Object Oriented Programming with Java (Subject Code: BCS-403)

Unit 1

Lecture 2

Lecture 2

- JVM, JRE, Java Environment
- Java Source File Structure
- Compilation

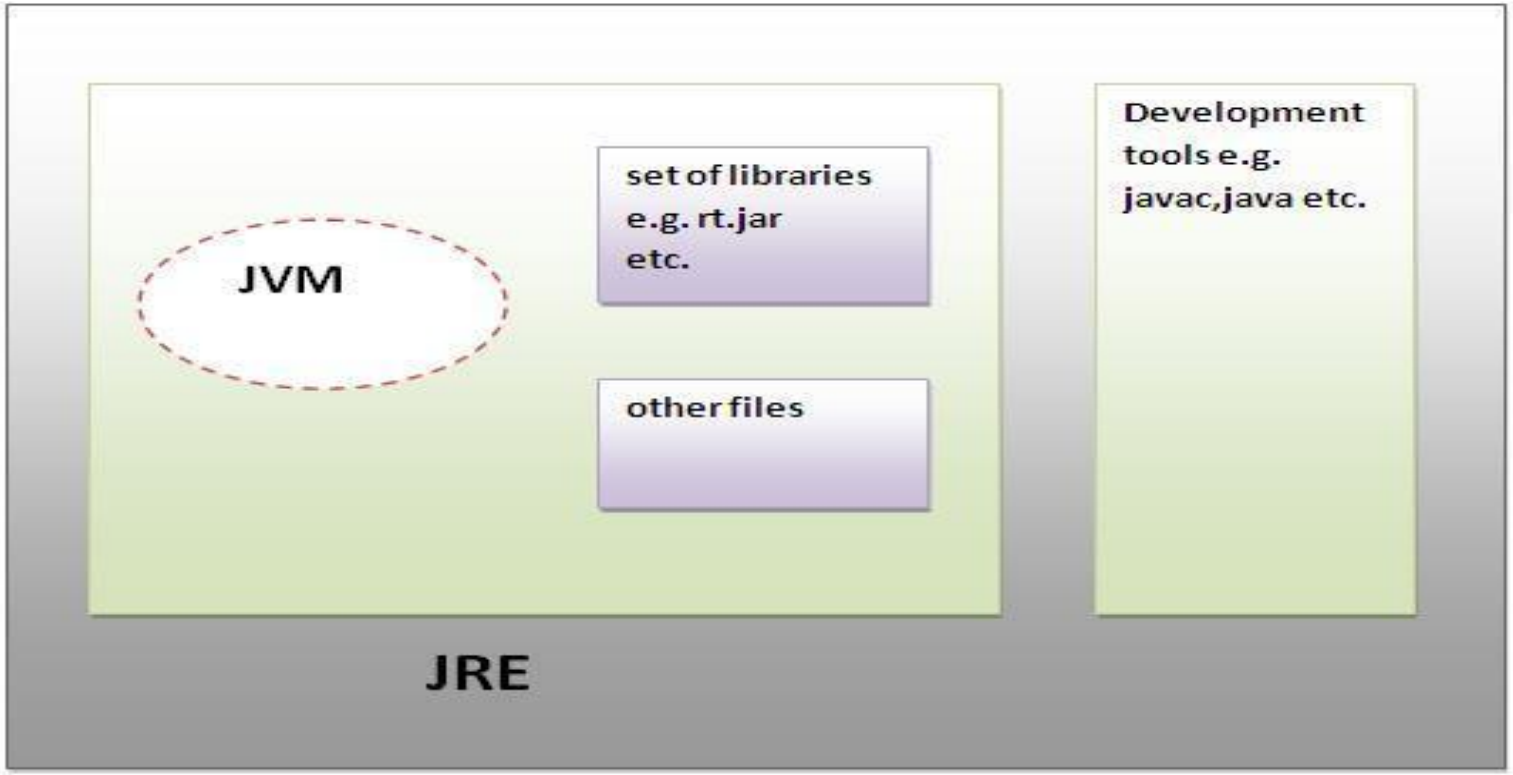
JDK

The Java Development Kit (JDK) is a cross-platformed software development environment that offers a collection of tools and libraries necessary for developing Java-based software applications.

It is a core package used in Java, along with the JVM (Java Virtual Machine) and the JRE (Java Runtime Environment).

JDK

- JDK is an acronym for Java Development Kit.It physically exists.It contains JRE + development tools.



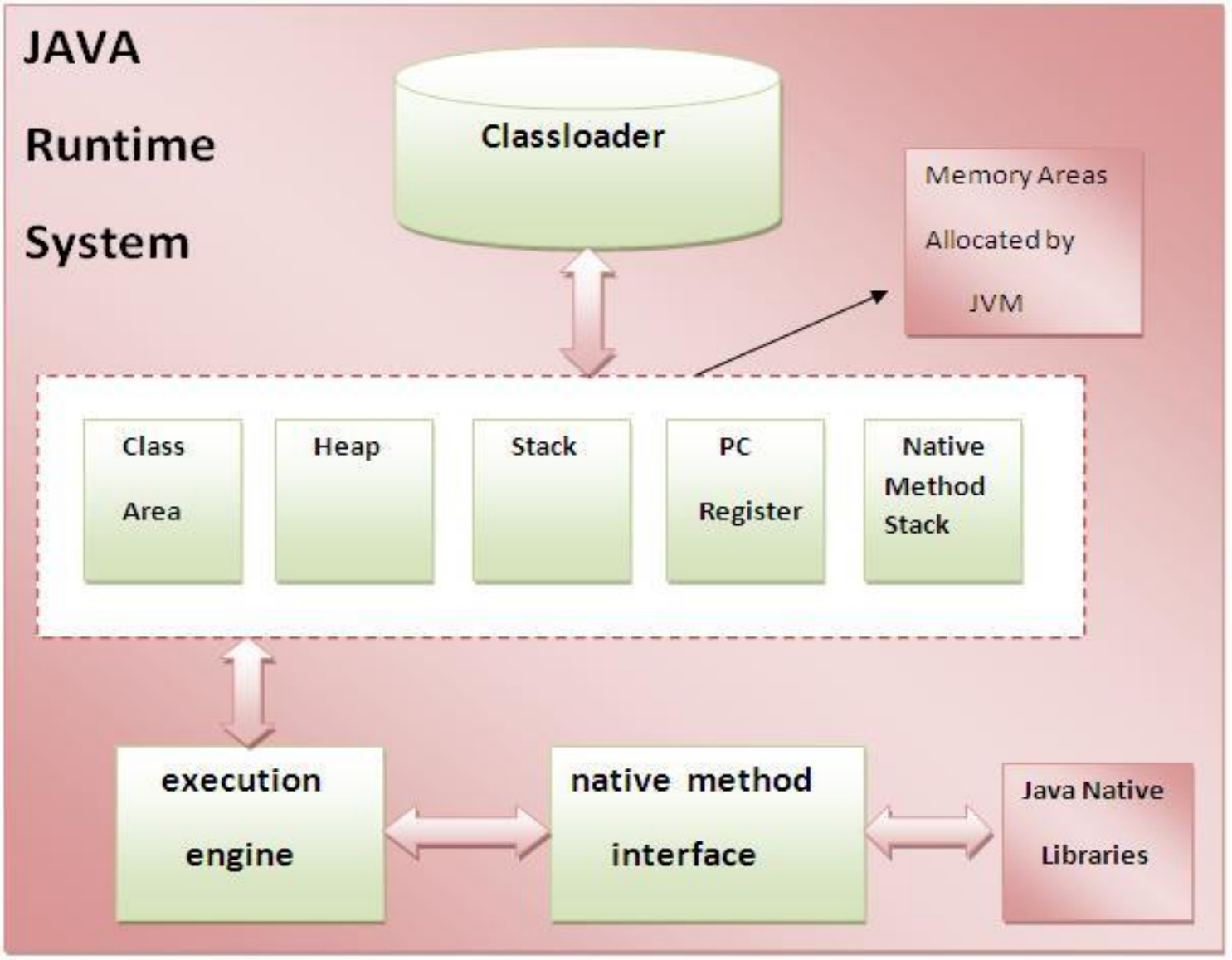
JVM

- JVM (Java Virtual Machine) is an abstract machine.
- It is a specification that provides runtime environment in which java byte code can be executed.
- JVMs are available for many hardware and software platforms.
- JVM, JRE and JDK are platform dependent because configuration of each OS differs. But, Java is platform independent.

The JVM performs following main tasks

- Loads code
- Verifies code
- Executes code
- Provides runtime environment

Internal Architecture of JVM



1) **Classloader**

Classloader is a subsystem of JVM that is used to load class files.

2) Class(Method) Area

Class(Method) Area stores per-class structures such as the runtime constant pool, field and method data, the code for methods.

3) Heap

It is the runtime data area in which objects are allocated.

4) Stack

It holds local variables and partial results, and plays a part in method invocation and return.

5) Program Counter Register

It contains the address of the Java virtual machine instruction currently being executed.

6) Native Method Stack

It contains all the native methods used in the application.

7) Execution Engine

It contains:

i) A virtual processor

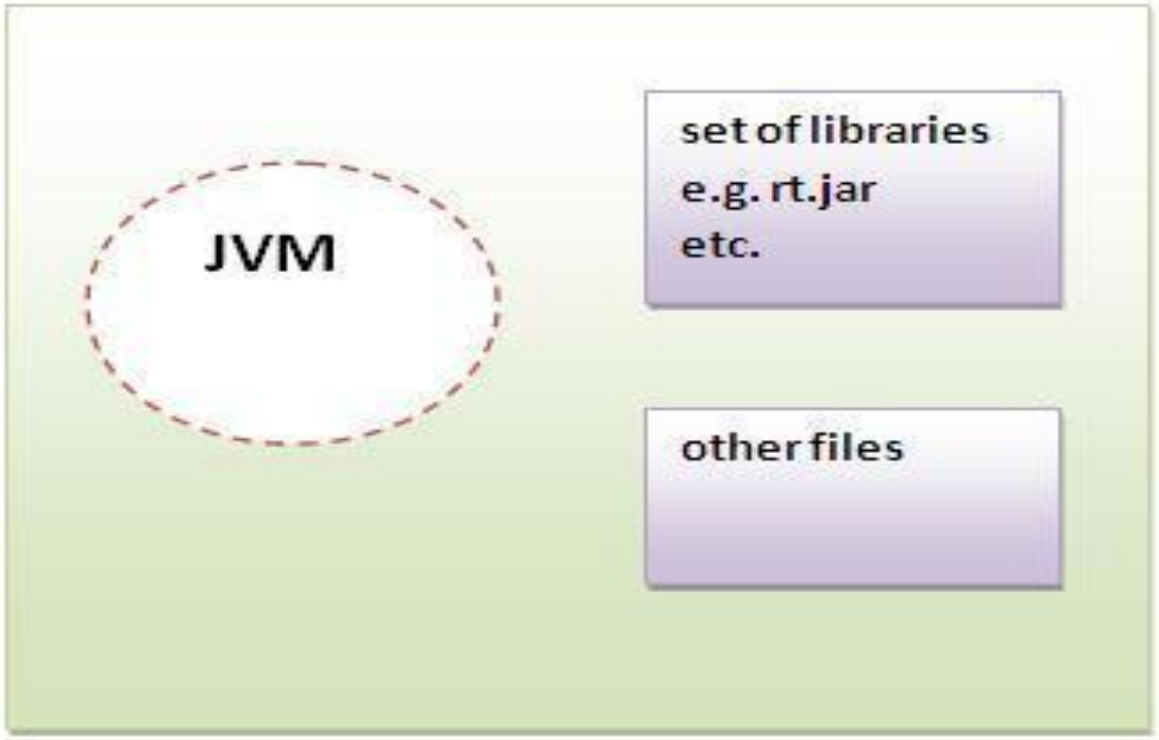
ii) Interpreter: Read bytecode stream then execute the instructions.

iii) Just-In-Time(JIT) compiler: It is used to improve the performance.JIT compiles parts of the byte code that have similar functionality at the same time, and hence reduces the amount of time needed.

JRE

- JRE is an acronym for Java Runtime Environment.
- It is used to provide runtime environment. It is the implementation of JVM.
- It physically exists.
- It contains set of libraries + other files that JVM uses at runtime.

JRE



JRE

Java Source File Structure

- In Java, a source file typically follows a specific structure to define classes, interfaces, and other elements of the program.
- Package Declaration (Optional):

  The package declaration is used to organize related classes and interfaces into a package. It is the first non-comment line in the file and is optional.

  For example: package com.example.myapp;

- **Import Statements (Optional):** Import statements are used to bring in classes or entire packages from other packages to use in the current source file. They appear after the package declaration (if present) and before the class declaration.

  For example:

  ```
  import java.util.ArrayList;
  import java.util.List;
  ```

- **Class Declaration:** A Java source file can contain one public class (with the same name as the file) and any number of non-public classes. The class declaration consists of the class keyword followed by the class name and optional modifiers (e.g., public, abstract, final). For example:

- **Interface Declaration (Optional):** Similar to classes, a Java source file can also contain interfaces. The interface declaration consists of the interface keyword followed by the interface name and optional modifiers.

  For example:

```
public interface MyInterface {
// interface body
}
```

- **Class or Interface Body:** The body of a class or interface contains fields, methods, constructors, and nested classes or interfaces. It is enclosed in curly braces {}.

  For example:

```
public class MyClass {
    private int myField;
    public MyClass(int value) {
        myField = value;
    }
    public void myMethod() {
        System.out.println("Hello, world!");
    }
    // nested class
    private class NestedClass {
        // nested class body
    }
}
```

- **Comments:** Java supports single-line comments (//) and multi-line comments (/* */) for adding explanations or documentation to the code.

How to Run Java Program

- Write Your Java Program: Create a Java source file with a .java extension.
- For example, let's say you have a simple program called HelloWorld.java

```
public class HelloWorld
{
   public static void main(String[] args)
   {
      System.out.println("Hello, world!");
   }
}
```

- **Compile Your Java Program**:

Open a command prompt and navigate to the directory containing your Java source file.

Use the javac command to compile your program.

javac HelloWorld.java

- **Run Your Java Program:** Use the java command to run your compiled Java program.

java HelloWorld

This will execute your HelloWorld class, and you should see the output Hello, world! printed to the console.