Object Oriented Programming

Course Instructor: Dr. R. Shathanaa

Java

Java and a Typical Java Development Environment

- Microprocessors are having a profound impact in intelligent consumer-electronic devices.
- 1991
  - Recognizing this, **Sun Microsystems** funded an internal corporate research project led by **James Gosling**, which resulted in a C++-based object-oriented programming language Sun called Java.
  - Key goal of Java is to be able to write programs that will run on a great variety of computer systems and computer-control devices.
  - This is sometimes called "write once, run anywhere."

Java and a Typical Java Development Environment (Cont.)

- 1993
  - The web exploded in popularity
  - Sun saw the potential of using Java to add dynamic content to web pages.
- Java garnered the attention of the business community because of the phenomenal interest in the web.
- Java is used to develop large-scale enterprise applications, to enhance the functionality of web servers, to provide applications for consumer devices and for many other purposes.

Java and a Typical Java Development Environment (Cont.)

- Sun Microsystems was acquired by Oracle in 2010.
- As of 2010 97% of enterprise desktops, three billion handsets, and 80 million television devices run Java.
- Java was the most widely used software development language in the world.

1.1 Why Java

- Java is the preferred language for meeting many organizations' enterprise programming needs.
- Java has become the language of choice for implementing Internet-based applications and software for devices that communicate over a network.
- Java USED TO BE most widely used computer programming language.
- **Java and JavaScript**
- Although their names are quite similar and they are both used to create dynamic tools and games on a Web page, Java and JavaScript are different languages.

1.1 Introduction (Cont.)

- The language and its frameworks allow building software that is scalable, highly secure and powerful, which are the three pillars of modern applications
- OOL:
- Java, C++, C#, Python, R, PHP, JavaScript, Kotlin, MATLAB

1.1 Introduction (Cont.)

- Java Editions: SE, EE and ME
  - SE – Standard Edition (Java SE)
  - Used for developing cross-platform, general-purpose applications.
  - Java is used in such a broad spectrum of applications that it has two other editions.
  - The Java Enterprise Edition (Java EE)
  - Geared toward developing large-scale, distributed networking applications and web-based applications.

1.1 Introduction (Cont.)

- Java Micro Edition (Java ME)
- geared toward developing applications for small, memory-constrained devices, such as smartphones.
- Google's Android operating system
- used on numerous smartphones, tablets (small, lightweight mobile computers with touch screens), e-readers and other devices—uses a customized version of Java not based on Java ME.

Version	Date
JDK Beta	1995
JDK1.0	January 23, 1996
JDK 1.1	February 19, 1997
J2SE 1.2	December 8, 1998
J2SE 1.3	May 8, 2000
J2SE 1.4	February 6, 2002
J2SE 5.0	September 30, 2004
Java SE 6	December 11, 2006
Java SE 7	July 28, 2011
Java SE 8	March 18, 2014
Java SE 9	September 21, 2017
Java SE 10	March 20, 2018
Java SE 11	September 25, 2018
Java SE 12	March 19, 2019
Java SE 13	September 17, 2019
Java SE 14	March 17, 2020
Java SE 15	September 15, 2020
Java SE 16	March 16, 2021

1.9 Java and a Typical Java Development Environment (Cont.)

- Java Class Libraries
  - Rich collections of existing classes and methods
  - Also known as the Java APIs (Application Programming Interfaces).

Bookmark this:

  - https://docs.oracle.com/en/java/javase/16/docs/api/index.html

Performance Tip 1.1

Using Java API classes and methods instead of writing your own versions can improve program performance, because they're carefully written to perform efficiently. This also shortens program development time.

Portability Tip 1.1

Although it's easier to write portable programs (i.e., programs that can run on many different types of computers) in Java than in most other programming languages, differences between compilers, JVMs and computers can make portability difficult to achieve. Simply writing programs in Java does not guarantee portability.

1.9 Java and a Typical Java Development Environment (Cont.)

- Java programs normally go through five phases
  - edit
  - compile
  - load
  - verify
  - execute.

What are the phases in executing a C program?

- Download the JDK and its documentation from
  - `www.oracle.com/technetwork/java/javase/downloads/index.html`.
- Visit Oracle's New to Java Center at:
  - `www.oracle.com/technetwork/topics/newtojava/overview/index.html`

1.9 Java and a Typical Java Development Environment (Cont.)

- Phase 1 consists of editing a file
  - Type a Java program (source code) using the editor.
  - Make any necessary corrections.
  - Save the program.
  - A file name ending with the .java extension indicates that the file contains Java source code.

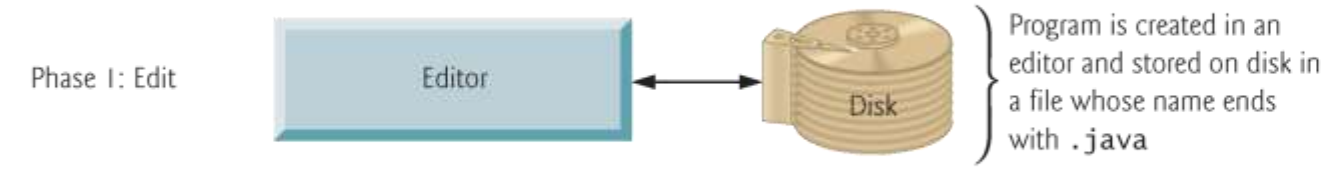


Fig. 1.6 | Typical Java development environment—editing phase.

1.9 Java and a Typical Java Development Environment (Cont.)

- Linux editors: `vi` and `gedit`.
- Windows editors:
  - Notepad
  - Notepad++
- Integrated development environments (IDEs)
  - Provide tools that support the software development process, including editors for writing and editing programs and debuggers for locating logic errors—errors that cause programs to execute incorrectly.

1.9 Java and a Typical Java Development Environment (Cont.)

- Popular IDEs
  - Eclipse (`www.eclipse.org`)
  - NetBeans (`www.netbeans.org`).
  - jGRASP™ IDE (`www.jgrasp.org`)
  - DrJava IDE (`www.drjava.org/download.shtml`)
  - BlueJ IDE (`www.bluej.org/`)
  - TextPad® Text Editor for Windows® (`www.textpad.com/`)

1.9 Java and a Typical Java Development Environment (Cont.)

- Phase 2: Compiling a Java Program into Bytecodes
  - Use the command javac (the Java compiler) to compile a program. For example, to compile a program called `Welcome.java`, you'd type
    - `javac Welcome.java`
  - If the program compiles, the compiler produces a .class file called `Welcome.class` that contains the compiled version of the program.

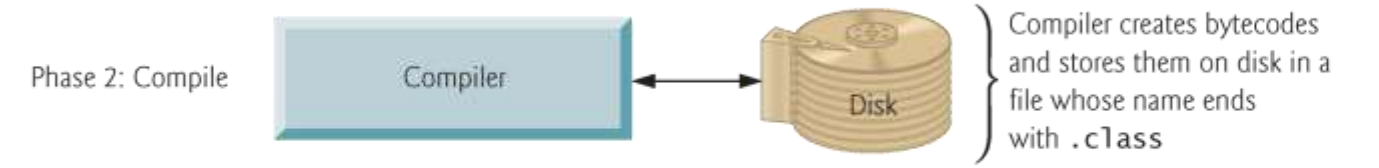


Fig. 1.7 | Typical Java development environment—compilation phase.

1.9 Java and a Typical Java Development Environment (Cont.)

- Java compiler translates Java source code into bytecodes that represent the tasks to execute.
- Bytecodes are executed by the Java Virtual Machine (JVM)—a part of the JDK and the foundation of the Java platform.
- Virtual machine (VM)—a software application that simulates a computer
  - Hides the underlying operating system and hardware from the programs that interact with it.
- If the same VM is implemented on many computer platforms, applications that it executes can be used on all those platforms.

1.9 Java and a Typical Java Development Environment (Cont.)

- Bytecodes are platform independent
  - They do not depend on a particular hardware platform.
- Bytecodes are portable
  - The same bytecodes can execute on any platform containing a JVM that understands the version of Java in which the bytecodes were compiled.
- The JVM is invoked by the java command. For example, to execute a Java application called `welcome`, you'd type the command
  - `java welcome`

1.9 Java and a Typical Java Development Environment (Cont.)

- Phase 3: Loading a Program into Memory
  - The JVM places the program in memory to execute it—this is known as loading.
  - Class loader takes the `.class` files containing the program's bytecodes and transfers them to primary memory.
  - Also loads any of the `.class` files provided by Java that your program uses.
- The `.class` files can be loaded from a disk on your system or over a network.

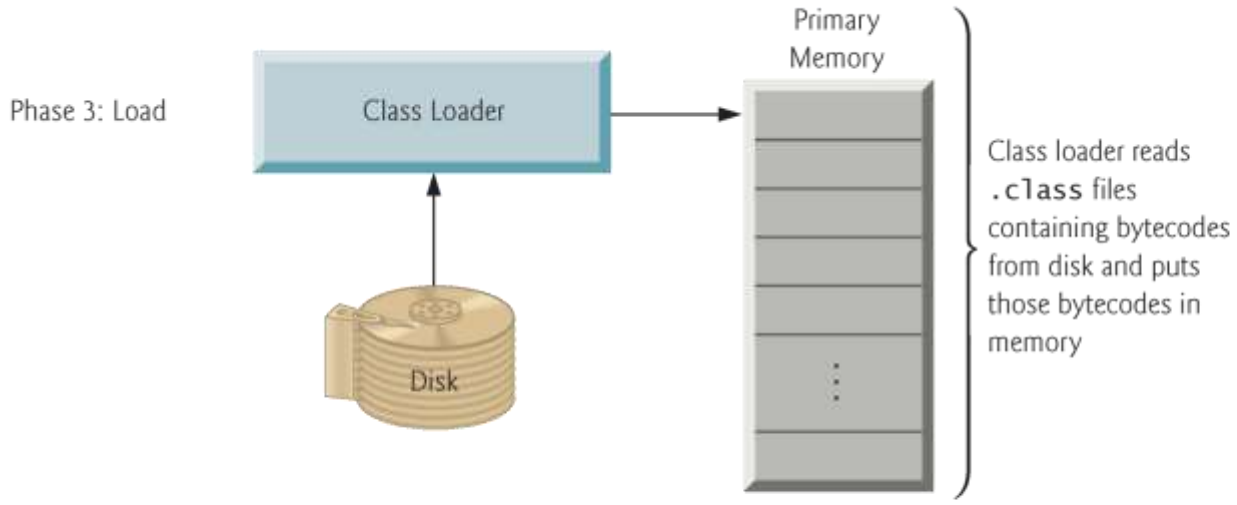


Fig. 1.8 | Typical Java development environment—loading phase.

1.9 Java and a Typical Java Development Environment (Cont.)

- Phase 4: Bytecode Verification
  - As the classes are loaded, the bytecode verifier examines their bytecodes
  - Ensures that they're valid and do not violate Java's security restrictions.
- Java enforces strong security to make sure that Java programs arriving over the network do not damage your files or your system (as computer viruses and worms might).

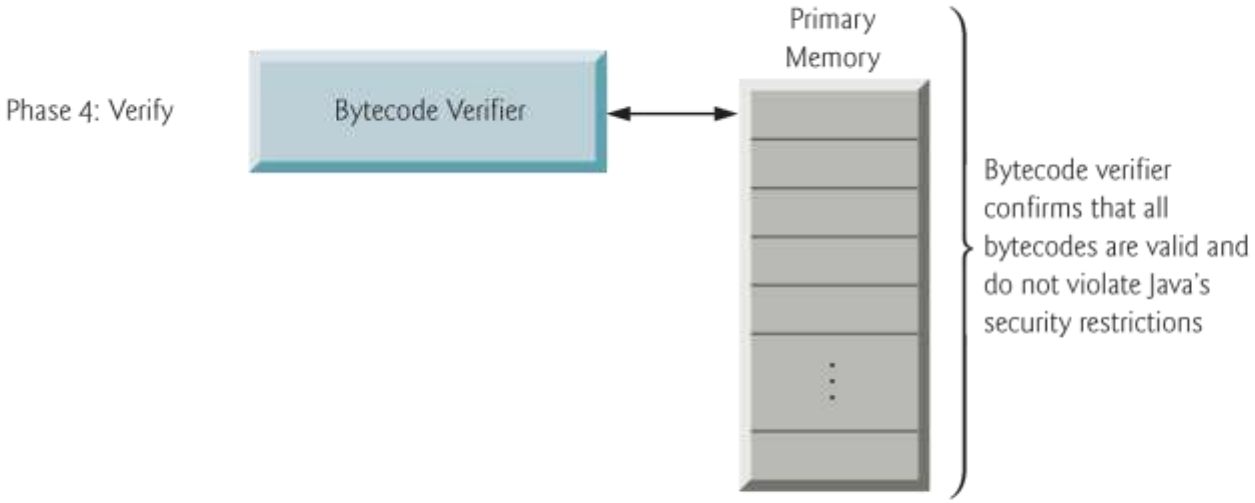
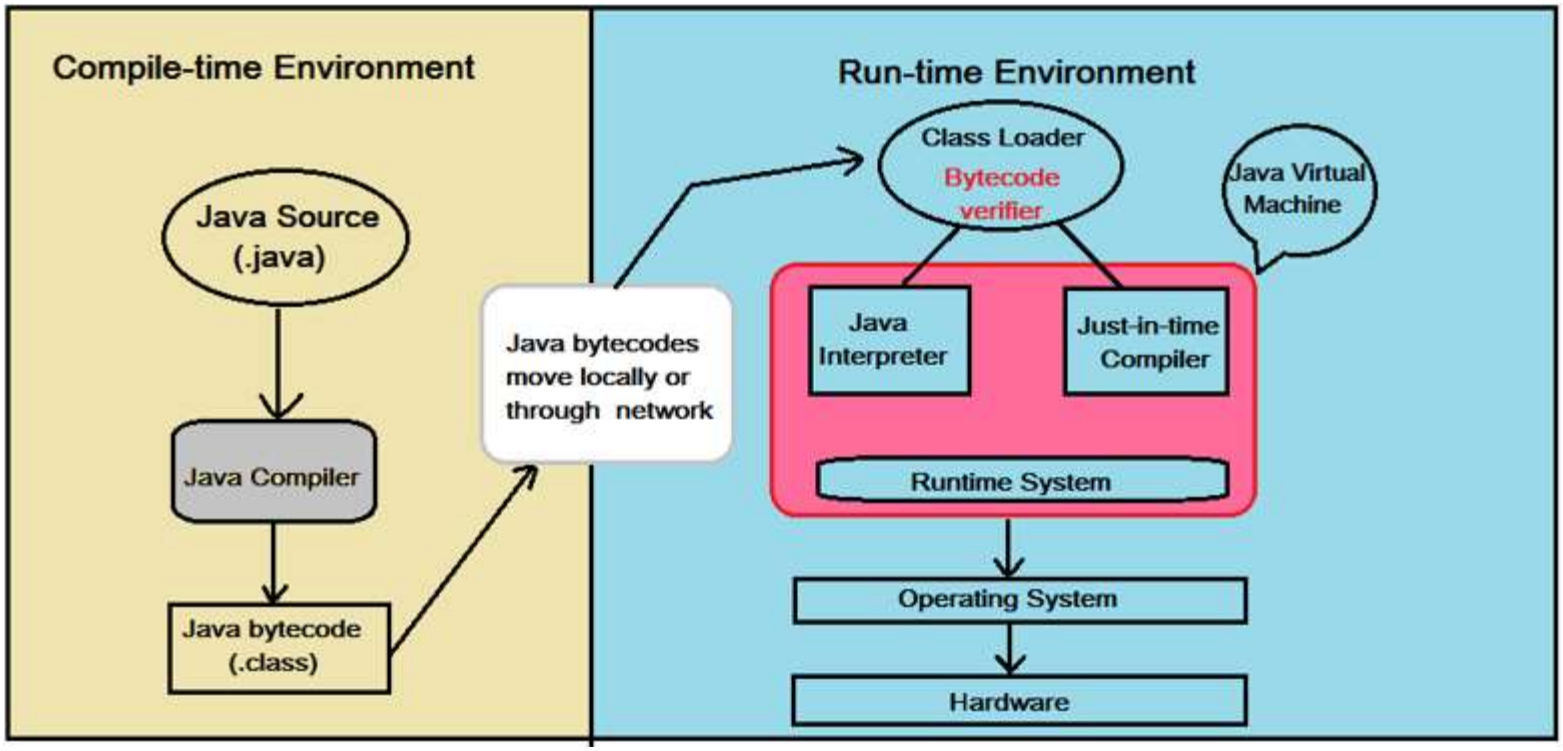


Fig. 1.9 | Typical Java development environment—verification phase.

1.9 Java and a Typical Java Development Environment (Cont.)

- Phase 5: Execution
  - The JVM executes the program's bytecodes.
  - JVMs typically execute bytecodes using a combination of interpretation and so-called just-in-time (JIT) compilation.
  - Analyzes the bytecodes as they're interpreted
  - A just-in-time (JIT) compiler—known as the Java HotSpot compiler—translates the bytecodes into the underlying computer's machine language.



1.9 Java and a Typical Java Development Environment (Cont.)

- When the JVM encounters these compiled parts again, the faster machine-language code executes.
- Java programs go through *two compilation phases*
- One in which source code is translated into bytecodes (for portability across JVMs on different computer platforms) and
- A second in which, during execution, the bytecodes are translated into machine language for the actual computer on which the program executes.

Java – Compiled and Interpreted language

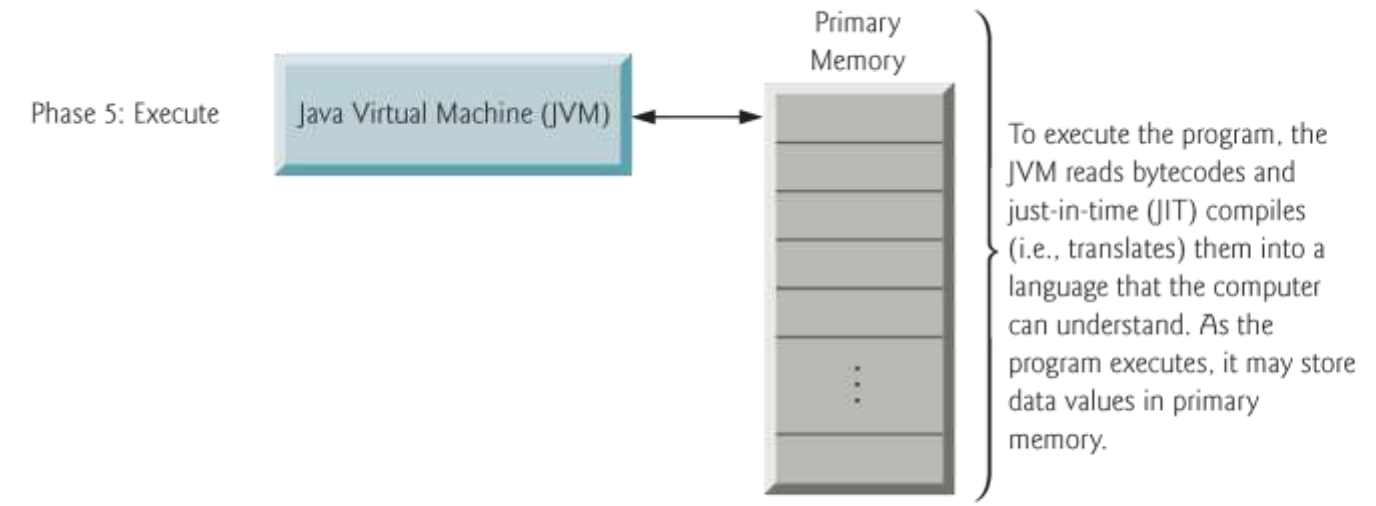


Fig. 1.10 | Typical Java development environment—execution phase.

In order to improve performance, **JIT compilers** interact with the Java Virtual Machine (JVM) at run time and compile suitable bytecode sequences into native machine code.

While using a JIT compiler, the hardware is able to execute the native code, as compared to having the JVM interpret the same sequence of bytecode repeatedly and incurring an overhead for the translation process.

Java Installation and Execution

- Demo
- https://www.oracle.com/java/technologies/javase-jdk16-downloads.html