Chapter 2
The Java Language

The Java language is a general-purpose programming language for platform-independent software development. As described in Java White Papers (Sun Microsystems, Inc.), "Java is simple, object-oriented, distributed, interpreted, robust, secure, architecture-neutral, portable, high-performance, multithreaded and dynamic."

Programmers for the Internet regard Java as a tool to create applets, which are mini-applications capable of running inside Web pages. After a Java applet is downloaded to a client browser, it may perform tasks without using resources from the server. Java is also a valuable programming language for distributed network environments.

For software developers who may not depend on networks, Java helps them produce bug-free code. It offers features like automated garbage collection, type-safe references, and multithreading to ease the task of developing robust, reliable, and complex software systems. The features make it easier to construct readable and manageable software.

Java has novel features such as exception handling, object reflection, and inner class to make programming simpler and easier. The objects created in a Java program can tell information about themselves. Control transfers in a Java program can be based on various types of information, including exceptions. The inner class feature of the Java language brings the structure of a Java program closer to a real-world application that the program models.

This chapter introduces the fundamental constructs of the Java language. It discusses some novel features of the language that are useful in data structure design and implementation. Specifically, we shall present

- the basic elements of Java programs, which include tokens, expressions, and statements;
- the control transfer mechanism of Java, which is supported by various types of statement and an exception handling mechanism;
- the object reflection model for querying objects; and
- the notion of inner class, which allows adapter and service classes to be embedded in classes that demand their functions.
2.1 Basic Elements of Java Program

2.1.1 White Space and Comment

2.1.1.1 White Space
A new line character, a return character, or a return character followed by a line feed character terminates a line in a Java program. The three types of line terminator are treated as white spaces. Other white spaces include the space character, horizontal tab, and form feed character.

Any number of white spaces can be inserted between Java program elements for improved readability. A statement or a block of statements may span several lines. The Java compiler ignores extra white spaces.

2.1.1.2 Comment
There are three types of comment in Java programs.

- A \textit{C-style comment} starts with symbol `/*` and ends with `*/`. It may span several lines.
- A \textit{C++-style comment} starts with symbol `//` and ends with a line terminator.
- A \textit{documentation comment} starts with symbol `/**` and ends with `*/`. As shown in Example 1.7, a documentation comment may span several lines.

The Java compiler ignores comments when it compiles source code. An automatic tool such as javadoc can collect documentation comments from the source code of classes into HTML documents.

2.1.2 Token

2.1.2.1 Token and Unicode
Syntactical symbols in a Java program that are not white spaces or comments are tokens. Specifically, a \textit{token} is a keyword, identifier, literal, separator, or operator. To satisfy international language programming, characters in comments, identifiers, string literals, and character constants are expressed in Unicode, which is an international standard character set. A Unicode character is 16 bits (2 bytes) long. Unicode character set is large enough for major international languages.

The first 128 Unicode characters represent ASCII characters. Elements other than comments, identifiers, string literals, and character constants in Java programs are formed with ASCII characters. Particularly, keywords and operator names are formed with ASCII characters.