

**Java Programming  
Fundamentals:  
Academic Student Guide**



**EVALUATION COPY**

# **Java Programming Fundamentals**

## **Developers**

Ken Cohen and Gregg Frosti

## **Contributors**

James E. Roos and James Stanger, Ph.D.

## **Editors**

Susan M. Lane and David Oberman

## **Publishers**

Scott Evanskey and Joseph A. Servia

## **Project Managers**

David De Ponte, Todd Hopkins and Sheila Ramirez

## **Trademarks**

ProsoftTraining is a trademark of ProsoftTraining. All product names and services identified throughout this book are trademarks or registered trademarks of their respective companies. They are used throughout this book in editorial fashion only. No such use, or the use of any trade name, is intended to convey endorsement or other affiliation with the book. Copyrights of any screen captures in this book are the property of the software's manufacturer.

## **Disclaimer**

ProsoftTraining makes a genuine attempt to ensure the accuracy and quality of the content described herein; however, ProsoftTraining, makes no warranty, express or implied, with respect to the quality, reliability, accuracy, or freedom from error of this document or the products it describes. ProsoftTraining makes no representation or warranty with respect to the contents hereof and specifically disclaims any implied warranties of fitness for any particular purpose. ProsoftTraining disclaims all liability for any direct, indirect, incidental or consequential, special or exemplary damages resulting from the use of the information in this document or from the use of any products described in this document. Mention of any product or organization does not constitute an endorsement by ProsoftTraining of that product or corporation. Data used in examples and labs is intended to be fictional even if actual data is used or accessed. Any resemblance to, or use of real persons or organizations should be treated as entirely coincidental. ProsoftTraining makes every effort to ensure the accuracy of URLs referenced in all its material, but cannot guarantee that all URLs will be available throughout the life of a course. When this course/disk was published, all URLs were checked for accuracy and completeness. However, due to the ever-changing nature of the Internet, some URLs may no longer be available or may have been re-directed.

## **Copyright Information**

This training manual is copyrighted and all rights are reserved by ProsoftTraining. No part of this publication may be reproduced, transmitted, stored in a retrieval system, modified, or translated into any language or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual or otherwise without written permission of ProsoftTraining, 3001 Bee Caves Road, Austin, TX 78746.

Copyright © 2000 - 2002 by ProsoftTraining  
All Rights Reserved

ISBN: 1-58143-662-9



EVALUATION COPY

# Table of Contents

Course Description.....	xii
ProsoftTraining Courseware .....	xiv
Course Objectives .....	xvi
Classroom Setup.....	xvii
System Requirements .....	xvii
Conventions and Graphics Used in This Book.....	xix
<b>Lesson 1: Java Runtime Environment.....</b>	<b>1-1</b>
Pre-Assessment Questions .....	1-2
The Java Virtual Machine.....	1-3
The Java 2 Software Development Kit .....	1-4
Java Comments .....	1-5
Lesson 1 Review .....	1-8
<b>Lesson 2: Data Types, Variables and Operators.....</b>	<b>2-1</b>
Pre-Assessment Questions .....	2-2
Data Types .....	2-3
Declaring Variables .....	2-4
Variable Scope.....	2-4
Casting .....	2-6
Operators.....	2-7
Automatic Casting .....	2-10
Lesson 2 Review .....	2-14
<b>Lesson 3: Control Statements .....</b>	<b>3-1</b>
Pre-Assessment Questions .....	3-2
Code Blocks.....	3-3
Conditional Statements .....	3-4
Iterative Statements (Loops).....	3-6
Lesson 3 Review .....	3-11
<b>Lesson 4: Methods.....</b>	<b>4-1</b>
Pre-Assessment Questions .....	4-2
Java Methods .....	4-4
Return Statements .....	4-4
Calling a Method.....	4-5
Parameters.....	4-5
Pass by Value.....	4-6
Overloading.....	4-7
Lesson 4 Review .....	4-10
<b>Lesson 5: Arrays .....</b>	<b>5-1</b>
Pre-Assessment Questions .....	5-2
What Is an Array? .....	5-3
Initializing an Array.....	5-3
Objects .....	5-4
Using an Array.....	5-5

Passing an Array to a Method .....	5-5
Garbage Collection .....	5-6
Command Line Parameters .....	5-7
Lesson 5 Review .....	5-10
<b>Lesson 6: Classes and Objects .....</b>	<b>6-1</b>
Pre-Assessment Questions .....	6-2
Object-Oriented Programming .....	6-4
What Is an Object? .....	6-4
Instance and Class Members .....	6-5
Abstraction .....	6-9
Object References .....	6-10
Lesson 6 Review .....	6-13
<b>Lesson 7: Inheritance .....</b>	<b>7-1</b>
Pre-Assessment Questions .....	7-2
What Is Inheritance? .....	7-4
Overriding Methods .....	7-5
Overridden Methods and Variables .....	7-7
Lesson 7 Review .....	7-10
<b>Lesson 8: Constructors .....</b>	<b>8-1</b>
Pre-Assessment Questions .....	8-2
What Is a Constructor? .....	8-4
Using Constructors .....	8-4
The Keyword this .....	8-6
Constructor Process .....	8-8
Constructors and Callbacks .....	8-9
String and StringBuffer .....	8-10
Lesson 8 Review .....	8-16
<b>Lesson 9: Interfaces and Abstract Classes .....</b>	<b>9-1</b>
Pre-Assessment Questions .....	9-2
What Is an Interface? .....	9-4
Polymorphism .....	9-6
What Is an Abstract Class? .....	9-9
Lesson 9 Review .....	9-13
<b>Lesson 10: Packages and Access Modifiers .....</b>	<b>10-1</b>
Pre-Assessment Questions .....	10-2
Introduction to Packages and Access Modifiers .....	10-4
Packages .....	10-4
Access Modifiers .....	10-5
Java 2 Application Programming Interface .....	10-6
Encapsulation .....	10-6
Lesson 10 Review .....	10-11

<b>Lesson 11: Swing Components .....</b>	<b>11-1</b>
Pre-Assessment Questions .....	11-2
What Is the AWT? .....	11-3
What Is Swing? .....	11-4
Basic Swing Components .....	11-5
Swing Containers .....	11-20
JavaBeans .....	11-25
Lesson 11 Review .....	11-27
<b>Lesson 12: Layout Managers .....</b>	<b>12-1</b>
Pre-Assessment Questions .....	12-2
What Is a Layout Manager? .....	12-5
FlowLayout .....	12-6
GridLayout .....	12-7
BorderLayout .....	12-8
BoxLayout .....	12-10
Combining Layouts .....	12-13
Lesson 12 Review .....	12-18
<b>Lesson 13: Graphics in Java .....</b>	<b>13-1</b>
Pre-Assessment Questions .....	13-2
Graphics Class .....	13-3
Color Class .....	13-9
Font Class .....	13-10
Lesson 13 Review .....	13-15
<b>Lesson 14: The Event Delegation Model .....</b>	<b>14-1</b>
Pre-Assessment Questions .....	14-2
What Is an Event? .....	14-3
JDK 1.0 Event Handling .....	14-3
SDK 1.2 Event Handling .....	14-4
Lesson 14 Review .....	14-16
<b>Lesson 15: Inner Classes .....</b>	<b>15-1</b>
Pre-Assessment Questions .....	15-2
What Is an Inner Class? .....	15-4
Inner Classes for Event Handling .....	15-4
Lesson 15 Review .....	15-10
<b>Lesson 16: Java Applets .....</b>	<b>16-1</b>
Pre-Assessment Questions .....	16-2
Programming Applets .....	16-3
Applets and Web Browsers .....	16-4
Converting an Application into an Applet .....	16-11
Converting an Applet into an Application .....	16-13
Lesson 16 Review .....	16-17

<b>Lesson 17: Exceptions</b> .....	<b>17-1</b>
Pre-Assessment Questions .....	17-2
What Is an Exception? .....	17-4
Handling Exceptions .....	17-5
Creating User-Defined Exceptions .....	17-8
Exception Handling Tips .....	17-10
Exceptions and Inheritance .....	17-11
Lesson 17 Review .....	17-13
<b>Lesson 18: Creating Threads and Thread Methods</b> .....	<b>18-1</b>
Pre-Assessment Questions .....	18-2
What Are Threads? .....	18-3
How Operating Systems Handle Multitasking .....	18-3
Types of Threads in Java .....	18-4
Creating Threads .....	18-5
Thread Methods .....	18-8
Lesson 18 Review .....	18-17
<b>Lesson 19: Thread Synchronization</b> .....	<b>19-1</b>
Pre-Assessment Questions .....	19-2
What Is Thread Synchronization?.....	19-3
Thread Racing.....	19-3
Synchronized and the Object Monitor .....	19-4
Thread Race Condition .....	19-5
Sophisticated Thread Synchronization .....	19-9
Stopping, Suspending and Resuming Threads.....	19-13
Deadlocks.....	19-16
Lesson 19 Review .....	19-19
<b>Lesson 20: Streams and Serialization</b> .....	<b>20-1</b>
Pre-Assessment Questions .....	20-2
What Is a Stream? .....	20-3
InputStream, OutputStream, Reader and Writer .....	20-4
Files.....	20-5
Stream Classes of java.io.* .....	20-9
Serialization .....	20-17
Lesson 20 Review .....	20-24
<b>Lesson 21: Networking in Java</b> .....	<b>21-1</b>
Pre-Assessment Questions .....	21-2
What Is Networking? .....	21-3
Connecting Computers Across the Internet .....	21-3
Networking Classes of java.net.* .....	21-6
The Java Client/Server Model .....	21-7
Building the EchoServer .....	21-7
Multithreading Your Client/Server Example.....	21-13
Lesson 21 Review .....	21-20

<b>Appendixes</b> .....	<b>Appendix-1</b>
<b>Glossary</b> .....	<b>Glossary-1</b>
<b>Index</b> .....	<b>Index-1</b>
<b>Supplemental CD-ROM Contents</b> .....	<b>Supplemental CD-ROM Contents-1</b>

## List of Labs

Lab 1-1: Compiling and running your first Java program .....	1-6
Lab 2-1: Using primitive variables and operators.....	2-11
Lab 3-1: Using while and for loops .....	3-9
Lab 7-1: Implementing inheritance in Java .....	7-6
Lab 8-1: Building constructors in Java .....	8-14
Lab 9-1: Using interfaces and polymorphism in Java .....	9-11
Lab 10-1: Using encapsulation, accessors and mutators in Java .....	10-9
Lab 11-1: Creating basic Swing components .....	11-23
Lab 12-1: Creating sophisticated layouts in Java .....	12-15
Lab 13-1: Drawing to a JFrame .....	13-12
Lab 14-1: Implementing a WindowListener for event handling in Java .....	14-13
Lab 15-1: Event-enabling your Java drawing application .....	15-7
Lab 16-1: Converting a Java application into an applet .....	16-15
Lab 18-1: Creating a threaded digital clock in Java .....	18-15
Lab 19-1: Enhancing the digital clock with advanced thread techniques in Java .....	19-17
Lab 20-1: Creating a simple word processor in Java .....	20-22
Lab 21-1: Building a client/server chat system.....	21-17

## List of Figures

Figure 1-1: Java development cycle .....	1-4
Figure 2-1: Casting rules chart.....	2-7
Figure 4-1: Value of myInteger following initialization.....	4-6
Figure 4-2: Value of method local variable tmpInt .....	4-6
Figure 5-1: Default values stored in myIntArray.....	5-3
Figure 5-2: Values stored in myIntArray following initialization.....	5-5
Figure 5-3: Two references to same array .....	5-6
Figure 5-4: Lost array reference .....	5-7
Figure 6-1: tmpEmp variable .....	6-10
Figure 7-1: Hierarchy of classes .....	7-6
Figure 10-1: Access levels.....	10-5
Figure 10-2: Object encapsulation.....	10-8
Figure 11-1: Portion of Swing API and relation to AWT.....	11-5
Figure 11-2: Output of ImageIconStuff .....	11-8
Figure 11-3: Output of JButtonStuff.....	11-10
Figure 11-4: New output of JButtonStuff is same .....	11-10
Figure 11-5: Output of LabelStuff .....	11-12
Figure 11-6: Output of JScrollBarStuff .....	11-13
Figure 11-7: Output of JTextFieldStuff .....	11-14

Figure 11-8: Output of JTextAreaStuff.....	11-15
Figure 11-9: Output of JScrollPaneStuff.....	11-17
Figure 11-10: Output of JFileChooserStuff.....	11-20
Figure 11-11: Output of JFrameStuff.....	11-21
Figure 11-12: Output of JFrameStuff with JButton.....	11-22
Figure 11-13: Output of JPanelStuff.....	11-23
Figure 12-1: FlowLayout resized.....	12-6
Figure 12-2: Output of FlowLayoutStuff.....	12-7
Figure 12-3: Output of GridLayoutStuff.....	12-8
Figure 12-4: Example of BorderLayout regions.....	12-8
Figure 12-5: Output of BorderLayoutStuff.....	12-9
Figure 12-6: Output of BoxLayoutStuff.....	12-10
Figure 12-7: Output of BoxLayoutStuff with struts.....	12-12
Figure 12-8: Output of BoxLayoutStuff with glue.....	12-13
Figure 12-9: Example of a sophisticated layout.....	12-13
Figure 12-10: Output of NestedLayoutStuff.....	12-15
Figure 12-11: Sophisticated GUI model.....	12-16
Figure 13-1: AWT class hierarchy (Graphics section).....	13-4
Figure 13-2: Coordinate system for positioning graphics.....	13-5
Figure 13-3: Output of PaintingStuff.....	13-6
Figure 13-4: Output of PaintingStuff with rectangle.....	13-9
Figure 13-5: Output of PaintingStuff with colors.....	13-10
Figure 13-6: Output of PaintingStuff with fonts.....	13-12
Figure 13-7: JFrame model.....	13-13
Figure 14-1: Interface with JButtons and JTextArea.....	14-10
Figure 14-2: Sources and event types generated.....	14-14
Figure 15-1: Drawing application.....	15-7
Figure 16-1: JApplet class hierarchy.....	16-3
Figure 16-2: Output of MyJApplet.....	16-8
Figure 16-3: Output of MyApplication.....	16-12
Figure 16-4: Output of MyJApplet after conversion from application.....	16-13
Figure 16-5: Output of MyJApplet after conversion to application.....	16-14
Figure 17-1: Exception class hierarchy.....	17-5
Figure 17-2: Overridden method.....	17-11
Figure 18-1: Main thread.....	18-5
Figure 18-2: Thread states.....	18-11
Figure 18-3: DigitalClock interface.....	18-15
Figure 19-1: Deadlock.....	19-16
Figure 20-1: Wrapper streams.....	20-14
Figure 20-2: Word processor GUI.....	20-22
Figure 21-1: Connecting Computer A to Computer B.....	21-5
Figure 21-2: Client GUI.....	21-17

## List of Tables

Table 2-1: Primitive data types in Java.....	2-3
Table 2-2: Default values for class variables in Java.....	2-5

---

Table 2-3: Relational operators in Java .....	2-8
Table 8-1: Common methods of String .....	8-12
Table 16-1: Methods inherited from the Applet class .....	16-10
Table 18-1: Methods used to control single thread.....	18-9
Table 20-1: Methods of File class.....	20-6
Table 20-2: Useful stream classes.....	20-9
Table 21-1: Well-known ports and their protocols.....	21-5
Table 21-2: Common networking classes.....	21-6

EVALUATION COPY



EVALUATION COPY

---

## Course Description

---

The *Java Programming Fundamentals* course teaches you how to write Java applications and applets. You will learn the Java language mechanics found in other programming languages, such as variables, iterations, control statements, methods and arrays. You will also discuss object-oriented theory as it relates to Java. You will create Graphical User Interfaces (GUIs) for both applications and applets, emphasizing components, layouts, and graphics. The course concludes with an in-depth study and implementation of the SDK 1.2 event delegation model, an essential element in further Java studies. You will also complete a course-long project to create an operational client/server messaging system.

Upon completion of this course, you will be experienced in writing Java applications and applets. We recommend and will frequently refer you to another excellent Java resource, David Flanagan's *Java in a Nutshell*. Students who plan to pursue the jCert Level I Certification or the Sun Java Programmer certification are encouraged to study the CIW *Sun Certified Java Programming Exam Preparation Guide* when preparing for the exams.

---

### Length

---

*Java Programming Fundamentals* is a 30-hour course.

---

### Series

---

*Java Programming Fundamentals* is a single course representing the CIW Java Programming series. This series is part of the Master CIW Enterprise Developer track as well as the jCert Initiative. For students interested in the jCert Level I certification or the Sun Java Programmer certification, we recommend the CIW *Sun Certified Java Programmer Exam Preparation Guide*.

---

### Prerequisites

---

Students must have a basic knowledge of programming fundamentals before taking this course.

# ProsoftTraining Courseware

This coursebook was developed for instructor-led training and will assist you during class. Along with comprehensive instructional text and objectives checklists, this coursebook provides easy-to-follow hands-on labs and a glossary of course-specific terms. It also provides Internet addresses needed to complete some labs, although due to the constantly changing nature of the Internet, some addresses may no longer be valid.

The student coursebook is organized in the following manner:

course title	
table of contents	
list of labs	
list of figures	
list of tables	
lessons	
lesson objectives	
pre-assessment questions	
narrative text	
<input checked="" type="checkbox"/> graphics	
<input checked="" type="checkbox"/> tables and figures	
<input checked="" type="checkbox"/> warnings	
<input checked="" type="checkbox"/> tech notes	
labs	
<input checked="" type="checkbox"/> graphics	
<input checked="" type="checkbox"/> tables and figures	
<input checked="" type="checkbox"/> warnings	
<input checked="" type="checkbox"/> tech notes	
lesson summary	
lesson review	
appendixes	
glossary	
index	
supplemental CD	

---

When you return to your home or office, you will find this coursebook to be a valuable resource for applying the skills you have learned. Each lesson concludes with questions that review the material. Lesson review questions are provided as a study resource only and in no way guarantee a passing score on CIW exams.

The course is available in either an academic or a learning center version, and each version has an instructor book and a student book. Check your book to verify that you have the correct version, and whether it is an instructor or a student book. Following is a brief description of each version.

- **Academic:** Designed for students in an academic classroom environment; typically taught over a quarter (10-week) or semester (16-week) time period. Example syllabi for both time frames are included on the instructor CD-ROM. The instructor's book and CD-ROM contain all answers, as well as optional labs (computer-based labs), quizzes, a course assessment, and handouts for the instructor to assign during class or as homework. No answers exist in the student book or on the student CD-ROM. Students must obtain answers from the instructor.
- **Learning Center:** Designed for students in a learning center classroom environment; typically taught over a one- to five-day time period (depending on the length of the course). An example implementation table is included on the instructor CD-ROM. Similar to the academic version, the instructor's book and CD-ROM contain all answers, as well as optional labs (computer-based labs), quizzes, a course assessment, and handouts for the instructor to assign during class or as homework. However, the student CD-ROM also contains answers, including those to the pre-assessment questions, labs, review questions, optional labs, quizzes, and the course assessment.

## Course Objectives

---

After completing this course, you will be able to:

- Describe the Java Runtime Environment (JRE).
- Use Java variables, control statements, methods and arrays.
- Discuss object-oriented theory, including abstraction, encapsulation, inheritance and polymorphism.
- Describe method overloading and overriding.
- Use Java static and instance members.
- Create Java constructors.
- Identify the differences between instance and class members.
- Use Java abstract classes and interfaces.
- Use Java Strings and StringBuffer.
- Describe Java packages and accessibility.
- Use the Java Abstract Windowing Toolkit (AWT) and Swing components central to SDK 1.2.
- Use the SDK 1.2.x event delegation model.
- Define applets and the applet life cycle.
- Throw exceptions.
- Create threads.
- Use streams.
- Create network applications in Java.

## Classroom Setup

Your instructor has probably set up the classroom computers based on the system requirements listed below. Most software configurations on your computer are identical to those on your instructor's computer. However, your instructor may use additional software to demonstrate network interaction or related technologies.

## System Requirements

### Hardware

The following table summarizes the hardware requirements for all courses in the CIW program. Each classroom should be equipped with enough personal computers to accommodate each student and the instructor with his or her own system.

*Note: The CIW hardware requirements are similar to the lowest system requirements for Microsoft implementation (Level 1 requirements) except that CIW requires increased hard disk space (8 GB) and RAM (128 MB). This comparison may be helpful for the many training centers that implement CIW and are also CTEC because personnel at these centers are familiar with the Microsoft hardware specifications.*

CIW hardware specifications	Greater than or equal to the following
Processor	Intel Pentium II (or equivalent) personal computer with processor speed greater than or equal to 300 MHz
L2 cache	256 KB
Hard disk	8-GB hard drive
RAM	At least 128 MB
CD-ROM	32X
Network interface card (NIC)	10BaseT or 100BaseTX (10 or 100 Mbps)
Sound card/speakers	Required for instructor's station, optional for student stations
Video adapter	At least 4 MB
Monitor	15-inch monitor
Network hubs	Two 10-port 10BaseT or 100BaseTX (10 or 100 Mbps) hubs
Router	Multi-homed system with three NICs (Windows NT 4.0/2000 server)*

\* *Must meet universal CIW hardware requirements.*

## Software

---

The recommended software configurations for computers used to complete the exercises in this book are as follows.

- Microsoft Windows 2000 or Millennium Edition (Me), Red Hat Linux 7.x, or any operating system with an available Java 2 SDK
- Sun Java 2 Software Development Kit (SDK), Standard Edition, version 1.2 or later
- Netscape Navigator 6 or higher.

## Connectivity

---

The minimum requirement is a networked classroom with TCP/IP installed. Each computer should have its own IP address, or have local loop-back capability (IP 127.0.0.1). Internet connectivity is not required, but it is helpful.

## Conventions and Graphics Used in This Book

The following conventions are used in Prosoft coursebooks.

<b>Terms</b>	Technology terms defined in the margins are indicated in <b>bold</b> the first time they appear in the text. Not every word in bold is a term requiring definition.
<b>Lab Text</b>	Text that you enter in a lab appears in <b>bold</b> . Names of components that you access or change in a lab also appear in <b>bold</b> .
<b>Notations</b>	<i>Notations or comments regarding screenshots, labs or other text are indicated in italic type.</i>
<b>Program Code or Commands</b>	Text used in program code or operating system commands appears in the Lucida Sans Typewriter font.

The following graphics are used in Prosoft coursebooks.



*Tech Notes* point out exceptions or special circumstances that you may find when working with a particular procedure. Tech Notes that occur within a lab are displayed without the graphic.



*Tech Tips* offer special-interest information about the current subject.



*Warnings* alert you about cautions to observe or actions to avoid.



This graphic signals the start of a lab or other hands-on activity.



Each lesson summary includes an *Application Project*. This project is designed to provoke interest and apply the skills taught in the lesson to your daily activities.



Each lesson concludes with a summary of the skills and objectives taught in that lesson. You can use the Skills Review checklist to evaluate what you have learned.



EVALUATION COPY

# Lesson 1:

# Java Runtime Environment

---

---

## **Objectives**

By the end of this lesson, you will be able to:

- ↻ Identify the differences between stand-alone applications and applets.
- ↻ Describe the role of the Java Virtual Machine (JVM).
- ↻ Create a simple Java program.
- ↻ Use Java comments.
- ↻ Compile and execute a Java program.
- ↻ Describe the differences between \*.java and \*.class files.

## Pre-Assessment Questions

---

1. What type of files are generated by the javac compiler?
  - a. Text files
  - b. \*.java files
  - c. \*.class files
  - d. \*.exe files
  
2. Which of the following employs the Java syntax for a multiline comment?
  - a. `// This is a comment.`
  - b. `/* This is a comment */`
  - c. `/** This is a comment */`
  - d. `/** This is a comment. */`
  
3. What benefit does the use of a virtual machine provide to Java developers?

---

---

---

# The Java Virtual Machine

Java is an object-oriented programming language developed at Sun Microsystems that is widely known for its portability. This platform independence makes Java ideal for Internet development. The Internet connects a wide variety of hardware and software systems. Java can be used to create a single program that will operate the same on many platforms. Even if you have never programmed using Java before, you are probably familiar with Java **applets**. Applets are small Java programs that can only be executed from within a Web browser. A Web site employing an applet will provide the same compiled program to Web browsers on all platforms. Java can also be used to create stand-alone applications.

The **Java Virtual Machine (JVM)** gives Java its platform independence. One of the primary design goals of the Java language is to enable the same code to run on any platform. The JVM is a software program that behaves like an entire computer. By using this artificial computer on different computer platforms (UNIX, Win32, Macintosh, and so forth), you can reuse programs without creating a version for each platform. Your Java programs will always run on a JVM. A Java program can be run on any platform for which a JVM is available.

You will begin writing a simple program by defining a **class**. A class is a type of container into which all Java code must be placed. You need not be familiar with classes already; you will learn more about them in a later lesson. A class consists of a name and a pair of curly braces to contain the body of the class. By convention, the class name should start with a capital letter as in the following example:

```
class HelloWorld
{
}
```

When you write a stand-alone application, the Java Virtual Machine must know where to begin executing the code. To begin executing code, the JVM looks for and calls a special **method** by the name of `public static void main(String[] args)`. The JVM uses this method to begin executing your program. Study the following code example:

```
class HelloWorld
{
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
    }
}
```

**applet**

A small Java program that must be run from a Web browser.

**Java Virtual Machine (JVM)**

The artificial computer that runs Java programs.

**class**

A type of container into which all Java code must be placed. Also, the template or blueprint for an object.

**method**

A procedural part of a class that contains executable Java statements.

This program is complete. Note that Java programs are case-sensitive; the word `Class` is not read the same as `class`. You must save the program with your text editor as `HelloWorld.java` (giving it the same name as your class), and compile it using a Java compiler. You can use the `javac` program if you have the Java 2 Software Development Kit (SDK) supplied by Sun.

```
javac HelloWorld.java
```

The Java compiler converts, or compiles, your source code (saved as `HelloWorld.java`, a text file) and creates a `HelloWorld.class` file. This `.class` file is no longer a standard text file, but a file compiled into bytecode.

**bytecode**

A platform-neutral file consisting of bytes that the JVM can execute.

**Bytecode** is another element that makes Java different from other programming languages. Other compiled programming languages generate machine code binaries, which are files that contain native machine language statements. To make Java portable, the `javac` compiler generates bytecode. Java bytecode is composed of an instruction set native only to the JVM. Because JVMs exist for multiple platforms, the Java bytecode is portable. Portability, however, comes at a price: speed.

Finally, to launch your program, you must run it in a JVM. You can accomplish this by invoking the Java interpreter `java` as follows:

```
java HelloWorld
```

Note that you type `java HelloWorld`, not `java HelloWorld.class`. When you enter the proper command, the JVM will display "Hello World!" in a command line window. The Java application development cycle is shown in Figure 1-1.

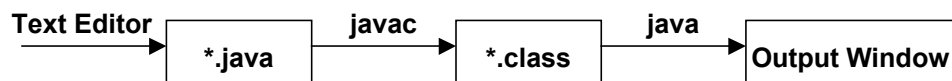


Figure 1-1: Java development cycle

## The Java 2 Software Development Kit

The development platform used in this class is the Java 2 Software Development Kit (SDK), Standard Edition, version 1.2.2 or later. The Java 2 SDK was previously known as the Java Development Kit (JDK) 1.2. In fact, if you check your installation of the Java 2 SDK, you will find that the directory reads JDK 1.2.2. The Java 2 SDK is backward-compatible with JDK 1.1. It contains several components, including:

The Java Runtime Environment (JRE), which provides the JVM.

- The development tools and compilation libraries necessary to create Java applications, servlets and applets. This grouping includes the Java compiler, debugging programs, and tools to run applets without a browser.

This coursebook is written to the Java 2 specification.

## Java Comments

Throughout the course, you will find it useful to add comments to your code. Java supports three types of comments; two will be recognizable to those familiar with C or C++, and the third type is unique to Java.

The three types of comments are as follows:

- Single-line comment     //
- Multiline comment     /\* ... \*/
- Javadoc comment       /\*\* ... \*/

The following code shows examples of all three comment types:

```
class HelloWorld
{
    /**
     * This is a javadoc comment. It is a multiline comment
     * unique to Java.
     * If you use the javadoc utility (which comes with Java),
     * it will automatically create HTML-based documentation
     * for you. The online help that comes with the Sun SDK was
     * created using javadoc comments.
     */
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
        // Two forward slashes indicate a single-line comment.
        // Notice that each line must begin with the
        // two forward slashes.

        /*
         * This is a multiline comment. It is possible to
         * have multiple lines of code within this block.
         */
    }
}
```

You will use the single-line and multiline comments similar to C++.



### Lab 1-1: Compiling and running your first Java program

In this lab, you will write, compile and run your first Java program. As you finish the lab, note the difference between the source file you write (\*.java) and the bytecode that is generated (\*.class). A Java application always starts with the `public static void main(String[] args)` method.

1. **Editor:** Create a new text file and enter the following Java code:

```
class HelloWorld
{
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
    }
}
```

2. **Editor:** Save the file as **HelloWorld.java**.
3. **Prompt:** Compile HelloWorld.java to generate a \*.class file. From the command line, enter the following:  
  

```
javac HelloWorld.java
```
4. **Prompt:** Execute HelloWorld. From the command line, enter the following:  
  

```
java HelloWorld
```
5. **Editor:** Add another `System.out.println("Hello Solar System!");` to the main method.
6. **Prompt:** Compile and execute your program.
7. *(Optional)* Modify the `public static void main(String[] args)` method in any way you choose (for example, delete the word `void`). What happens?
8. *(Optional)* Add some comments to the file to document each step that you made while programming this application.

---

## Lesson Summary



### Application project

---

When Java was originally released, it was widely praised for its platform independence. This platform independence made Java applets possible, allowing Web site operators to develop applications that could be downloaded and executed by any client with a compatible browser, regardless of platform. Search the Web for at least five examples of Java applets in which the functionality could not be easily duplicated using server-side or client-side scripting technologies.



### Skills review

---

Java consists of three types of programs: applications, applets, and servlets. In this lesson, you learned that applications contain a main method. After the program is written as a text source file (having a name \*.java), the program is converted into bytecode (having a name \*.class) by the Java compiler. The compiled bytecode can be executed using the Java interpreter. In the following lessons, you will learn the basic syntax of the Java programming language.

Now that you have completed this lesson, you should be able to:

- ✓ Identify the differences between stand-alone applications and applets.
  - ✓ Describe the role of the Java Virtual Machine (JVM).
  - ✓ Create a simple Java program.
  - ✓ Use Java comments.
  - ✓ Compile and execute a Java program.
  - ✓ Describe the differences between \*.java and \*.class files.
-

## Lesson 1 Review

---

1. How does a Java applet differ from a Java application?

---

---

---

2. What is the Java Virtual Machine (JVM)?

---

---

---

3. What is a class?

---

---

---

4. What is a method?

---

---

---

5. Are Java programs case-sensitive? Give an example that describes its policy.

---

---

6. What is bytecode? How does it affect Java's portability?

---

---

---

---

---

---

7. Identify the three types of comments you can use in Java.

---

---

8. How does a \*.java file differ from a \*.class file?

---

---

EVALUATION COPY