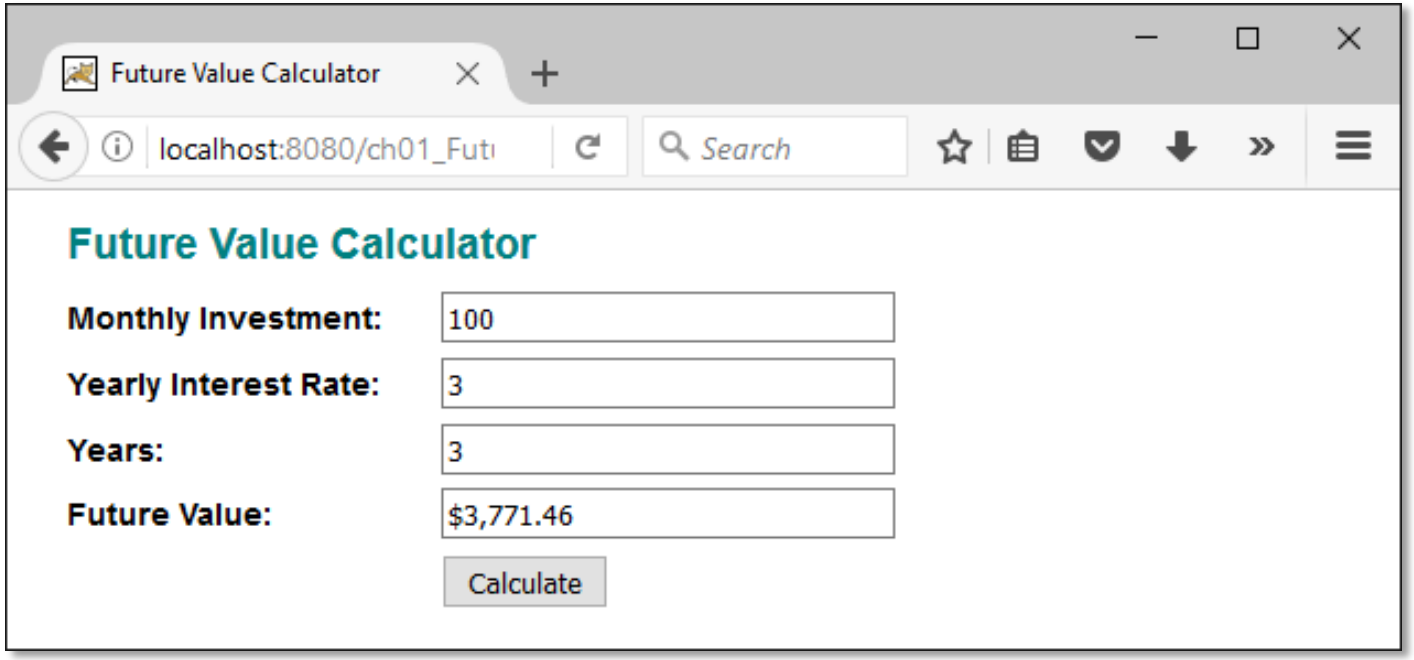
Command Prompt - java -jar FutureValueConsole.jar

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\Joel>cd \murach\java\files

C:\murach\java\files>java -jar FutureValueConsole.jar
Welcome to the Future Value Calculator

Enter monthly investment:   100
Enter yearly interest rate: 3
Enter number of years:      3
Future value:               $3,771.46

Continue? (y/n):
```

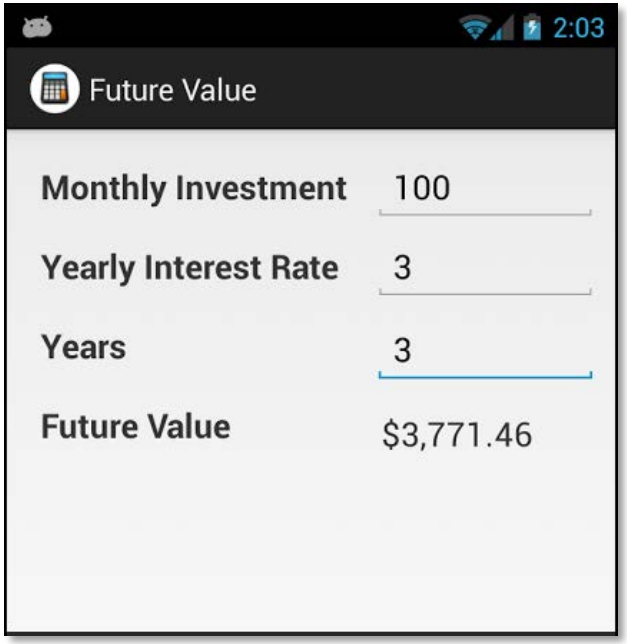
A GUI application



A web application



A mobile app



The code for a console application

```
import java.text.NumberFormat;
import java.util.Scanner;

public class FutureValueApp {

    public static void main(String[] args) {
        System.out.println(
            "Welcome to the Future Value Calculator");
        System.out.println();

        Scanner sc = new Scanner(System.in);

        String choice = "y";
        while (choice.equalsIgnoreCase("y")) {

            // get input from user
            System.out.print(
                "Enter monthly investment:   ");
            double monthlyInvestment = sc.nextDouble();
```

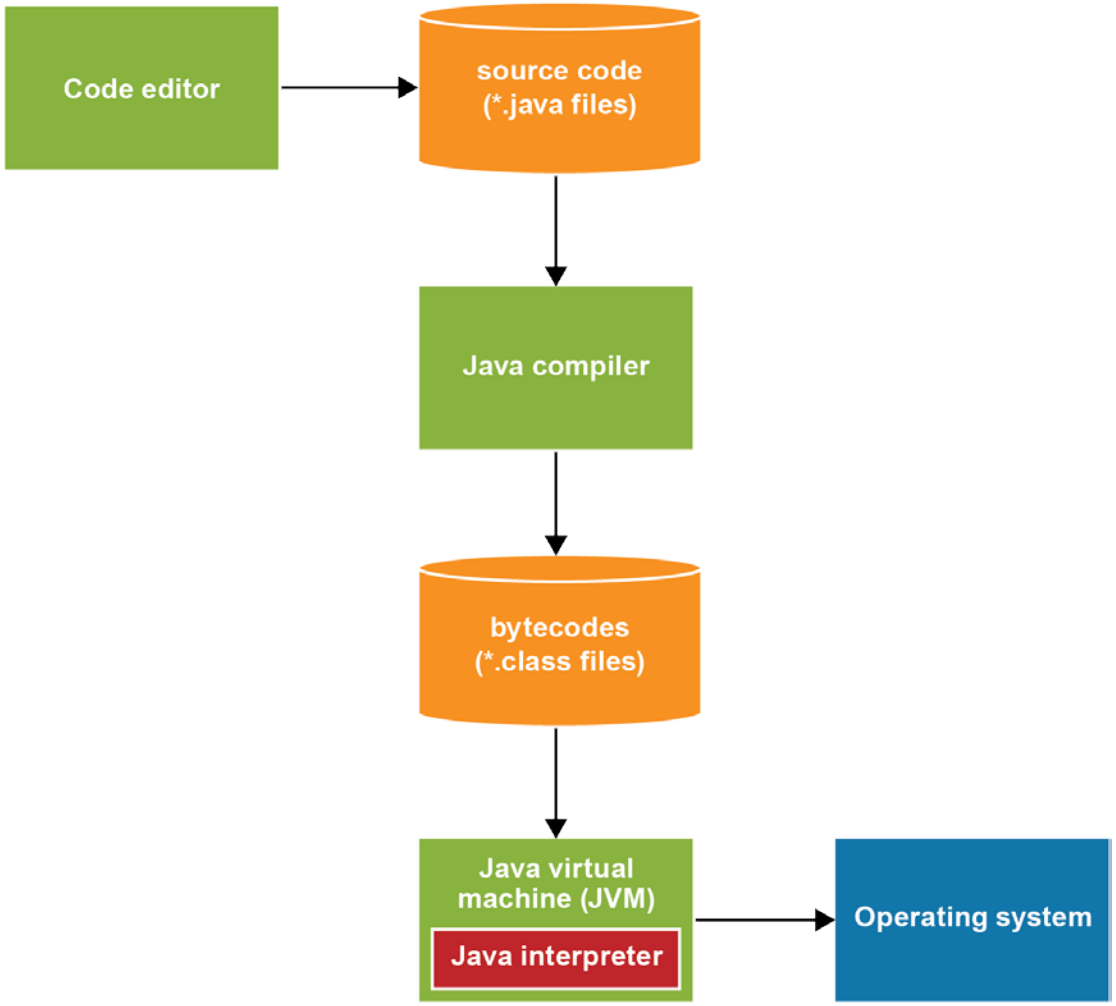
The code for a console application (cont.)

```
System.out.print(
    "Enter yearly interest rate: ");
double interestRate = sc.nextDouble();
System.out.print(
    "Enter number of years:      ");
int years = sc.nextInt();

// calculate the future value
double monthlyInterestRate =
    interestRate / 12 / 100;
int months = years * 12;
double futureValue = calculateFutureValue(
        monthlyInvestment, monthlyInterestRate,
        months);

// format and display the result
NumberFormat currency =
    NumberFormat.getCurrencyInstance();
System.out.println(
    "Future value:               "
        + currency.format(futureValue) + "\n");
```

The code for a console application (cont.)

```
            // see if the user wants to continue
            System.out.print("Continue? (y/n): ");
            choice = sc.next();
            System.out.println();
        }
    }

    private static double calculateFutureValue(
            double monthlyInvestment,
            double monthlyInterestRate, int months) {
        double futureValue = 0;
        for (int i = 1; i <= months; i++) {
            futureValue = (futureValue + monthlyInvestment)
                    * (1 + monthlyInterestRate);
        }
        return futureValue;
    }
}
```

How Java compiles and interprets code



Popular Java IDEs

- NetBeans
- Eclipse
- IntelliJ IDEA
- Android Studio

Features provided by most IDEs

- A code editor with code completion and error detection.
- Automatic compilation of classes when you run the application.
- A debugger that lets you set breakpoints, step through code, and view the values of active variables.

NetBeans with a Java project

ch01_FutureValueConsole - NetBeans IDE 8.1

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects

- ch01_FutureValueConsole
  - Source Packages
    - <default package>
      - FutureValueApp.java
  - Libraries
    - JDK 1.8 (Default)

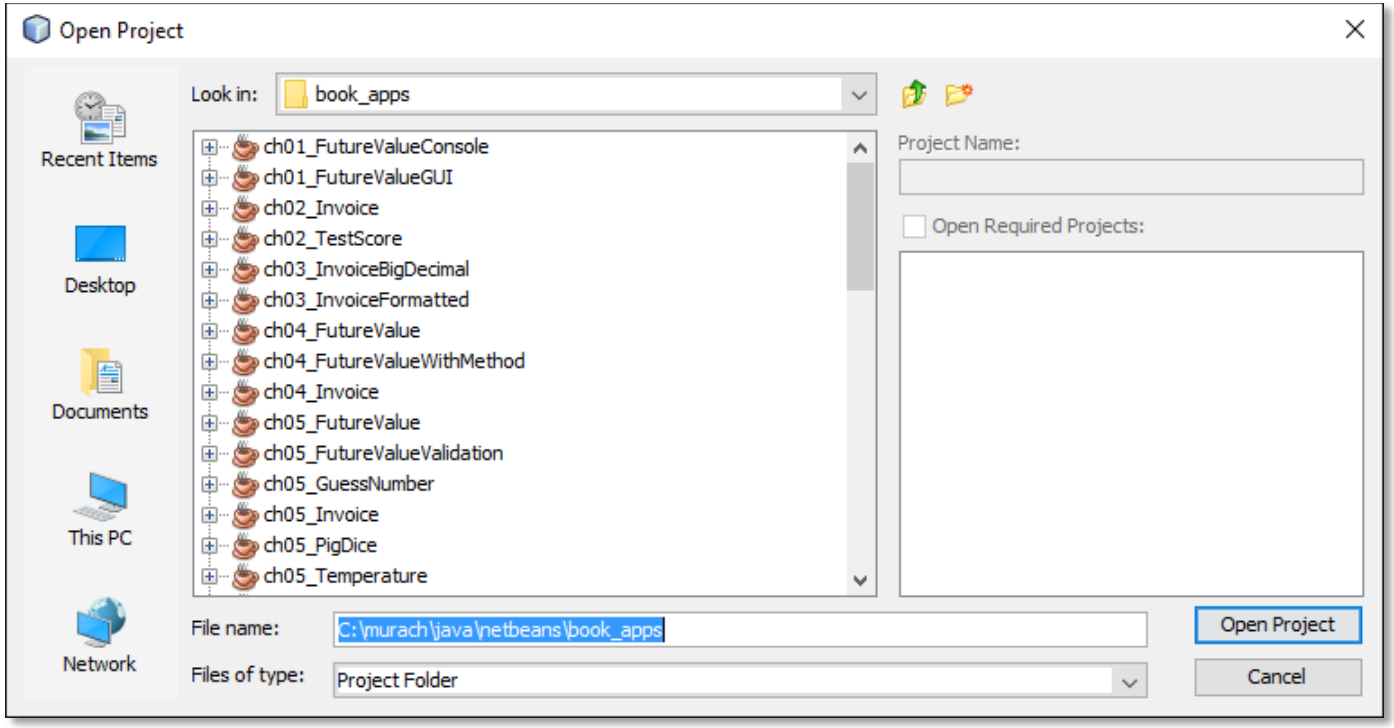
main - Navigator

- FutureValueApp
  - calculateFutureValue(double mo
  - main(String[] args)

FutureValueApp.java

```java
import java.text.NumberFormat;
import java.util.Scanner;

public class FutureValueApp {

    public static void main(String[] args) {
        System.out.println("Welcome to the Future Value Calculator");
        System.out.println();

        // get a Scanner object to scan for user input
        Scanner sc = new Scanner(System.in);

        String choice = "y";
        while (choice.equalsIgnoreCase("y")) {

            // get input from user
            System.out.print("Enter monthly investment:   ");
            double monthlyInvestment = sc.nextDouble();

            System.out.print("Enter yearly interest rate: ");
            double interestRate = sc.nextDouble();

            System.out.print("Enter number of years:      ");
```

The dialog box for opening a project



How to open, close, and delete a project

- To open a project, click the Open Project button in the toolbar or select the File→Open Project command. Then, use the Open Project dialog box that's displayed to locate and select the project and click the Open Project button.
- You can also open a project by using the File→Open Recent Project command and then selecting the project from the list that's displayed.
- To close a project, right-click on the project in the Projects window and select the Close command, or select the project and then use the File→Close Project command.
- To delete a project, right-click on the project in the Projects window and select the Delete command. When you do, you'll have the option of deleting just the files that NetBeans uses to manage the project or deleting all the folders and files for the project.

How to compile and run a project

- To run a project, press F6 or click the Run Project button in the toolbar.
- When you run a project, NetBeans automatically compiles it. As a result, you usually don't need to compile a project separately.
- To compile a project without running it, you can right-click on the project in the Projects window and select the Build command.
- To delete all compiled files for a project and compile them again, you can right-click on the project and select the Clean and Build command. This removes files that are no longer needed and compiles the entire project.

Mac OS X note

- To enable right-clicking with Mac OS X, you can edit the system preferences for the mouse.

The Output window for input and output

ch01_FutureValueConsole - NetBeans IDE 8.1

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects

- ch01_FutureValueConsole
  - Source Packages
    - <default package>
      - FutureValueApp.java
  - Libraries
    - JDK 1.8 (Default)

FutureValueApp.java

```
1  import java.text.NumberFormat;
2  import java.util.Scanner;
3
4  public class FutureValueApp {
5
6      public static void main(String[] args) {
7          System.out.println("Welcome to the Future Value Calculator");
8          System.out.println();
9
10         // get a Scanner object to scan for user input
11         Scanner sc = new Scanner(System.in);
```

main - Navigator

- FutureValueApp
  - calculateFutureValue(double mo
  - main(String[] args)

Output

```
run:
Welcome to the Future Value Calculator

Enter monthly investment:   100
Enter yearly interest rate: 3
Enter number of years:      3
Future value:               $3,771.46

Continue? (y/n):
```

NetBeans with two open projects

ch02_Invoice - NetBeans IDE 8.1

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects
- ch01_FutureValueConsole
  - Source Packages
    - <default package>
      - FutureValueApp.java
  - Libraries
    - JDK 1.8 (Default)
- ch02_Invoice
  - Source Packages
    - <default package>
      - InvoiceApp.java
  - Libraries

Navigator
- Members
- InvoiceApp
  - main(String[] args)

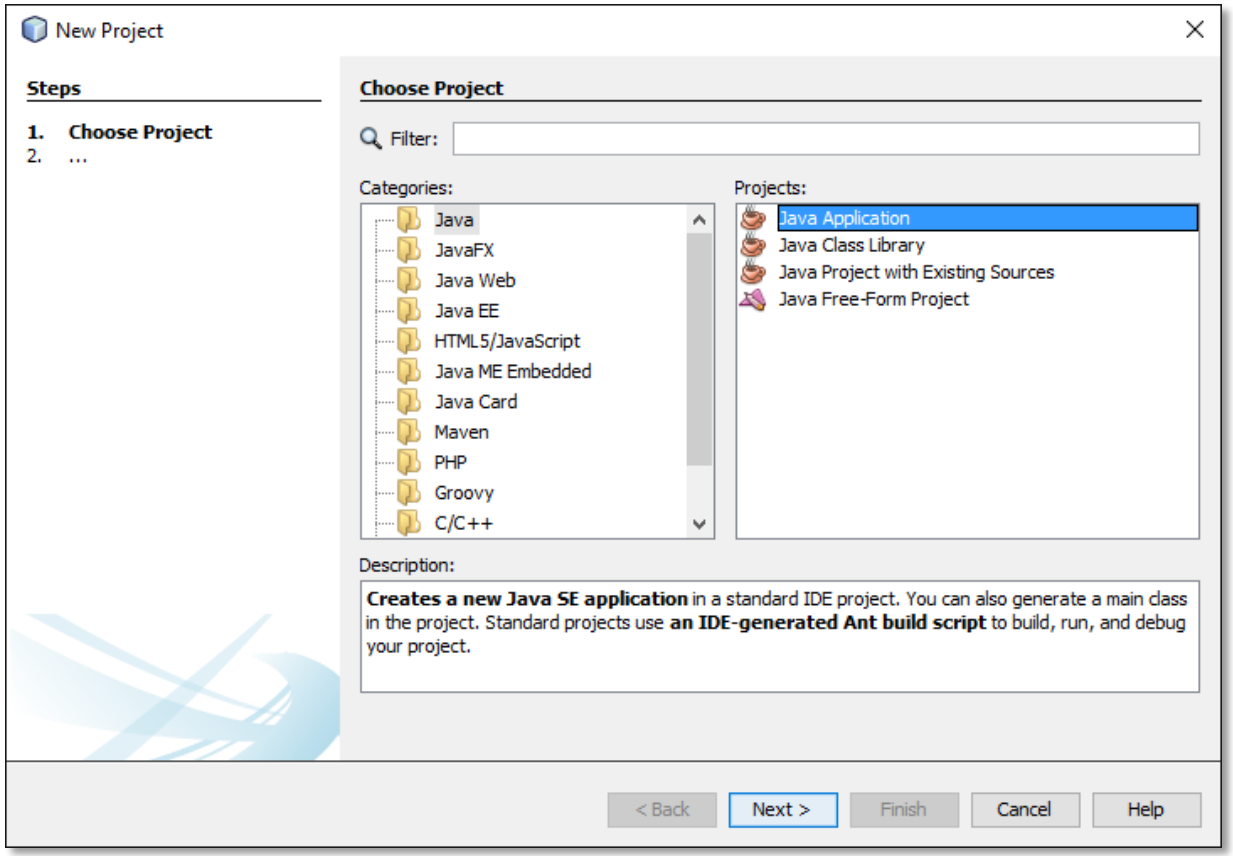
FutureValueApp.java | InvoiceApp.java

Source | History

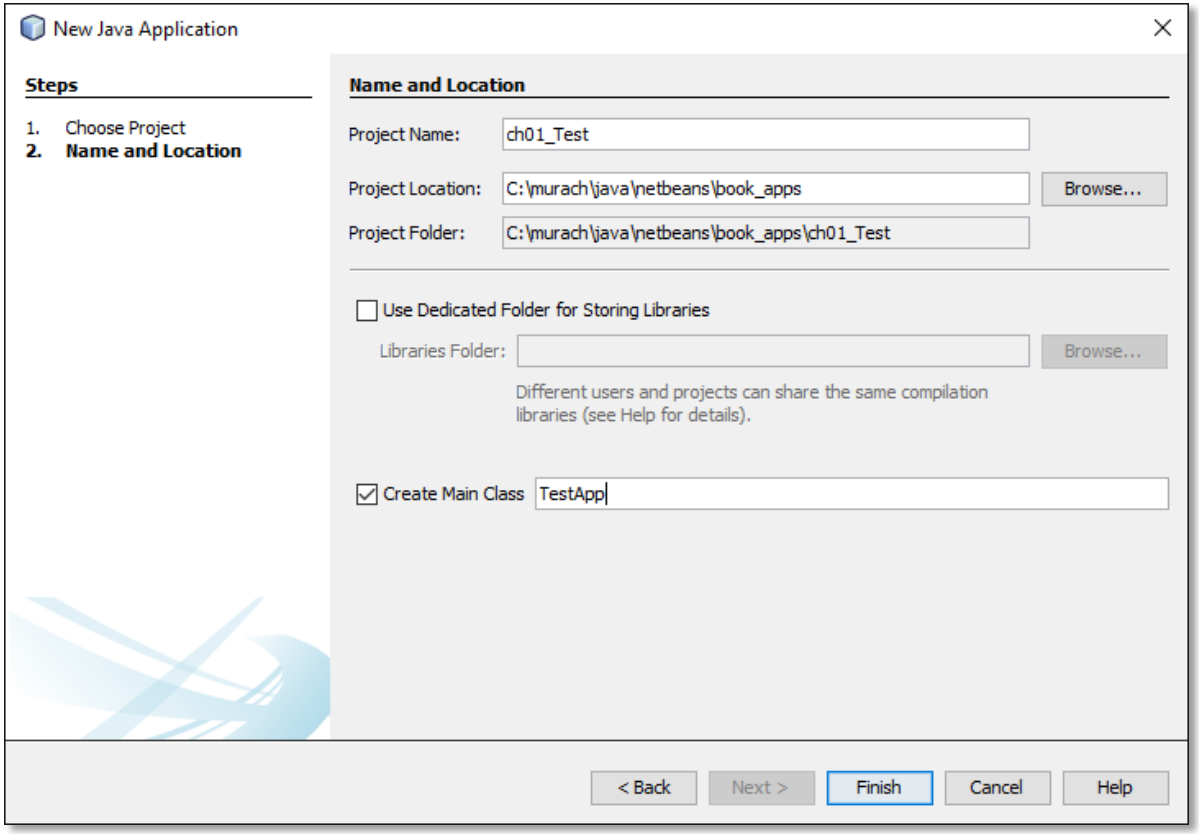
```
import java.util.Scanner;

public class InvoiceApp {

    public static void main(String[] args) {
        // welcome the user to the program
        System.out.println("Welcome to the Invoice Total Calculator");
        System.out.println();  // print a blank line

        // create a Scanner object named sc
        Scanner sc = new Scanner(System.in);

        // perform invoice calculations until choice isn't equal to "y" c
        String choice = "y";
        while (choice.equalsIgnoreCase("y")) {
            // get the invoice subtotal from the user
            System.out.print("Enter subtotal:   ");
            double subtotal = sc.nextDouble();

            // calculate the discount amount and total
            double discountPercent;
            if (subtotal >= 200) {
```

1:1 INS

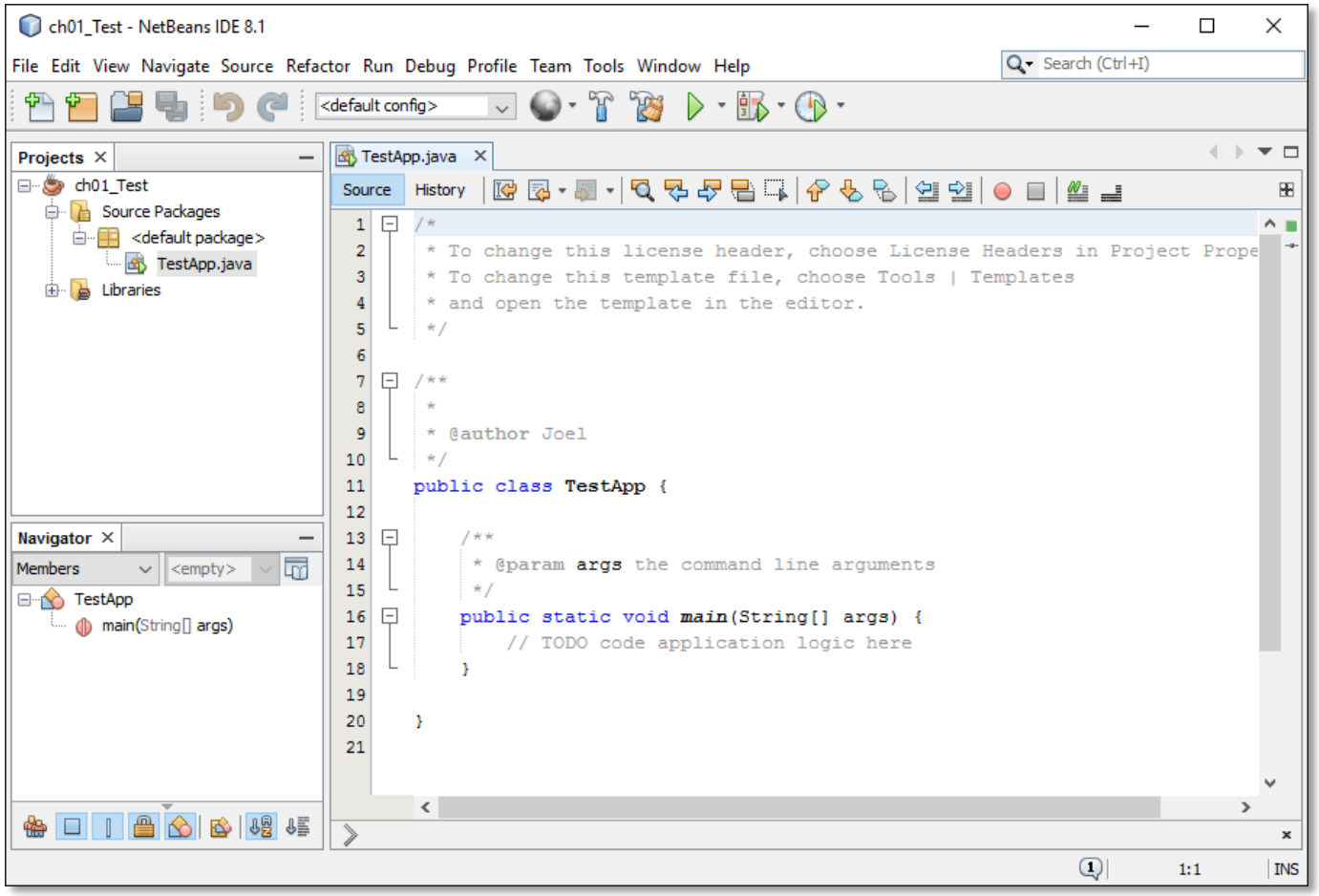
The first dialog box for creating a new project



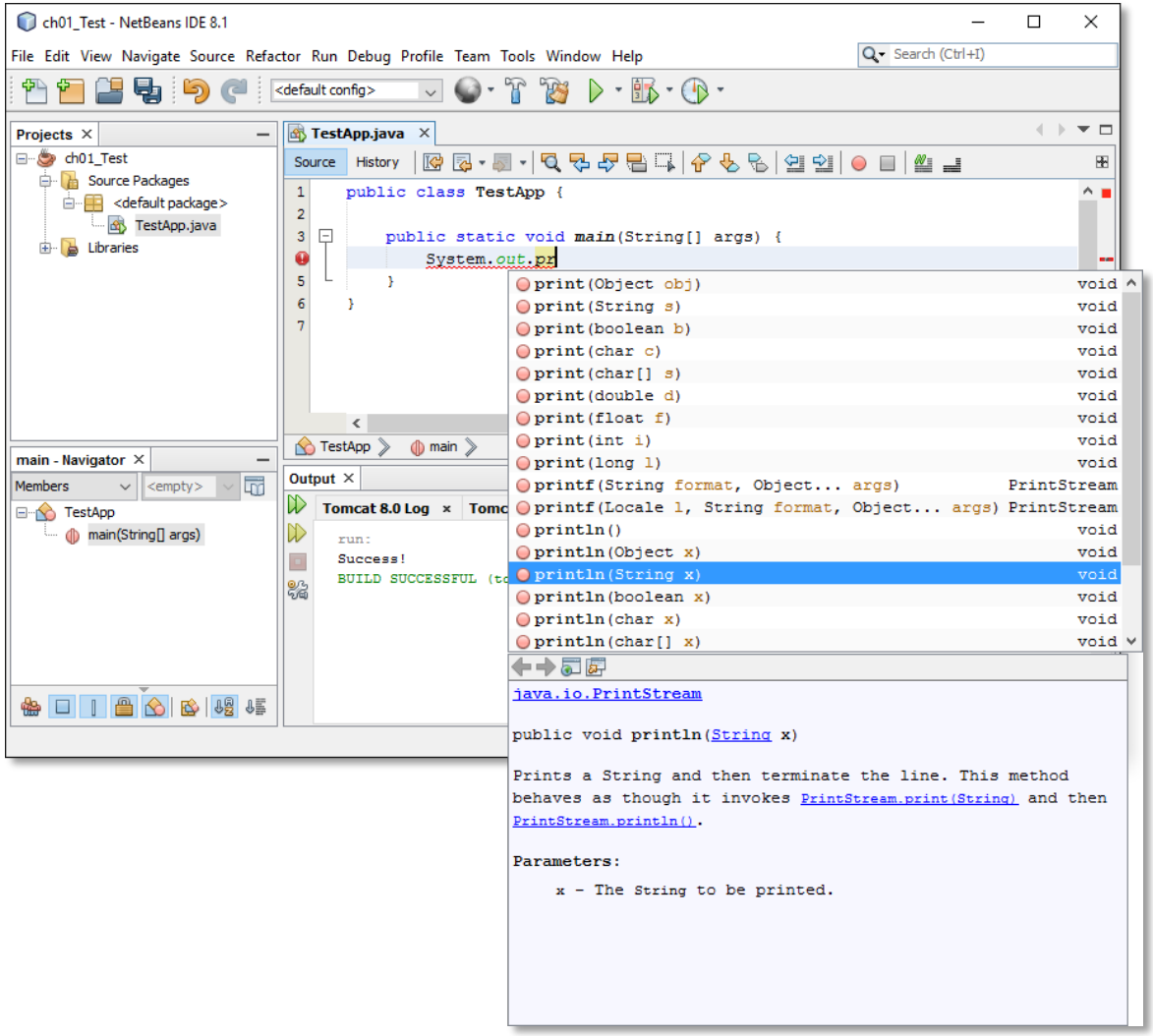
The second dialog box for creating a new project



The starting source code for a project



The code editor with a code completion list



The code editor with an error displayed