

Java Programming

Binnur Kurt
binnur.kurt@ieee.org

Istanbul Technical University
Computer Engineering Department



Version 9.8.4

About the Lecturer



- ❑ BSc
İTÜ, Computer Engineering Department, 1995
- ❑ MSc
İTÜ, Computer Engineering Department, 1997
- ❑ Areas of Interest
 - Digital Image and Video Analysis and Processing
 - Real-Time Computer Vision Systems
 - Multimedia: Indexing and Retrieval
 - Software Engineering
 - OO Analysis and Design

Java Programming

2

Welcome to the Course

- Introduction
 - Name
 - Company affiliation
 - Title, function, and job responsibility
 - Programming experience
 - Reasons for enrolling in this course
 - Expectations for this course
- Course Hours
 - Wednesday, Computer Lab.
 - 18:30–21:30

Java Programming

3

Course Overview

1. Java Technology
2. Object-Oriented Programming
 - Encapsulation, Class, Method, Attribute, Accessing Object Members, Constructor
3. Identifiers, Keywords, and Types
 - Java Keywords, Primitive Types, Variables, Declarations, Assignment, Reference Type
 - Constructing and initializing Objects, Assigning Reference Types, Pass-by-Value
4. Expressions and Flow Control
 - Variable and Scope, Initializing Variables, Operators, Logical Operators, Branching Statement, Looping Statement, Special Loop Flow Control

Java Programming

4

5. Arrays
 - Declaring and Creating Arrays, Initialization of Arrays
 - Multidimensional Arrays, Resizing and Copying Arrays
6. Class Design
 - Inheritance, Access Control, Method Overriding, Super
 - Polymorphism, Virtual Method Invocation, instanceof
 - Casting Objects, Overloading Constructors, Object and Class Classes, Wrapper Classes
7. Advanced Class Features
8. When things go Wrong: Exceptions
9. Text-Based Applications
10. Building Java GUIs
11. GUI Event Handling
12. GUI-Based Applications
13. Threads
14. Advanced I/O Streams
15. Networking

Java Programming

5

1 Java Technology

6

Java Technology 1

Objectives

- ▶ Describe the key features of Java Technology
- ▶ Define the terms class and application
- ▶ Write, compile, and run a simple Java Technology application
- ▶ Describe the Java Virtual Machine's function
- ▶ Define Garbage collection
- ▶ List the three tasks performed by the Java platform that handle code security

Java Programming 7

Java Technology 1

What is Java Technology?

- ▶ A programming language
- ▶ A development environment
- ▶ An application environment
- ▶ A deployment environment

Java Programming 8

Java Technology 1

Primary Goals of the Java Technology


- ▶ Provides an easy-to-use language by
 - Avoiding the pitfalls of other languages
 - Being object-oriented
 - Enabling users to create streamlined and clear code
- ▶ Provides an interpreted environment for
 - Improved speed of development
 - Code portability

Java Programming 9

Java Technology 1

Primary Goals of the Java Technology

- ▶ Enables users to run more than one thread of activity,
- ▶ Load classes dynamically; that is, at the time they are actually needed,
- ▶ Supports dynamically changing programs during runtime by loading classes from disparate sources,
- ▶ Furnishes better security



Java Programming 10

Java Technology 1

Primary Goals of the Java Technology

- ▶ The following features fulfill these goals:
 - JVM,
 - Garbage Collection,
 - Code security

Java Programming 11

Java Technology 1

The Java Virtual Machine

- ▶ Provides hardware platform specifications,
- ▶ Reads compiled byte codes that are platform independent,
- ▶ Is implemented as software or hardware,
- ▶ Is implemented in a Java technology development tool or a Web browser.

Java Programming 12

Java Technology 1

- ## Java Programming

13

Java Technology 1

- Java Programming

14

Java Technology 1



15

Java Technology 1

Java Programming

16

Java Technology 1



17

Java Technology 1



18

Java Technology 1

A Java Class File (Cont'd)

```

ClassFile {
    u4 magic;
    u2 minor_version;
    u2 major_version;
    u2 constant_pool_count;
    cp_info constant_pool[constant_pool_count-1];
    u2 access_flags;
    u2 this_class;
    u2 super_class;
    u2 interfaces_count;
    u2 interfaces[interfaces_count];
    u2 fields_count;
    field_info fields[fields_count];
    u2 methods_count;
    method_info methods[methods_count];
    u2 attributes_count;
    attribute_info attributes[attributes_count];
}
        
```

19

Java Programming

Java Technology 1

Byte Code for Java Program

Location	Code	Mnemonic	Meaning
0x00e3	0x10	bipush	Push next byte onto stack
0x00e4	0x0f	15	Argument to bipush
0x00e5	0x3c	istore_1	Pop stack to local variable 1
0x00e6	0x10	bipush	Push next byte onto stack
0x00e7	0x09	9	Argument to bipush
0x00e8	0x3d	istore_2	Pop stack to local variable 2
0x00e9	0x03	iconst_0	Push 0 onto stack
0x00ea	0x3e	istore_3	Pop stack to local variable 3
0x00eb	0x1b	iload_1	Push local variable 1 onto stack
0x00ec	0x1c	iload_2	Push local variable 2 onto stack
0x00ed	0x60	iadd	Add top two stack elements
0x00ee	0x3e	istore_3	Pop stack to local variable 3
0x00ef	0xb1	return	Return

20

Java Programming

Java Technology 1

Garbage Collection

- ▶ Allocated memory that is no longer needed should be deallocated
- ▶ In other languages, deallocation is the programmer's responsibility
- ▶ The Java programming language provides a system-level thread to track memory allocation
- ▶ Garbage collection
 - Checks for and frees memory no longer needed
 - Is done automatically
 - Can vary dramatically across JVM implementations

21

Java Programming

Java Technology 1

Java Runtime Environment

The JRE performs as follows:

22

Java Programming

Java Technology 1

The Java Runtime Environment

- ▶ Performs three main tasks:
 - Loads code – Performed by the class loader
 - Verifies code – Performed by the bytecode verifier
 - Executes code – Performed by the runtime interpreter

23

Java Programming

Java Technology 1

The Class Loader

- ▶ Loads all classes necessary for the execution of a program
- ▶ Maintains classes of the local file system in separate "namespaces"
- ▶ Prevents spoofing

24

Java Programming

The Bytecode Verifier

► Ensures that

- The code adheres to the JVM specification
- The code does not violate system integrity
- The code causes no operand stack overflows or underflows
- The parameter types for all operational code are correct
- No illegal data conversions (the conversion of integers to pointers) have occurred

Java Technology 1
Java Programming

25

A Basic Java Application: TestGreeting.java

```
1 //  
2 // Sample "Hello World" application  
3 //  
4 public class TestGreeting{  
5     public static void main (String[] args) {  
6         Greeting hello = new Greeting();  
7         hello.greet();  
8     }  
9 }
```



Java Technology 1
Java Programming

26

Greeting.java

```
1 // The Greeting class declaration.  
2 public class Greeting {  
3     public void greet() {  
4         System.out.println("hi");  
5     }  
6 }
```

Java Technology 1
Java Programming

27

Running and Compiling

- Compiling TestGreeting.java
 - javac TestGreeting.java
- Greeting.java is compiled automatically
- Running an application
 - java TestGreeting
- Locating common compile and runtime errors

Java Technology 1
Java Programming

28

Compile-Time Errors

- javac: Command not found
- Greeting.java:4: cannot resolve symbol
symbol : method printl (java.lang.String)
location: class java.io.PrintStream
System.out.printl("hi");
- TestGreet.java:4: Public class TestGreeting
must be defined in a file called
"TestGreeting.java".

Java Technology 1
Java Programming

29

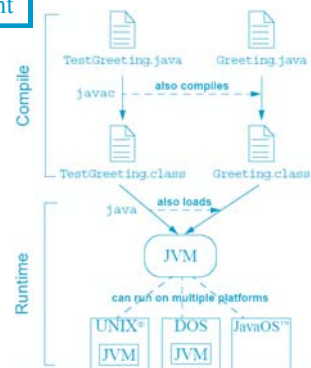
Run-Time Errors

- Can't find class TestGreeting
- Exception in thread "main"
java.lang.NoSuchMethodError: main

Java Technology 1
Java Programming

30

Java Runtime Environment



Java Programming

31

1# JAVA TECHNOLOGY

1. Compile Test1.java
2. Compile Test2.java
3. Compile Test3.java
4. Compile Test4.java



32