Chapter 1 – Introduction to Computers, the Internet, and the Web

Outline

Chapter 1 – Introduction to Computers, the Internet, and the Web

1.1 Introduction

- *Java How to Program, Fifth Edition*
  - Java 2 Standard Edition
  - Object-oriented programming

1.2 What Is a Computer?

- Computer
  - Performs computations and makes logical decisions
  - Millions / billions times faster than human beings
- Computer programs
  - Sets of instructions for which computer processes data
- Hardware
  - Physical devices of computer system
- Software
  - Programs that run on computers

1.3 Computer Organization

- Six logical units of computer system
  - Input unit
    - Mouse, keyboard
  - Output unit
    - Printer, monitor, audio speakers
  - Memory unit
    - Retains input and processed information
  - Arithmetic and logic unit (ALU)
    - Performs calculations
  - Central processing unit (CPU)
    - Supervises operation of other devices
  - Secondary storage unit
    - Hard drives, floppy drives

1.4 Evolution of Operating Systems

- Batch processing
  - One job (task) at a time
  - Operating systems developed
    - Programs to make computers more convenient to use
    - Switch jobs easier
- Multiprogramming
  - “Simultaneous” jobs
  - Timesharing operating systems

1.5 Personal, Distributed and Client/Server Computing

- Personal computing
  - Computers for personal use
- Distributed computing
  - Computing performed among several computers
- Client/server computing
  - Servers offer common store of programs and data
  - Clients access programs and data from server

1.6 Machine Languages, Assembly Languages and High-Level Languages

- Machine language
  - "Natural language" of computer component
  - Machine dependent
- Assembly language
  - English-like abbreviations represent computer operations
  - Translator programs convert to machine language
- High-level language
  - Allows for writing more "English-like" instructions
    - Contains commonly used mathematical operations
  - Compiler convert to machine language
- Interpreter
  - Execute high-level language programs without compilation

1.7 History of C++

- C++
  - Evolved from C
    - Evolved from BCPL and B
  - Provides object-oriented programming capabilities
- Objects
  - Reusable software components that model real-world items

1.8 History of Java

- Java
  - Originally for intelligent consumer-electronic devices
  - Then used for creating Web pages with *dynamic content*
  - Now also used for:
    - Develop large-scale enterprise applications
    - Enhance WWW server functionality
    - Provide applications for consumer devices (cell phones, etc.)

1.9 Java Class Libraries

- Classes
  - Include *methods* that perform tasks
    - Return information after task completion
  - Used to build Java programs
- Java contains class libraries
  - Known as Java APIs (Application Programming Interfaces)

1.10 FORTRAN, COBOL, Pascal and Ada

- Fortran
  - FORmula TRANslator
- COBOL
  - COmmon Business Oriented Language
- Pascal
  - Structured programming
- Ada
  - Multitasking

1.11 BASIC, Visual Basic, Visual C++, C# and .NET

- BASIC
  - Beginner's All-Purpose Symbolic Instruction Code
- Visual Basic .NET
  - Framework Class Library (FLC)
- Visual C++
  - Microsoft Foundation Classes (MFC)
- C#
  - C-Sharp
- .NET
  - .NET platform

1.12 The Internet and the World Wide Web

- Internet
  - Developed more than four decades ago with DOD funding
  - Originally for connecting few main computer systems
  - Now accessible by hundreds of millions of computers
- World Wide Web (WWW)
  - Allows for locating/viewing multimedia-based documents

1.13 Basics of a Typical Java Environment

- Java programs normally undergo five phases
  - Edit
    - Programmer writes program (and stores program on disk)
  - Compile
    - Compiler creates *bytecodes* from program
  - Load
    - Class loader stores bytecodes in memory
  - Verify
    - Verifier ensures bytecodes do not violate security requirements
  - Execute
    - Interpreter translates bytecodes into machine language

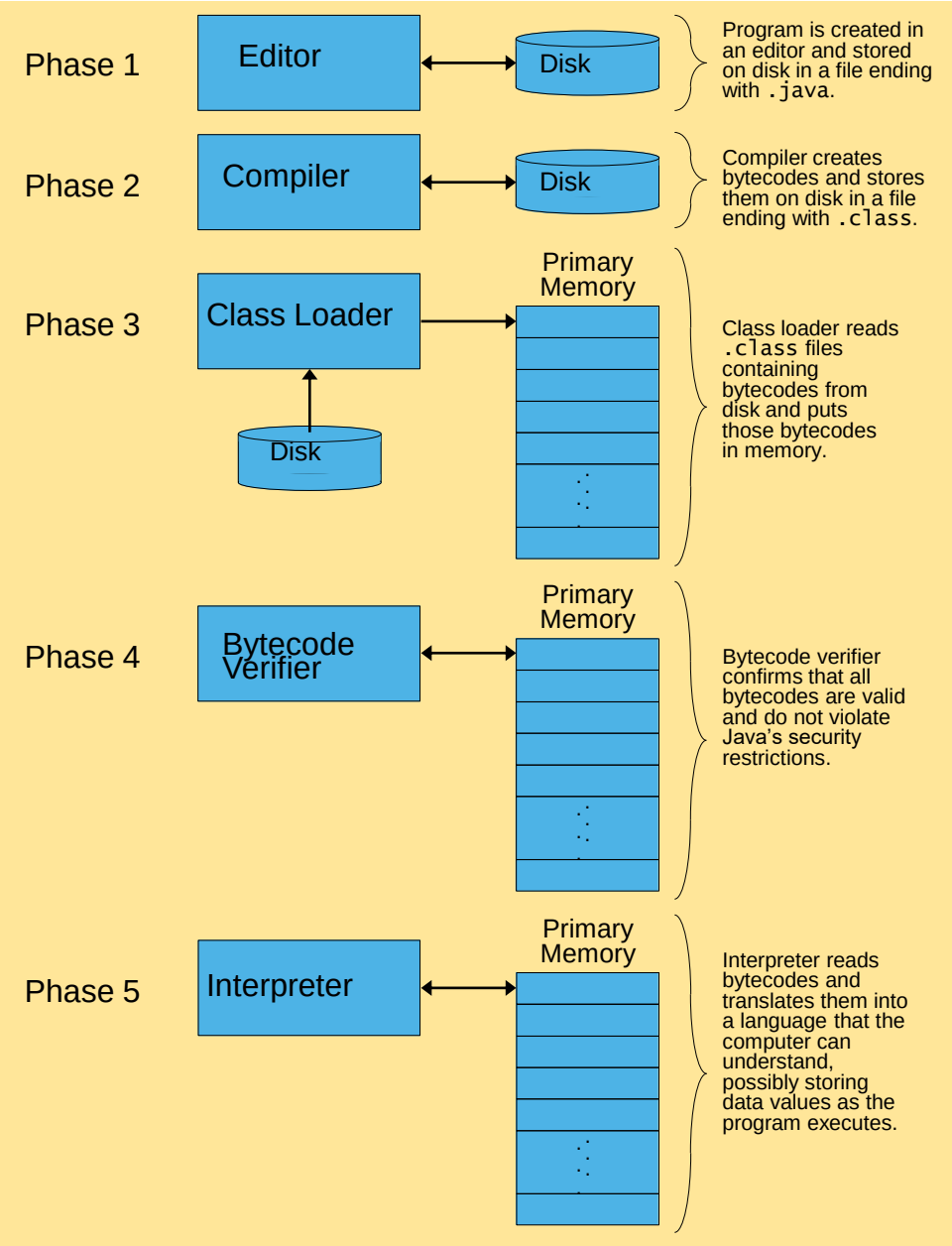

Fig. 1.1 Typical Java environment.

1.14 General Notes about Java and This Book

- Geared for novice programmers
- We stress clarity

1.15 Thinking About Objects: Introduction to Object Technology and the Unified Modeling Language

- Object orientation
- Unified Modeling Language (UML)
  - Graphical language that uses common notation
  - Allows developers to represent object-oriented designs

1.15 Thinking About Objects (cont.)

- Objects
  - Reusable software components that model real-world items
  - Look all around you
    - People, animals, plants, cars, etc.
  - Attributes
    - Size, shape, color, weight, etc.
  - Behaviors
    - Babies cry, crawl, sleep, etc.

1.15 Thinking About Objects (cont.)

- Object-oriented design (OOD)
  - Models real-world objects
  - Models communication among objects
  - *Encapsulates* attributes and operations (behaviors)
    - Information hiding
    - Communication through well-defined interfaces
- Object-oriented language
  - Programming in object oriented languages is called *object-oriented programming* (*OOP*)
  - Java

1.15 Thinking About Objects (cont.)

- Object-Oriented Analysis and Design (OOA/D)
  - Essential for large programs
  - Analyze program requirements, then develop solution
  - UML
    - Unified Modeling Language

1.15 Thinking About Objects (cont.)

- History of the UML
  - Need developed for process with which to approach OOA/D
  - Brainchild of Booch, Rumbaugh and Jacobson
  - Object Management Group (OMG) supervised
  - Version 1.4 is current version
    - Version 2.0 scheduled tentatively for release in 2003

1.15 Thinking About Objects (cont.)

- UML
  - Graphical representation scheme
  - Enables developers to model object-oriented systems
  - Flexible and extendible

1.16 Discovering Design Patterns: Introduction

- Effective design crucial for large programs
- Design patterns
  - Proven architectures for developing object-oriented software
    - Architectures created from accumulated industry experience
  - Reduce design-process complexity
  - Promotes design reuse in future systems
  - Helps identify common design mistakes and pitfalls
  - Helps design independently of implementation language
  - Establishes common design “vocabulary”
  - Shortens design phase in software-development process

1.16 Discovering Design Patterns (cont.)

- Design patterns
  - Similar to architectural elements
    - arches and columns
  - Used by developers to construct sets of classes and objects
- Developers
  - Familiarity with patterns to understand how to use patterns

1.16 Discovering Design Patterns (cont.)

- History of Design Patterns
  - Gamma, Helm, Johnson and Vlissides
    - "Gang of Four"
    - *Design Patterns, Elements of Reusable Object-Oriented Software* (Addison Wesley: 1995)
    - Established 23 design patterns
      - Creational
        - Instantiate objects
      - Structural
        - Organize classes and objects
      - Behavioral
        - Assign responsibilities to objects