CS 213 – Programming Language 2

Dr. Ahmed Hesham Mostafa

Lecture 0 – Course Introduction & Plan

Course Materials & Announcements

- All course material (lecture notes “slides", announcements, assignments, any supplemental notes or documentation), will be made available “posted” online on weekly basis, on Microsoft Teams:
- Access code:

xk5lm08

Course Aims and Objectives

- Learn the basic programming with java (Java Architecture, Tokens, Data types, Operators, Conditional and Looping statements)
- Understand the Object-oriented programming with java using object-oriented concepts
- Learn the advanced java programming using exception handling, multithreading, file handling, and graphical applications
- Lean basic concepts about Arrays and methods in java
- Classes and Objects - Creating Classes and Objects, Constructors, Method overloading, "this" Reference, Garbage Collection
- Inheritance - Implementing Inheritance, Method overriding, "final" Keyword, Abstract Classes and Methods, Dynamic Binding
- Leann the Exception Handling
- Learn how to use Java collections
- GUI Applications with Java Swing
- Learn basic Network programming with java
- Learn basic multithreading concepts

Textbooks

- Paul Deitel, Harvey Deitel, Java: How to Program, , Early Objects, 11th Edition.
- Cay S. Horstmann, **Big Java: Late Objects**
- Introduction to Java Programming and Data Structures, Comprehensive Version 12th Edition, by Y. Liang (Author), Y. Daniel Liang
- Java documentation https://docs.oracle.com/javase

Topic covered

Java basics	Data Types, Methods, 1D array and 2D array, Scanner, Methods, Operators, If statement, loops.
OOP	Classes, objects, Inheritance, Encapsulation, Abstraction, Polymorphism
Files	Handling IO Text and Binary files
Exceptions	Handling Exceptions
JavaSwing	Graphical User Interface Elements based JAvaSwing
Event Driven programming	Mouse Listeners, Actions, Window listener
Collections	ArrayList, List, Vecto , HashSet ,…..etc
DB	JDBC - Java Database Connectivity
Threads	Threads and shared objects
Network Programming	Sockets Programming

Grade Policy

- Final 50
- Midterm 20
- Practical Quizzes 20
- Project 15
- Note total semester work grade is 50
- If the student gets a mark above 50, he will only get 50.

Section

- Get involved with the story.

Practice.. Practice.. Practice..

Course Project Policy

01

Project groups: Each group will be 4 to 5 students.

02

Project Must be implemented based om MVC concept

03

GUI can be implemented using Swing or Javafx[Bounce]

Academic Integrity

You can discuss ideas and methodology for the homework (assignments / sheets) with other students in the course, but you must write your solutions completely independently.

We will be code-checking to assess similar submissions or submissions that use code from other sources.

Java Setting up

- Integrated Development Environment (IDE): IDE provides a sophisticated GUI and editor, with integrated facilities for compiling, running, and debugging programs
- Java 2 Standard Edition SDK (Software Development Kit): //What we need..
  - SDK includes everything you need to compile and run Java.
  - SDK tools are used for compiling and running Java programs on a variety of platforms
  - The Java compiler (**javac**) translates Java source into bytecode.
  - The Java interpreter (**java**) translates and executes Java bytecode. Java interpreters are available for many different computer systems.
- Java 2 Enterprise Edition (J2EE) → for large-scale systems
- Java 2 Micro Edition (J2ME) → for mobile phones
- Java 2 Standard Edition (J2SE) – Our Course -

Java Setting up

- We can work with Netbeans/Eclipse/ IntelliJ
  - https://netbeans.org/downloads/
  - https://eclipse.org/downloads/
  - JDK http://www.oracle.com/technetwork/java/javase/downloads/index.html
  - Java documentation https://docs.oracle.com/javase

Java Setting up For JavaFX

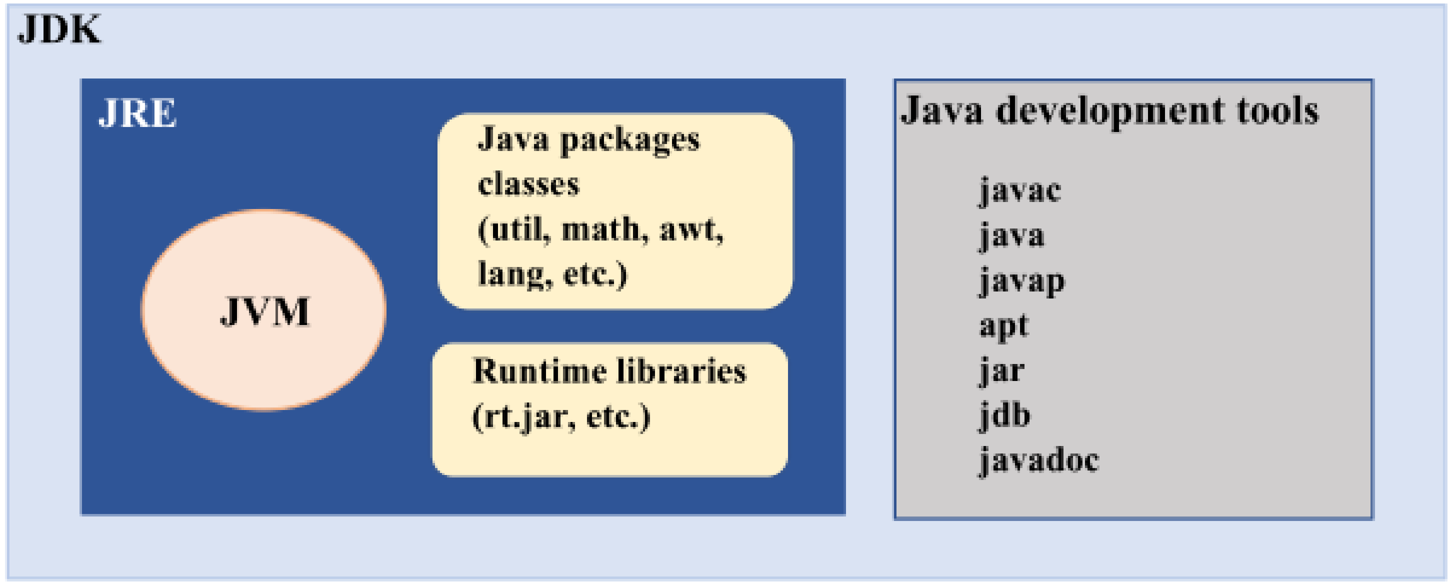
- Install first the JDK and We will use Zulu JDK-fx
- Java Download | Java 7, Java 8, Java 11, Java 13, Java 15, Java 17, Java 19 - Linux, Windows & macOS (azul.com)
- IntelliJ IDEA Community Edition
- Download IntelliJ IDEA: The Capable & Ergonomic Java IDE by JetBrains
- We will use Scene Builder for GUI (Drag & Drop)
- Scene Builder - Gluon (gluonhq.com)

Video for Configuration

- https://youtu.be/LTgClBqDank

What Is JDK?

- JDK, or **Java Development Kit**, consists of tools that we use JDK to develop and run Java code. Before developing and running Java code, you should install it on your computer or system.
- The picture represents the structure of JDK


JDK
JRE
JVM
Java packages classes (util, math, awt, lang, etc.)
Runtime libraries (rt.jar, etc.)
Java development tools
javac
java
javap
apt
jar
jdb
javadoc

What Is JDK?

- The **Java Development Kit (JDK)** is a software development environment used for developing Java applications and applets.
- It includes the Java Runtime Environment (JRE), an interpreter/loader (Java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc), and other tools needed in Java development.

What Is JDK?

As you see, **JDK** consists of **JRE** and **Java development tools.**
JRE or ***Java Runtime Environment is*** a package that provides an environment to **only run (**not develop) the Java program(or application)on your machine. It is only used to run Java programs.

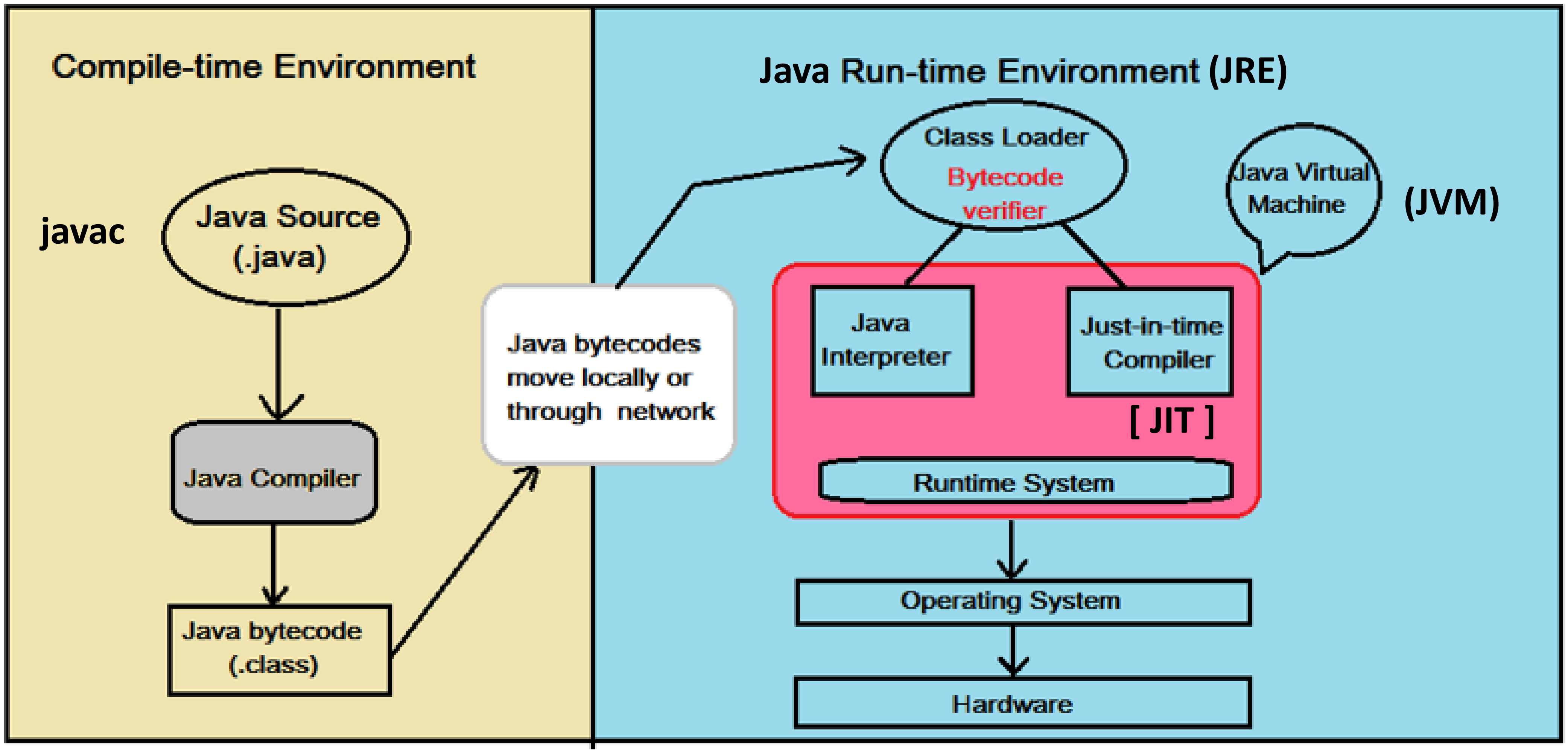
JDT or **Java Development tools** consist of many tools like compilers, debuggers, and other development tools.

The most important part of JDK and JRE is **JVM** (**Java Virtual Machine)** and its responsibility is the execution of code line-by-line. It's also known as an Interpreter. In the following, we will get more acquainted with it.

Java Virtual Machine (JVM)

- The Java Virtual Machine is a specification that provides a runtime environment in which java bytecode can be executed. It means JVM creates a platform to run Java bytecode(.class file) and converting into different languages (native machine language) which the computer hardware can understand. Actually, there is nothing to install as JVM. When the JRE is installed, it will deploy the code to create a JVM for the particular platform. JVMs are available for many hardware and software platforms.

How does Java Code Compile And Run?



How does Java Code Compile And Run?

- **Step 1:** Writing the source and save it with extinction **.java**
- **Step 2:** compile the code in the command prompt by using the command line or atomically with IDE javac filename.java).
- This invokes the Java Compiler. The compiler checks the code for syntax errors and any other compile time errors and if no error is found the compiler converts the java code into an intermediate code(**filename. class** file) known as **bytecode.**
- This **intermediate code is platform-independent** (you can take this bytecode from a machine running windows and use it in any other machine running Linux or macOS etc).
- Also, this bytecode is an intermediate code, hence it is only understandable by the **JVM (Java Virtual Machine)** and not the user or even the hardware /OS layer.

How does Java Code Compile And Run?

- **Step 3:** This is the start of the Run Time phase, where the bytecode is loaded into the JVM by the **class loader**(another inbuilt program inside the JVM).
- **Step 4:** Now the **bytecode verifier** checks the bytecode for its integrity and if no issues are found passes it to the interpreter. For example, if in the program, we use a variable that has not been declared, or if the run-time stack overflows, it will throw an Exception and the compiling process will stop.
- **Step 5:** Since java is both compiled and interpreted language, now the java **interpreter and Just in time Compiler (JIT)** inside the JVM convert the bytecode into executable machine code and passed it to the OS/Hardware i.e. the CPU to execute.

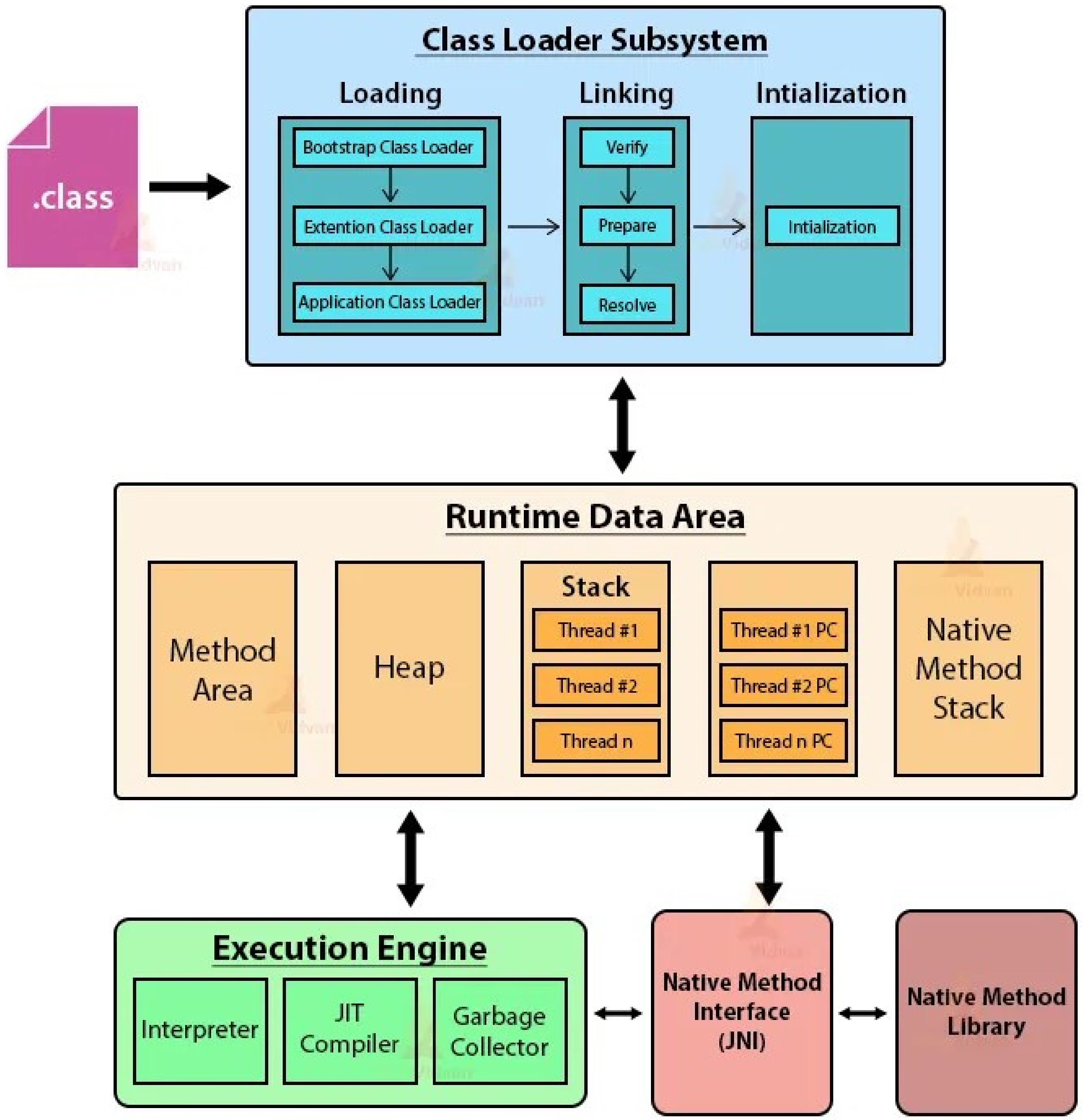
Just in time Compiler (JIT)

- The JIT compiler allows Java to be both a compiled and interpreted language. It combines the performance advantage of a compiled language with the flexibility of an interpreted language.
- While the JVM interprets bytecode, the JIT analyses execution and dynamically compiles frequently executed bytecode to machine code. This prevents the JVM from having to interpret the same bytecode over and over.

How the JIT Compiler Works in Java

- The source code you write (.java) is compiled by javac to bytecode. When your program executes, the JVM starts interpreting this bytecode.
- The interpreter translates the bytecode into machine code and feeds it to the CPU for processing.
- While this happens, the JIT profiles and analyses the code execution. It tracks the most frequently called methods and dynamically compiles these to machine code at runtime.

What is a JVM in Java ?



What is a JVM in Java?

- There are three main mechanisms inside the JVM as shown in the above diagram.
- ClassLoader
- Memory Area
- Execution Engine

Next Lecture

References

- Java: How To Program, Early Objects, 11th edition
- Setup IntelliJ IDEA (2021) for JavaFX & SceneBuilder and Create Your First JavaFX Application – YouTube
- Introduction to Java Programming and Data Structures, Comprehensive Version 12th Edition, by Y. Liang (Author), Y. Daniel Liang
- Java documentation https://docs.oracle.com/javase
- https://www.youtube.com/watch?v=LTgClBqDank&feature=youtu.be
- Understanding Java Virtual Machine (JVM) Architecture | by Jalitha Dewapura | Java For Beginners | Medium

Thanks