Introduction to Java Programming

"Overview and basic programming constructs"

Advanced Programming

Shakirullah Waseeb
shakir.waseeb@gmail.com

Nangarhar University

February 27, 2018

Agenda

Why so many languages?

Language evolution, innovation and development occurs for two fundamental reasons:

- To adapt to changing environments and uses
- To implement refinements and improvements in the art of programming

Agenda

Java introduction

- Java is conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank, and Mike Sheridan at Sun Microsystems, Inc. in 1991.
- Widely used programming language (handheld devices, network, computers)
- Java editions: Standard Edition (SE), Enterprise Edition (EE), Micro Edition (ME)

Java Buzzwords

- Java is:
  - Simple (inherit C and C++ syntax, adopted by C#)
  - Secure (confining java program to java execution environment)
  - Robust (auto memory management, error handling)
  - Portable (platform independent, byte code, JVM)
  - Object oriented (pure object oriented paradigm)
  - Multithreaded (do many things simultaneously)
  - Distributed (client/server programming, RMI)

Agenda

Review on Object Technology

- Demands for new and powerful software: where quickness, economy, and correctness remains an elusive goal
- Objects are instances (single occurrence) of *classes* which are essentially *reusable software components*

Examples

Date object, time object, audio object, video object, people object etc.

- Almost any noun can be reasonably represented as a software object in terms of attributes (e.g., name, color and size) and behaviors (e.g., calculating, moving and communicating).

Review on Object Technology – Continue

Bank account example

Account Class
Attributes : account_balance, date_opened, account_type
Functions : *inquireBalance, depositAmount, withdrawAmount*

- Instantiation: The process of creating objects of a class, object being created is refered to as instance of that class.
- Reusability: Reuse of existing classes when building new classes and programs save time and efforts, also helps in building reliable and effective systems; because they passed extensive *testing, debugging and performance* tuning

Review on Object Technology – Continue

- Messages and Methods calls: Sending message to an object; message is implemented as *method call*
- Encapsulation: wrapping of attributes and methods into objects
- Inheritance: creating new classes quickly and conveniently by *absorbing* the characteristics of an existing class, possibly *customizing* them and *adding* unique characteristics of its own

Review on Object Technology – Continue

- Creating best solution requires:
  - detailed *analysis* in order to
  - determine project's *requirements* (i.e defining *what* the system is supposed to do)
  - and developing a *design* (i.e deciding *how* the system should do it) that satisfies them
- Object-Oriented Analysis and Design (OOAD): analyzing and designing system from object-oriented point of view
- Single graphical language is used for communicating the results of any OOAD process, known as *Unified Modeling Language* (UML)
- UML: graphical language for modeling object-oriented systems

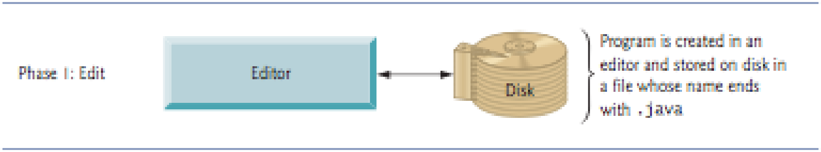
Creating and Executing Java application

- Java program creation and execution normally go through five phases – edit, compile, load, verify, and execute
- We discuss these phases in the context of the Java SE Development Kit (JDK) [1]

[1] `www.oracle.com/technetwork/java/javase/downloads/index.html`

Creating and Executing Java application - Phase 1

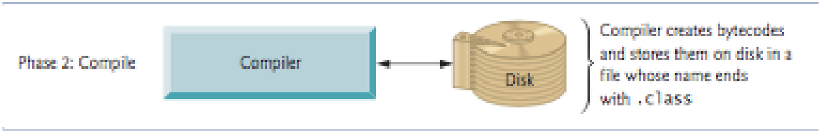
- Use an editor (vi, emacs on linux, notepad in windows)
- Type your java program typically referred as source code
- Save file with .java extension


Phase 1: Edit
Editor
Disk
Program is created in an editor and stored on disk in a file whose name ends with .java

[1]

Creating and Executing Java application - Phase 2

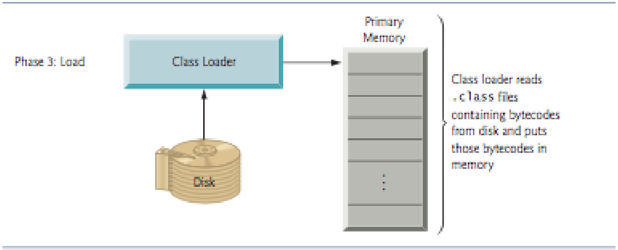
- Use command javac (java compiler) to compile the source program
- For example a program called Sallam.java we use following command javac Sallam.java
- If program compiles successfully it will produce Sallam.class file which is the compiled version of program and called bytecode


Phase 2: Compile
Compiler
Disk
Compiler creates bytecodes and stores them on disk in a file whose name ends with .class

[1]

Loading a Program into Memory - Phase 3

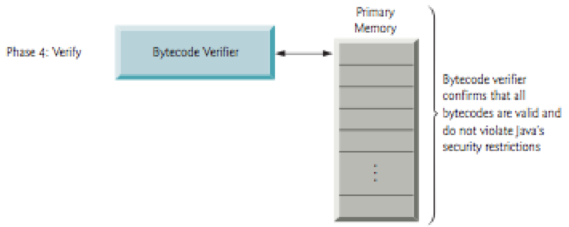
- JVM places program in memory to execute it – known as loading
- JVM's class loader takes the .class files into memory (also .class files that the program uses)
- These .class files can be loaded from hard disk or from network


Phase 3: Load
Class Loader
Disk
Primary Memory
Class loader reads .class files containing bytecodes from disk and puts those bytecodes in memory

[1]

Bytecode Verification - Phase 4

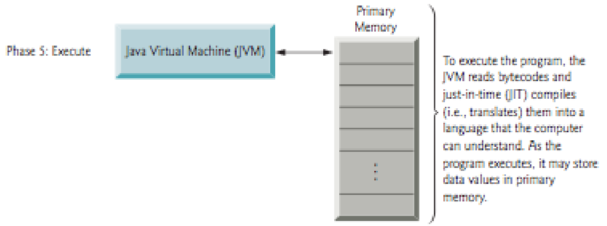
- Bytecode Verifier examines their bytecodes to ensure that they're valid and do not violate Java's security restrictions
- Java enforces strong security


Phase 4: Verify
Bytecode Verifier
Primary Memory
Bytecode verifier confirms that all bytecodes are valid and do not violate Java's security restrictions

[1]

Execution - Phase 5

- JVM executes the bytecode
- Performing specified actions in the program
- These .class files can be loaded from hard disk or from network


Phase 5: Execute
Java Virtual Machine (JVM)
Primary Memory
To execute the program, the JVM reads bytecodes and just-in-time (JIT) compiles (i.e., translates) them into a language that the computer can understand. As the program executes, it may store data values in primary memory.

[1]

Agenda

Simple Java Program

A Simple Program Example

```
class MyClass {
                public static void main (String args[]){
                       System.out.println("Assalam-o-Alikum");
                       }
                }
```

- Save it as MyClass.java
- Compile it as: javac MyClass.java
- Execute it as: java MyClass

Simple Java Program for integer input

A simple example program; adding two numbers

```
class Adder {
        public static void main (String args[]){
            short num1;
            short num2;
            int sum;
            Scanner scan = new Scanner(System.in);
            System.out.println("Enter first number: ");
            num1= scan.nextShort();
            System.out.println("Enter second number: ");
            num2= scan.nextShort();
            sum= num1 + num2;
            System.out.printf(" %d + %d = %d", num1, num2, sum);
            }
        }
```

Agenda

Data Types

- Java is strongly typed language
  Note: every variable has a type, every expression has a type, and every type is strictly defined
  All assignments, whether explicit or via parameters passing in methods call are checked for type compatibility
- There are no automatic coercions or conversions of conflicting types as in some languages.
- Simple Types
  - Integers: **byte, short, int, long**
  - Floating point: **float, double**
  - Characters: **char**
  - Boolean: **boolean**

Variables

- Variables are defined via a combination of type, identifier, and an optional initializer
  type identifier [= value] [, identifier [= value] ...] ;
- Dynamic initialization
- Scope and life time of variables; two general catagories global and local, however java define *class* and *method* scope
- Type conversion and type casting
  - Automatic conversion: takes place if; two types are compatible, destination type is larger than source type
  - Casting incompatible type: a cast is used to make a conversion between incompatible types: *narrowing conversion* (e.g casting a large value type into small value type ***int*** *to* ***byte***) *truncation* (e.g converting float type into integer type)
    (target-type) value
- Automatic type promotion and promotion rules

Arrays

- An array is a group of like-typed variables that are referred to by a common name.
- One dimensional array: The general form of a one dimensional array declaration is
  type array-var[ ]; no memory will set aside
  array-var = new type[size]; memory of given size will be reserved
  array-var[0] = value1;
  array-var[1] = value2;
  array-var[ ] = {value1, value2};

Arrays –continue

- Two dimensional array: The general form of declaration is
  type array-var[ ][ ];
  array-var = new type[row-size][column-size];
  array-var[0][0] = value1;
  array-var[0][1] = value2;
  array-var[1][0] = value3;
  array-var[1][1] = value4;
  array-var[ ][ ] = {
  {value1, value2},
  {value3, value4}
  };

Your Turn: Time to hear from you!

[2]

[2] https://fensafitters.files.wordpress.com/2013/07/3d095.jpg

References

- P.J. Deitel, H.M. Deitel
  Java How to program, 9th Edition .
  Prentice Hall, 2012.
- Herbert Schildt
  The complete reference Java2, 5th Edition .
  McGraw-Hill/Osborne, 2002.