Object Oriented Programming Using Java 1.8 (Spider)

Evaluation Strategy

- **Duration** : **90** Hours(46 Theory + 44 Lab)
- **Evaluation** : Total 100 marks
- **Theory Exam** : 40% (CCEE)
- **Lab Exam** : 40% (End-Module Exam)
- **Internals** : 20%
- Reference Books :
  1. Java, The Complete Reference – Herbert Schildt
  2. Core Java Volume I and II – Cay S. Horstmann
  3. Java 8 Programming Black Book – D.T. Editorial Services
  4. Object Oriented Analysis and Design with Applications – Grady Booch
- Reference links :
  1. https://docs.oracle.com/javase/tutorial/
  2. https://dev.java/
  3. https://docs.oracle.com/javase/8/docs/api/

Agenda-1

- Language, Technology and Platform
- History of Java programming language
- Java version history
- Java Software Development Kit
- Simple ”Hello World!!” application
- Exploring bytecode using javap
- Java comments
- Java entry point
- Primitive & non primitive types,
- Wrapper classes
- Command line arguments

Classification of Programming Languages

- **Machine level programming language**
  - Binary language
- **Low level programming language**
  - Assembly programming language
- **High level programming language**
  - Procedure oriented programming languages
    - Algol, Fortran, COBOL, Pascal, C etc.
  - Object oriented programming languages
    - Simula, Smalltalk, C++, Java, C#, Go, Python etc.
  - Object based programming languages
    - Ada, Java Script, Action Script, VB Script, Visual Basic etc.
  - Functional programming languages
    - Haskell, Lisp, F#, Scala, Clojure, Ocaml, Python, C#, C++, Java

“Hello World” Application using Machine Language

- Here is the “Hello World!!” program in binary:

  01001000 01100101 01101100 01101100

  01101111 00100000 01010111 01101111

  01110010 01101100 01100100 00100001

  00100001

- Advantages of machine-level language:
  1. Very fast execution times
  2. Complete control over the computer's hardware
  3. No need for a compiler or interpreter

- Limitations of machine-level language
  1. Very difficult to read and write
  2. Specific to a particular computer architecture, making it non-portable
  3. Requires a deep understanding of the underlying hardware

“Hello World” Application using Low Level Language

- Here is the "Hello World!!" program in x86 assembly language:

```
section .data
    msg db 'Hello World!!',0

section .text
    global _start

_start:
    ; Write the string to stdout
    mov eax, 4      ; System call for "write"
    mov ebx, 1      ; File descriptor for stdout
    mov ecx, msg    ; Pointer to the message
    mov edx, 13     ; Length of the message
    int 0x80        ; Call the kernel

    ; Exit the program with a status of 0
    mov eax, 1      ; System call for "exit"
    xor ebx, ebx    ; Status code (0 = success)
    int 0x80        ; Call the kernel
```

Advantages and Limitations of Low Level Language

- Advantages of low-level languages include:
  1. Faster execution times than high-level languages
  2. Direct access to the computer's hardware
  3. More portable than machine-level language

- Limitations of low-level languages include:
  1. Still difficult to read and write compared to high-level languages
  2. Can be error-prone due to the need for manual memory management
  3. Not as user-friendly as high-level languages

“Hello World” Application using High Level Language

- Here is the “Hello World!!” program in Python:

```
print("Hello, world!")
```

- Advantages of high-level language:
  1. Easy to read and write
  2. Less error-prone due to built-in error checking and automatic memory management
  3. More portable than machine-level and low-level languages

- Limitations of high-level language
  1. Slower execution times than low-level languages
  2. Less control over the computer's hardware than low-level languages
  3. More memory-intensive than low-level languages

What is Language?

- Language is a system of communication that allows humans to convey meaning to each other through the use of symbols, such as words, gestures, and facial expressions.
- In the context of computer science, programming languages are used to write software programs that can be executed by computers.
- In General programming languages contains:
  - Syntax and Semantics
  - Tokens(Identifier, Keyword, Constant, operator, Separator)
  - Data Types
  - Comments
  - Built in unique features
- Even though languages are used to implement business logic, they can be used to create applications too:
  - Console User Interface(CUI) / Command Line Interface(CLI) Application
  - Graphical User Interface(GUI) application
  - Library Application(.lib/.dll/.jar)
- Example of Languages:
  - C, C++, Java, C#, Python, GO etc.

What is Technology?

- Technology refers to the tools and techniques used to create and manipulate information. In Simple words, we can develop application using technology.
- Example:
  - Java
  - Artificial Intelligence (AI)
  - Blockchain
  - Augmented Reality (AR)
  - Internet of Things (IoT)
  - Cloud Computing
- Note: Every language can be considered as technology but every technology can not be considered as language.

What is Platform?

- A *platform* is the hardware or software environment in which a program runs.
- Most platforms can be described as a combination of the operating system and underlying hardware:
  - Microsoft Windows
  - Linux
  - Mac OS
  - Android
- Software-only platforms are the applications that runs on top of other hardware-based platforms:
  - Java
  - Microsoft .NET

History of Java Programming Language

- Java is high-level, statically type checked, Object-oriented as well as Functional programming language.
- Java language is both technology as well as platform.
- Code name of the Java was “Green” Project.
- Java was developed by (“**Green Team**”) James Gosling, Patrick Naughton, Chris Warth, Ed Frank, and Mike Sheridan at Sun Microsystems, which was later acquired by Oracle Corporation(2010).
- The name "Green" was chosen because the team wanted to emphasize that their goal was to create an environmentally-friendly language that would allow programs to run efficiently on a variety of devices, including small handheld devices and large servers.
- Green Team started working on “Green Project” in 1991 and first public version released in 1996.
- In fact, the original name for Java was "**Oak**", but it was changed to "Java" because another company had already registered the name "Oak" for their product.
- Java's slogan is "**Write Once, Run Anywhere**" (sometimes abbreviated as "WORA").
- The standardization of the Java language is overseen by the **Java Community Process** (JCP).
- “*7” is the first project delivered by Green Team.
  - An extremely intelligent TV remote control.
- Reference to read a short history of Java: Core Java Volume I

Java Version History

(https://en.wikipedia.org/wiki/Java_version_history)

Version	class file format version[8]	Release date	End of Free Public Updates[9][10][11][12][13][14][15]	Extended Support Until
JDK 1.0	45	23rd January 1996	May 1996	–
JDK 1.1	45.3	2nd February 1997	October 2002	?
J2SE 1.2	46	4th December 1998	September 2003	?
J2SE 1.3	47	8th May 2000	October 2010	?
J2SE 1.4	48	13th February 2002	October 2008	February 2013
Java SE 5	49	29th September 2004	November 2009	April 2015
Java SE 6	50	11th December 2006	April 2013	December 2018 for Oracle[9] December 2026 for Azul[12]
Java SE 7	51	28th July 2011	September 2022 for OpenJDK Maintained by Oracle until May 2015[16], Red Hat until August 2020[17] and Azul until September 2022[18]	July 2022 for Oracle[9] June 2020 for Red Hat[13] December 2027 for Azul[12]
Java SE 8 (LTS)	52	18th March 2014	(OpenJDK currently maintained by Red Hat)[19] March 2022 for Oracle (commercial) December 2030 for Oracle (non-commercial) December 2030 for Azul[12] May 2026 for IBM Semeru[14] At least May 2026 for Eclipse Adoptium[10] At least May 2026 for Amazon Corretto[11]	December 2030 for Oracle[9] November 2026 for Red Hat[13]
Java SE 9	53	21st September 2017	March 2018 for OpenJDK	–
Java SE 10	54	20th March 2018	September 2018 for OpenJDK	–
Java SE 11 (LTS)	55	25th September 2018	(OpenJDK currently maintained by Red Hat)[20] September 2026 for Azul[12] October 2024 for IBM Semeru[14] At least October 2024 for Eclipse Adoptium[10] At least September 2027 for Amazon Corretto[11] At least October 2024 for Microsoft[21][15]	September 2026 for Oracle[9] September 2026 for Azul[12] October 2024 for Red Hat[13]
Java SE 12	56	19th March 2019	September 2019 for OpenJDK	–
Java SE 13	57	17th September 2019	(OpenJDK currently maintained by Azul)[22] March 2023 for Azul[12]	–
Java SE 14	58	17th March 2020	September 2020 for OpenJDK	–
Java SE 15	59	16th September 2020	(OpenJDK currently maintained by Azul)[23] March 2023 for Azul[12]	–
Java SE 16	60	16th March 2021	September 2021 for OpenJDK	–
Java SE 17 (LTS)	61	14th September 2021	(OpenJDK currently maintained by SAP)[24] September 2029 for Azul[12] October 2027 for IBM Semeru[14] At least September 2027 for Microsoft[15] At least September 2027 for Eclipse Adoptium [10]	September 2029 or later for Oracle[9] September 2029 for Azul[12] October 2027 for Red Hat[13]
Java SE 18	62	22nd March 2022	September 2022 for OpenJDK and Adoptium	–
Java SE 19	63	20th September 2022	March 2023 for OpenJDK	–
Java SE 20	64	21st March 2023	September 2023 for OpenJDK	–
Java SE 21 (LTS)	65	September 2023	September 2028	September 2031 for Oracle[9]

Legend: Old version | Older version, still maintained | **Latest version** | Future release

Java Code Names

- Code names are used by developers and companies to refer to different versions of Java during the development process.
  1. JDK 1.1: Sparkler
  2. J2SE 1.2: Playground
  3. J2SE 1.3: Kestrel
  4. J2SE 1.4: Merlin
  5. J2SE 5.0: Tiger
  6. Java SE 6: Mustang
  7. Java SE 7: Dolphin
  8. **Java SE 8: Spider**
  9. Java SE 9: Project Jigsaw
  10. Java SE 10: Project Valhalla
  11. Java SE 11: Project Amber
  12. Java SE 12: Project Skara
  13. Java SE 13: Project ZGC
  14. Java SE 14: Project Panama
  15. Java SE 15: Project Loom
  16. Java SE 16: Project Metropolis
  17. Java SE 17: Project Panama 2.0

JLS and JSR

- JLS stands for Java Language Specification.
- It is a document that defines the syntax, semantics, and behavior of the Java programming language.
- Reference: https://docs.oracle.com/javase/specs/
- JSR stands for Java Specification Request.
- It is a formal proposal for adding new features or making changes to the Java platform.
- JSRs are submitted by individuals, organizations, or companies to the Java Community Process (JCP) for review and approval.
- Reference: https://jcp.org/en/jsr/detail?id=336

Editions of Java Platform

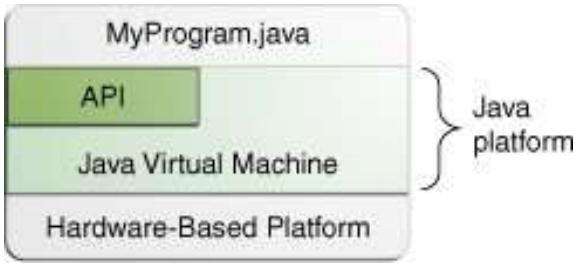
1. **Java SE**
   - Java Standard Edition(also called as Core Java)
   - It is designed for developing and running desktop, server, and standalone applications.
   - It includes the core Java API and provides a foundation for building Java applications.
2. **Java EE**
   - Java Enterprise Edition(also called as Web / Advanced / Enterprise Java)
   - It is designed for building web applications and web services.
   - It includes the Java Servlet API, Java Server Pages (JSP), and Enterprise JavaBeans (EJB).
3. **Java ME**
   - Java Micro Edition.
   - It is designed for developing applications for resource-constrained devices such as mobile phones, PDAs, and other embedded systems.
   - It includes a subset of the Java SE APIs and provides a smaller footprint than Java SE.
4. **Java Card**
   - It is designed for building smart card applications.
   - It includes a subset of the Java ME APIs and provides security and cryptographic features for smart card applications.

Software Development Kit

- **SDK = Development Tools + Documentation + Supporting Libraries + Execution Environment.**
- **JDK = Java Development Tools + Java Docs + rt.jar + Java Virtual Machine.**
  - Java Development Tools: javac, java, javap, javadoc, etc.
  - Java Docs: HTML pages, which contains help of Core Java API
  - rt.jar: It contains core Java API
  - Java Virtual Machine: It is abstract computer which manages execution of bytecode.
- JDK is a platform dependent software that we can download from oracle site.
- Developer must install JDK to develop Java application.
- **JRE = rt.jar + Java Virtual Machine.**
- JRE is platform dependent software that we can download from oracle site.
- To run Java application on Client's machine we must install/deploy JRE.
- **JVM = Java Virtual Machine**
- The JVM is responsible for interpreting the bytecode and translating it into machine code that can be executed by the underlying operating system. It also provides memory management, security, and other runtime services necessary for executing Java applications.
- Reference: https://www.oracle.com/in/java/technologies/downloads/

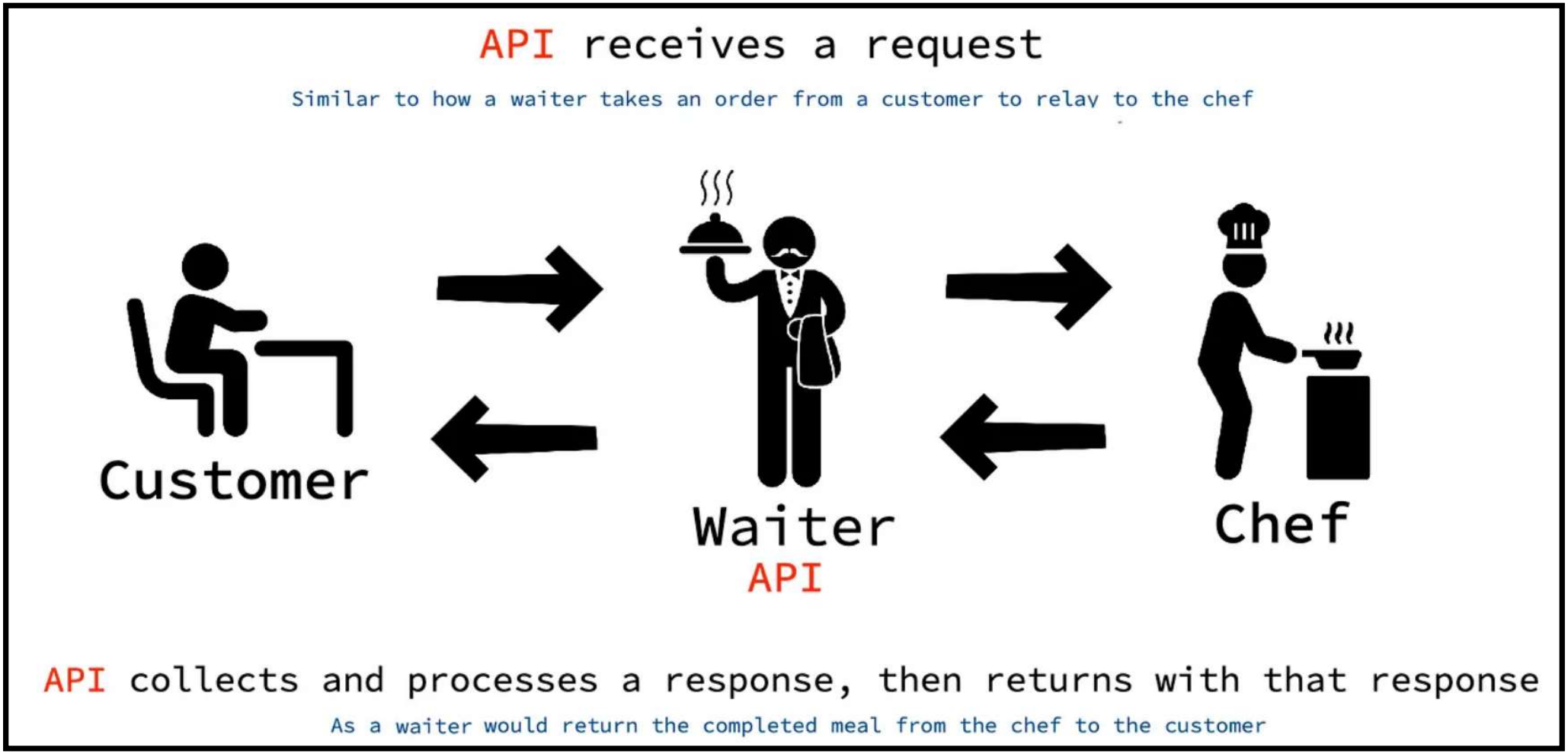
Components of Java Platform

- The Java platform has two components:
  1. The *Java Virtual Machine*
  2. The *Java Application Programming Interface* (API)



- Reference: https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html

Application Programming Interface



Agenda-2

- - main method
- - meaning of "System.out.println"
- - print, println and printf
- - Java Comments
- - Primitve & Non Primitive types
- - wrapper class
- - Initialization and Assignment
- - Narrowing & Widening
- - Boxing & Unboxing
- - NumberFormatException
- - Commandline arguments
- - Assignment to do:
- - Assignment 1

Packages in Java

Packages can contains:

1. Sub package
2. Interface
   - Nested Type
   - Constant Fields
   - Abstract Methods
   - Default Methods
   - Static Interface Methods
3. Class
   Nested Types(Interface /Class /Enum)
   Fields
   Constructor
   Method
4. Enum
5. Exception
6. Error
7. Annotation Type

Modifiers in Java

- PUBLIC
- PRIVATE
- PROTECTED
- STATIC
- FINAL
- SYNCHRONIZED
- VOLATILE
- TRANSIENT
- NATIVE
- INTERFACE
- ABSTRACT
- STRICT

Reference: https://docs.oracle.com/javase/8/docs/api/java/lang/reflect/Modifier.html

Main Method & Modifiers

- Access Modifiers: Modifiers which are used to control visibility of members of the class is called access modifier.
  - private
  - Package level private(also called as default)
  - protected
  - Public
- In java, main method is considered as entry point method. With the help of main thread, JVM invoke main method.
  - Syntax:
  - public static void main(String args[])
  - public static void main(String[] args)
  - public static void main(String... args)
- We can define main method inside class as well as interface.

Reference: https://docs.oracle.com/javase/tutorial/getStarted/application/index.html