Website
https://www.curiousinfotechsolutions.com

Java Programming

Day-6

About Java Platform/Editions/Keywords	https://youtu.be/FJIRqjS5DIk

Email id:
info@curiousinfotechsolutions.com

Syllabus Day-6

- Types of Java Applications
- Java Platforms / Editions
- Difference between JDK, JRE, and JVM
- JVM (Java Virtual Machine) Architecture
- Java Keywords

Types of Java Applications

There are mainly 4 types of applications that can be created using Java programming:

1. Standalone Application
2. Web Application
3. Enterprise Application
4. Mobile Application

Standalone Application

- Standalone applications are also known as desktop applications or window-based applications.
- These are traditional software that we need to install on every machine.
- Examples of standalone application are Media player, antivirus, etc.
- AWT (**Abstract Window Toolkit**) and Swing are used in Java for creating standalone applications.

Web Application

- An application that runs on the server side and creates a dynamic page is called a web application.
- Currently, Servlet, JSP, Struts, Spring, Hibernate, JSF, etc. technologies are used for creating web applications in Java.
- **Servlet** technology is used to create a web application (resides at server side and generates a dynamic website)
- **JSP** technology provides more functionality than servlet such as JSTL (**JavaServer Pages Standard Tag Library**)
- Struts is **an open source framework that extends the Java Servlet API** and employs a Model, View, Controller (MVC) architecture. It enables you to create maintainable, extensible, and flexible web applications based on standard technologies, such as JSP pages, JavaBeans, resource bundles, and XML.
- Spring is a *lightweight* framework. It can be thought of as a *framework of frameworks* because it provides support to various frameworks such as Struts, Hibernate, Tapestry, EJB (Jakarta Enterprise Beans), JSF(JavaServer Faces), etc. ***The framework, in broader sense, can be defined as a structure where we find solution of the various technical problems***.

Enterprise Application

- An application that is distributed in nature, such as banking applications, etc. is called an enterprise application.
- It has advantages like **high-level security**, **load balancing**, and clustering.
- In Java, EJB (Jakarta Enterprise Beans) is used for creating enterprise applications.

Mobile Application

- An application which is created for mobile devices is called a mobile application.
- Currently, Android and Java ME are used for creating mobile applications.

2. Java Platforms / Editions

- There are 4 platforms or editions of Java:
  - Java SE (Java Standard Edition)
  - Java EE (Java Enterprise Edition)
  - Java ME (Java Micro Edition)
  - JavaFX (Java (F)orm E(X)perience)

Contd..

1) Java SE (Java Standard Edition)

- It is a Java programming platform.
- It includes Java programming APIs such as java.lang, java.io, java.net, java.util, java.sql, java.math etc.
- It includes core topics like OOPs, String, Regex, Exception, Inner classes, Multithreading, I/O Stream, Networking, AWT, Swing, Reflection, Collection, etc.

2) Java EE (Java Enterprise Edition)

- It is an enterprise platform that is mainly used to develop web and enterprise applications.
- It is built on top of the Java SE platform. It includes topics like Servlet, JSP, Web Services, EJB, JPA, etc.

3) Java ME (Java Micro Edition)

- It is a micro platform that is dedicated to mobile applications.

4) JavaFX

- JavaFX is an open source, next generation client application platform for desktop, mobile and embedded systems built on Java.
- It is used to develop rich internet applications. It uses a lightweight user interface API.

3. Difference between JVM, JRE, and JDK

- JVM (Java Virtual Machine) is an abstract machine.
- JRE is Java Runtime Environment.
- JDK is Java Development Kit.

JVM (Java Virtual Machine)

- JVM (Java Virtual Machine) is an abstract machine. It is called a virtual machine because it doesn't physically exist.

There are three notions of the JVM:
specification,
implementation, and
instance.

Contd..

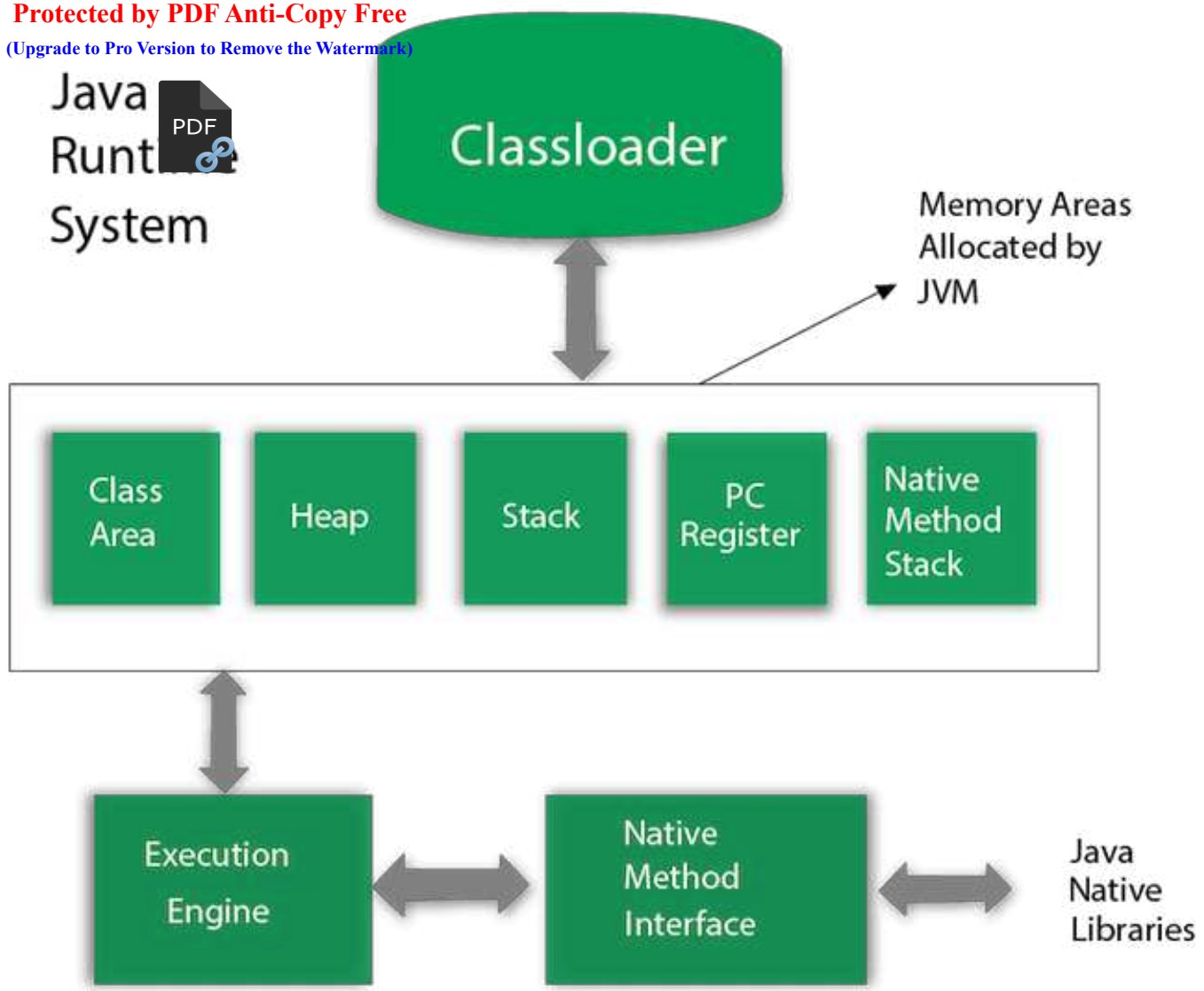
1. **A specification** where working of Java Virtual Machine is specified. But implementation provider is independent to choose the algorithm. Its implementation has been provided by Oracle and other companies.
2. **An implementation** Its implementation is known as JRE (Java Runtime Environment).
3. **Runtime Instance** Whenever you write java command on the command prompt to run the java class, an instance of JVM is created.

Contd..

- **The JVM performs following operation:**
  - Loads code
  - Verifies code
  - Executes code
  - Provides runtime environnent

- **JVM provides definitions for the:**
  - Memory area
  - Class file format
  - Register set
  - Garbage-collected heap
  - Fatal error reporting

JVM Architecture



Contd..

1) Classloader

- Classloader is a subsystem of JVM which is used to load class files. Whenever we run the java program, it is loaded first by the classloader.

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.

Contd..

4) Stack

- Java Stack stores frames. It holds local variables and partial results, and plays a part in method invocation and return.

5) Program Counter Register

- PC (program counter) register 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:

- A virtual processor
- Interpreter: Read bytecode stream then execute the instructions.
- 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 for compilation.

8) Java Native Interface

- Java Native Interface (JNI) is a framework which provides an interface to communicate with another application written in another language like C, C++, Assembly etc.
- Java uses JNI framework to send output to the Console or interact with OS libraries.

Keywords In Java

1. abstract
2. assert
3. boolean
4. break
5. byte
6. case
7. catch
8. char
9. class
10. continue
11. default
12. do
13. double
14. else
15. enum
16. extends
17. final
18. finally
19. float
20. for
21. if
22. implements
23. import
24. instanceof
25. int
26. interface
27. long
28. native
29. new
30. package
31. private
32. protected
33. public
34. return
35. short
36. static
37. strictfp
38. super
39. switch
40. synchronized
41. this
42. throw
43. throws
44. transient
45. try
46. void
47. volatile
48. while

Contd..

- **abstract:** Java abstract keyword used to declare an abstract class. An abstract class can provide the implementation of the interface.
- **assert:** An assertion is a statement in Java which ensures the correctness of any assumptions which have been done in the program.
- **implements:** Java implements keyword is used to implement an interface.
- **instanceof:** Java instanceof keyword is used to test whether the object is an instance of the specified class or implements an interface.
- **native**: Java native keyword is used to specify that a method is implemented in native code using JNI (Java Native Interface).

Contd..

- **Strictfp:** Java strictfp is used to restrict the floating-point calculations to ensure portability.

Synchronized: Java synchronized keyword is used to specify the critical sections or methods in multithreaded code.

this: Java this keyword can be used to refer the current object in a method or constructor.

throw: The throw keyword is mainly used to throw custom exceptions. It is followed by an instance.

throws: Checked exceptions can be propagated with throws.

transient: Java transient keyword is used in serialization. If you define any data member as transient, it will not be serialized.