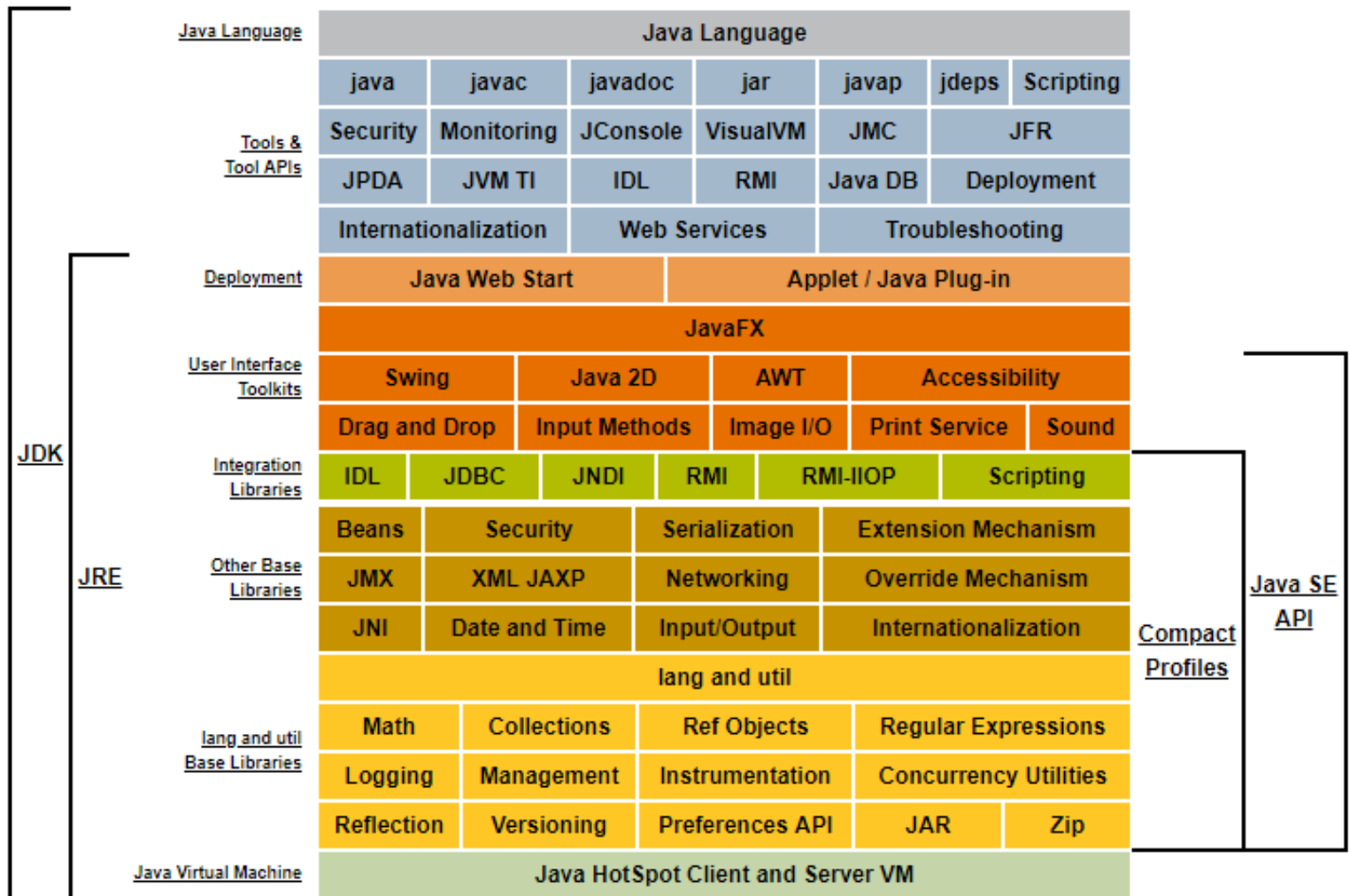
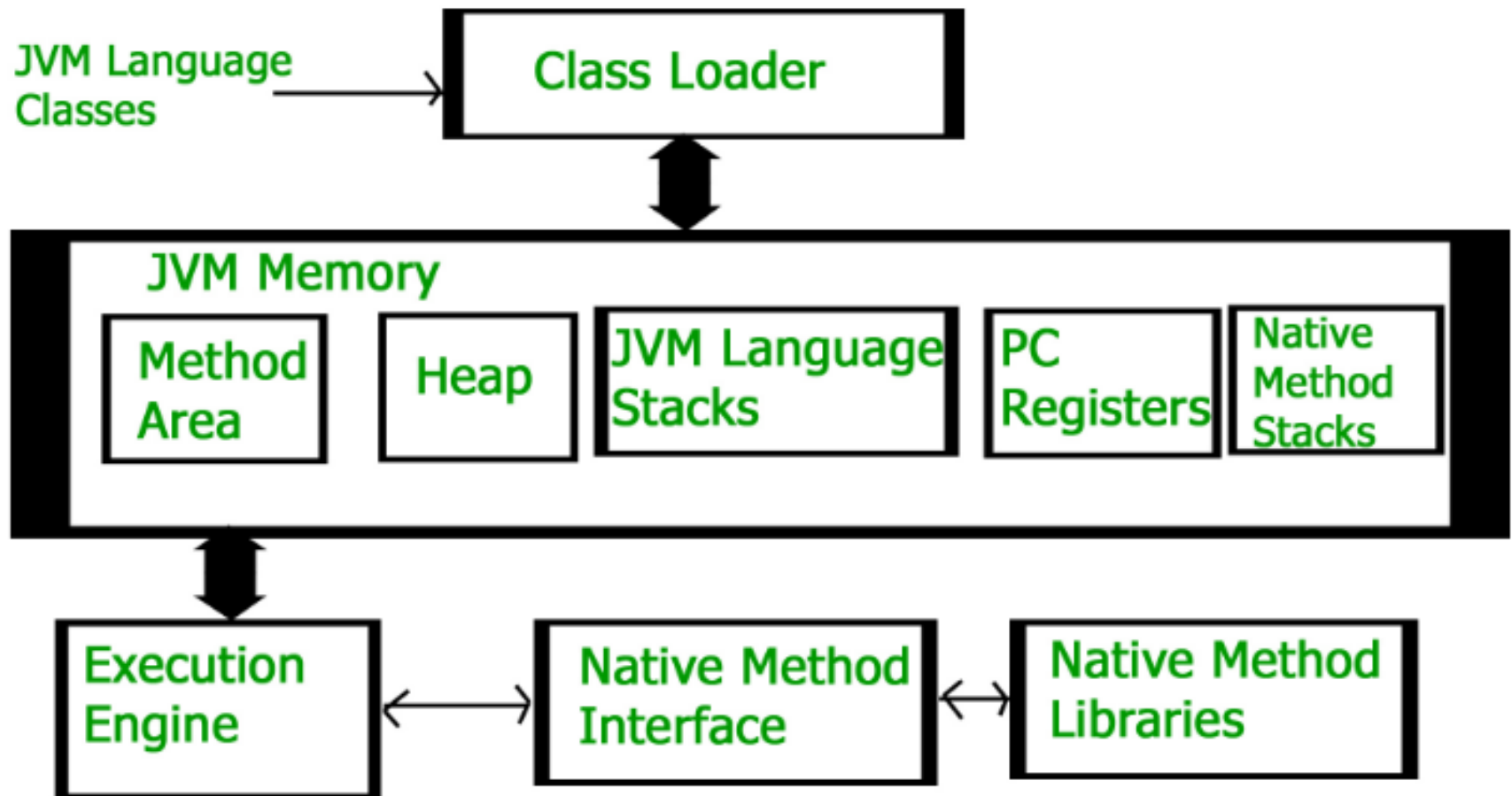


JAVA INTERNALS / ARCHITECTURE

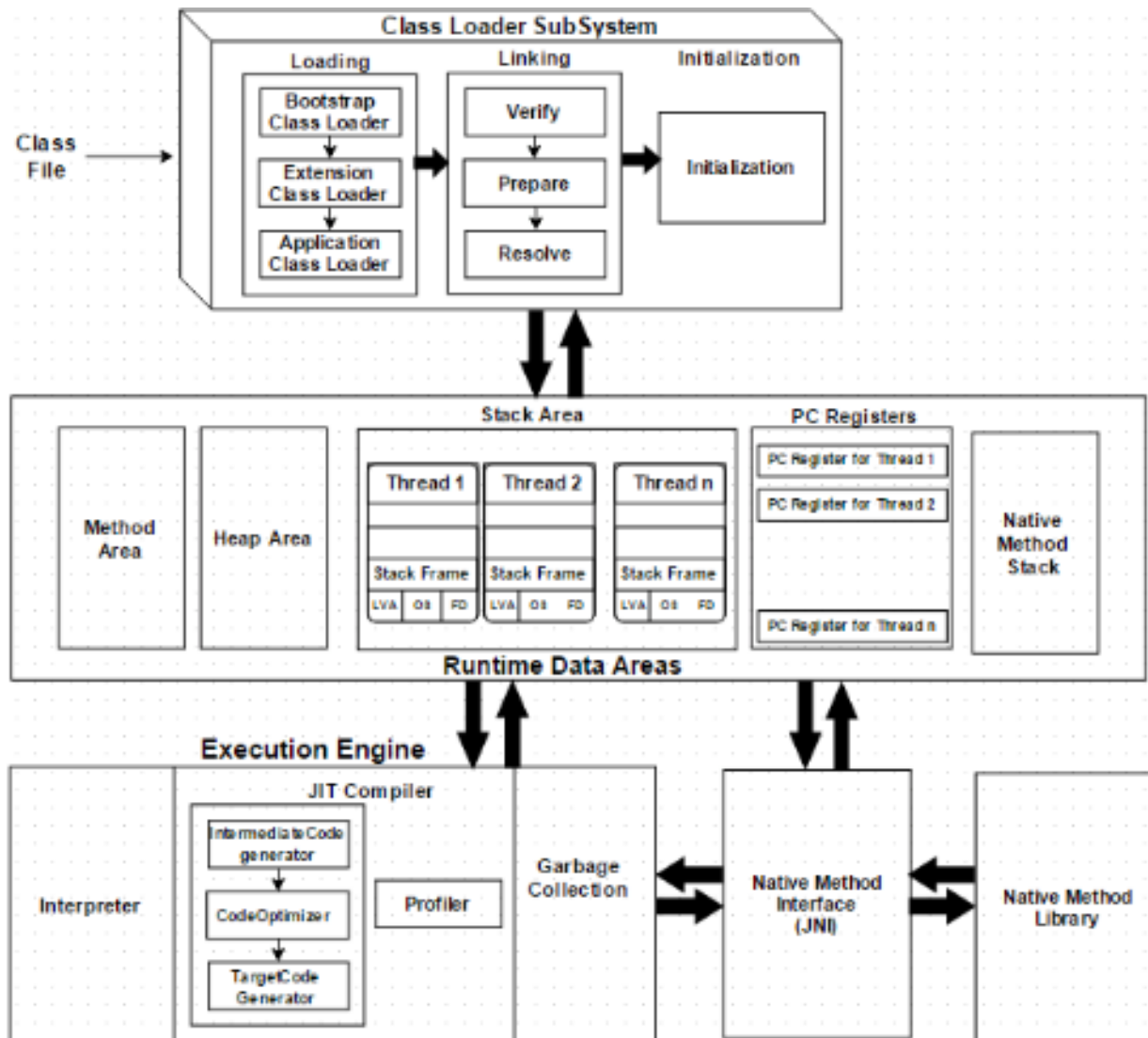
Java Conceptual Model (JVM/JRE/JDK)



JVM Architecture



JVM Architecture (detailed)



JVM Components – Class Loader Subsystem

} **Loading** - Classes will be loaded by this component

} Boot Strap – Loads classes from the bootstrap classpath

} Extension – Loads classes which are inside the ext folder

} Application – Loads from Application Level Classpath, Environment Variable etc

} **Linking**

} Verify - Bytecode verifier will verify whether the generated bytecode is proper or not

} Prepare - For all static variables memory will be allocated and assigned with default values

} Resolve - All symbolic memory references are replaced with the original references from Method Area

} **Initialization**

} All static variables will be assigned with the original values, and the static block will be executed

JVM Components – Runtime Data Area

- } **Method Area** - All the class level data will be stored here, including static variables
- } **Heap Area** - All the Objects and their corresponding instance variables and arrays will be stored here
- } **Stack Area** - For every thread, a separate runtime stack will be created.
All local variables will be created in the stack memory.
- } **PC Registers** - Each thread will have separate PC Registers, to hold the address of current executing instruction once the instruction is executed the PC register will be updated with the next instruction
- } **Native Method Stacks** - Native Method Stack holds native method information.
For every thread, a separate native method stack will be created.

JVM Components – Execution Engine

- } Interpreter
- } JIT Compiler
 - Intermediate Code Generator
 - Code Optimizer
 - Target Code Generator
 - Profiler
- } Garbage Collectors
- } Java Native Interface
- } Native Method Libraries

JVM Internals - Memory Management

} Memory Spaces

- } Heap - Primary storage of the Java program class instances and arrays
 - Young Generation [Eden Space, Survivor Space]
 - Old Generation
- } PermGen/Metaspace - Primary storage for the Java class metadata
- } Native Heap - native memory storage for the threads, stack, code cache including objects such as MMAP files and third party native libraries

JVM Internals – Garbage Collectors

- } Serial Garbage Collector - Single threaded. Freezes all app threads during GC
- } Parallel Garbage Collector - Multi threaded. Freezes all app threads during GC
- } Concurrent Mark Sweep - Multi threaded with shorter GC pauses
- } G1 Garbage Collector - Divides heap space into many regions and GCs region have more garbage

JVM Internals – Hotspot

- } Region of a computer program where a high proportion of executed instructions occur or where most time is spent during the program's execution
- } **Client VM** - Tuned for quick loading. It makes use of interpretation.
- } **Server VM** - Loads more slowly, putting more effort into producing highly optimized JIT compilations to yield higher performance
- } **Tiered Compilation** - uses both the client and server compilers in tandem to provide faster startup time than the server compiler, but similar or better peak performance

Thank You!