Java Architecture Overview

An Introduction to the Java Platform

Introduction

- • Definition of Java
- • Why Java is important (platform independence, security, performance)
- • Brief overview of the architecture components

Java Architecture Components

- • JVM (Java Virtual Machine)
- • JRE (Java Runtime Environment)
- • JDK (Java Development Kit)

JVM (Java Virtual Machine)

- • Responsibilities of JVM (Memory management, Garbage collection, Class loading)
- • Converts bytecode to machine code
- • Just-In-Time (JIT) compilation

Class Loader Subsystem

- • Class loading process
- • Bootstrap, Extension, Application Class Loaders
- • Role in memory management and security

Memory Management in Java

- • Stack vs. Heap memory
- • Garbage collection process
- • Types of garbage collectors (Serial, Parallel, CMS, G1)

Execution Engine

- • Interpreter: Translates bytecode to machine code
- • JIT Compiler: Improves performance by compiling frequently used code
- • HotSpot optimization

Java Native Interface (JNI)

- • Purpose: Interact with native applications (C/C++)
- • When to use JNI
- • Java interaction with external code

Java Security Architecture

- • Bytecode verification
- • Class loader security
- • Sandboxing and security manager

Java Framework and Libraries

- • Overview of standard libraries (Collections, I/O, Networking)
- • Mention of third-party frameworks (Spring, Hibernate)

Java Development Tools (JDK)

- • javac (compiler)
- • java (JVM launcher)
- • javadoc (documentation generator)
- • jdb (debugger)

Advantages of Java Architecture

- • Platform independence (Write Once, Run Anywhere)
- • Scalability and flexibility
- • Security features
- • Rich APIs and community support

Conclusion

- • Recap of Java architecture components (JVM, JRE, JDK)
- • Importance for development
- • Java's relevance in modern software development