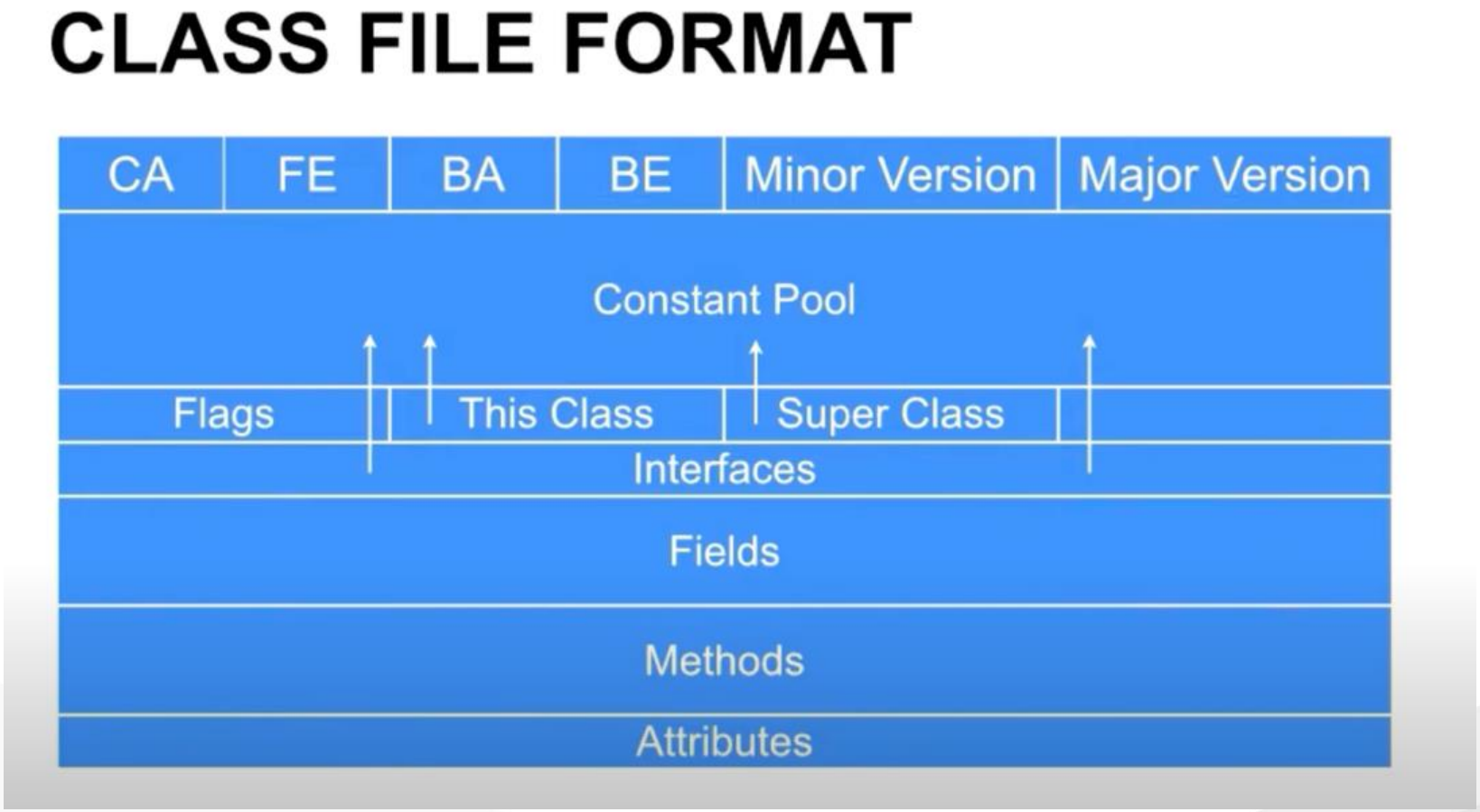
JVM

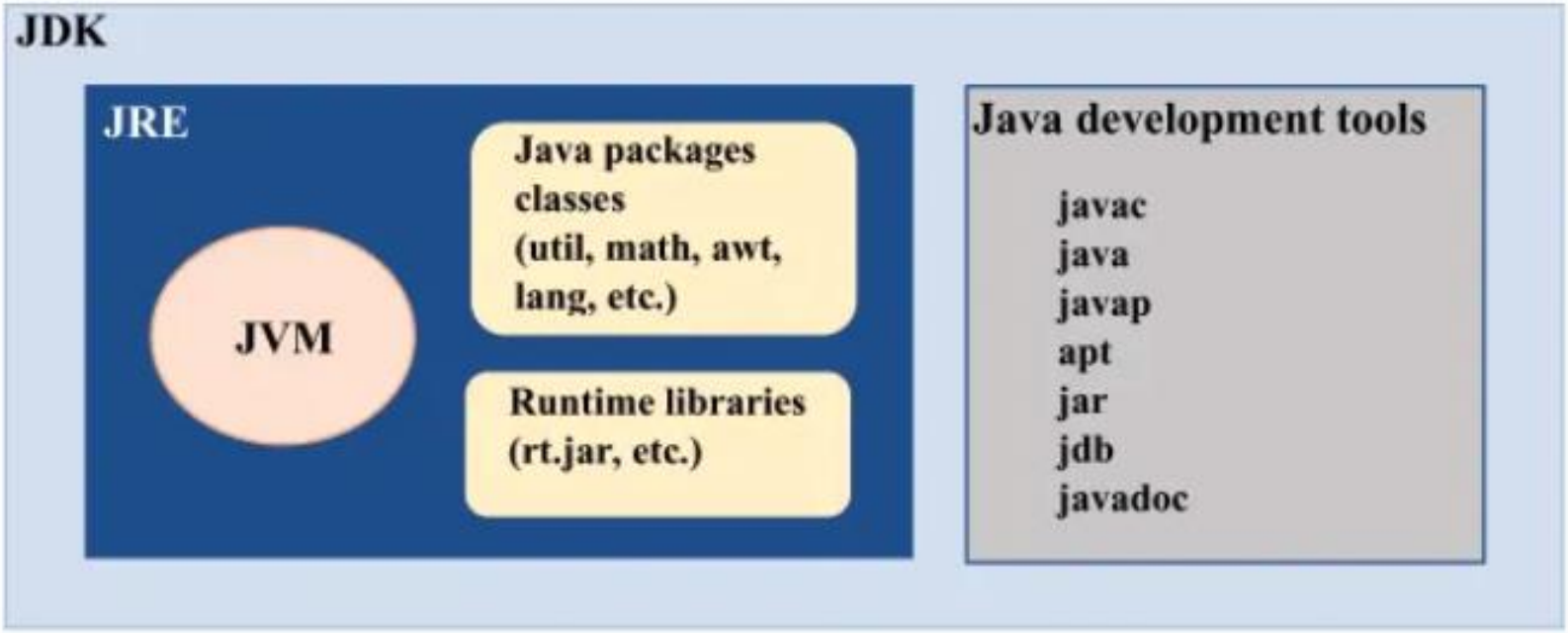
Shalini Mittal
Corporate Trainer

Class File Format



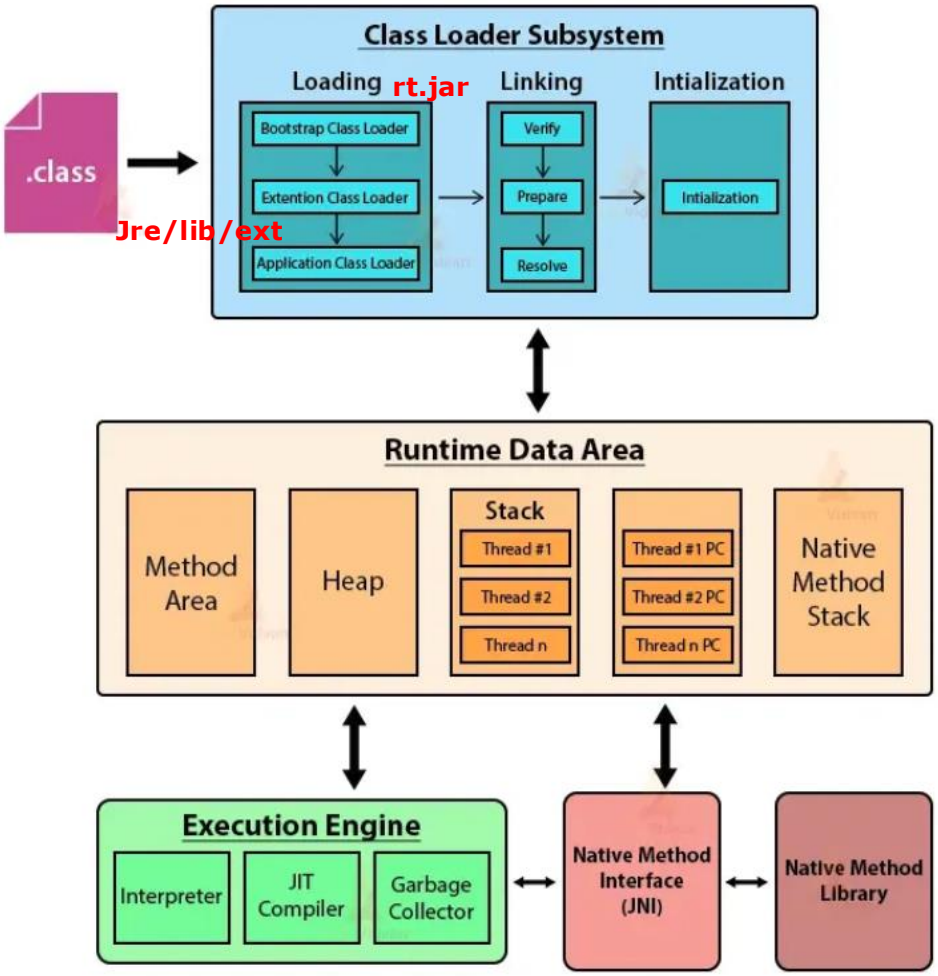
javap to see the format
To see list of all JVM flags:
java -XX:+UnlockDiagnosticVMOptions -XX:+PrintFlagsFinal -version

Java Architecture

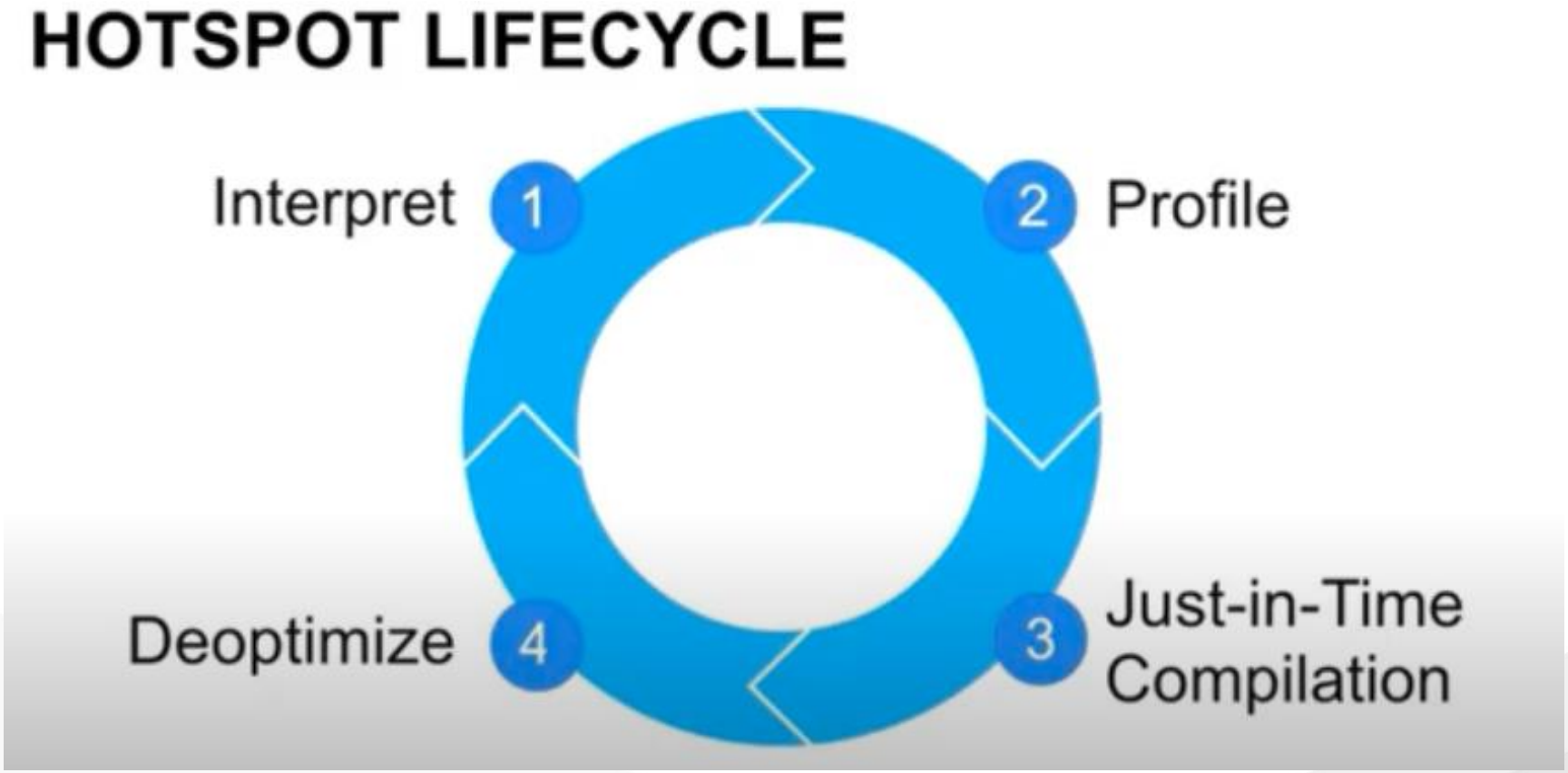


Java Architecture

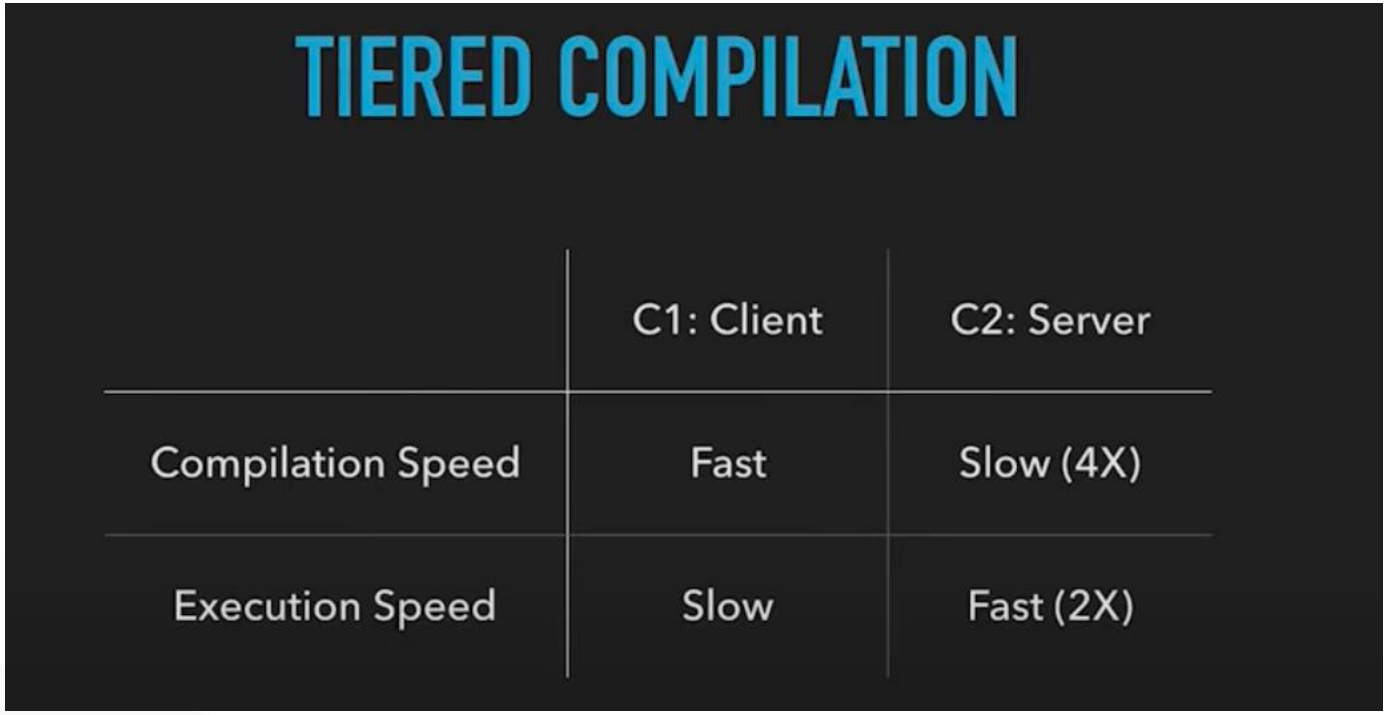
JVM Architecture



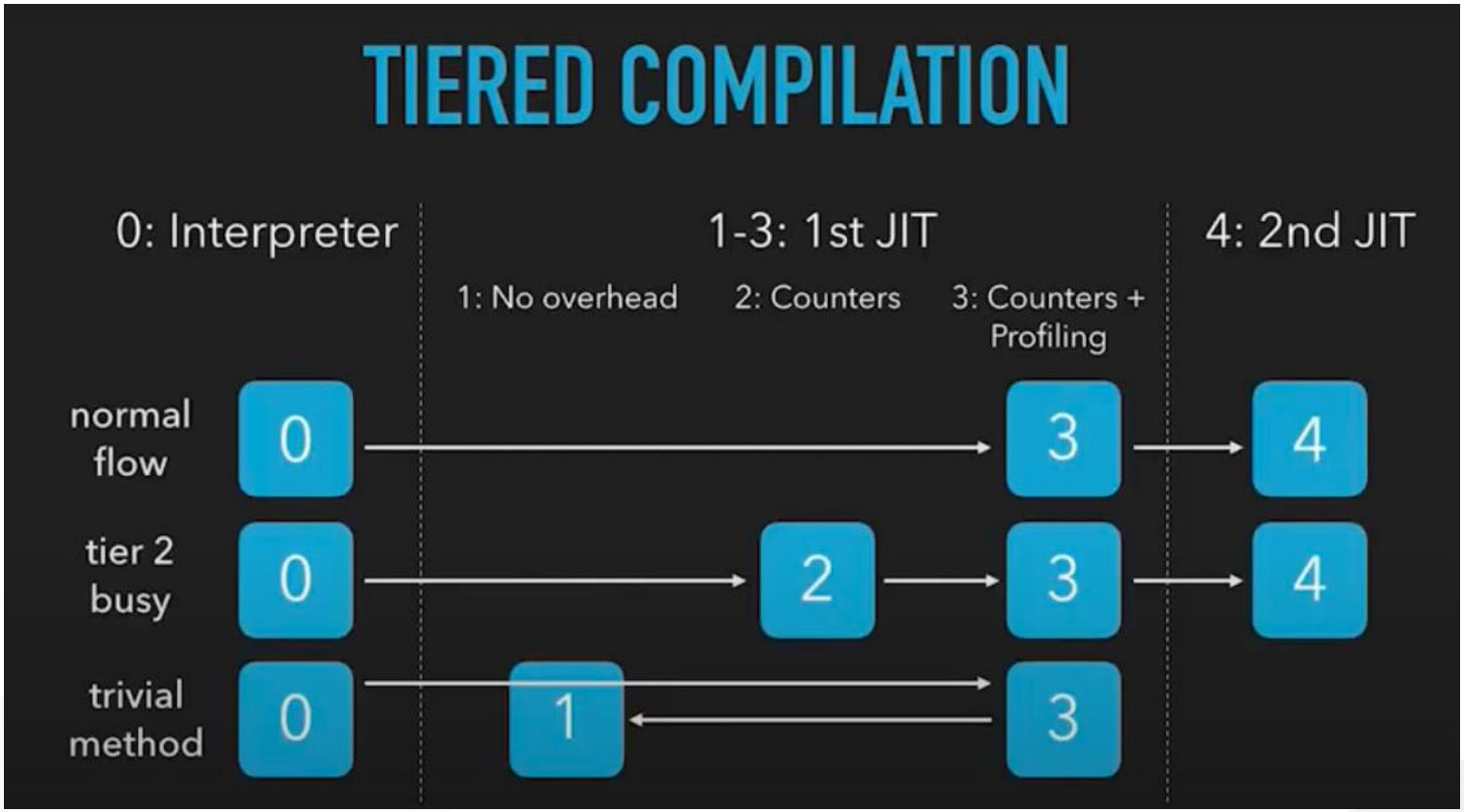
Hotspot Lifecycle



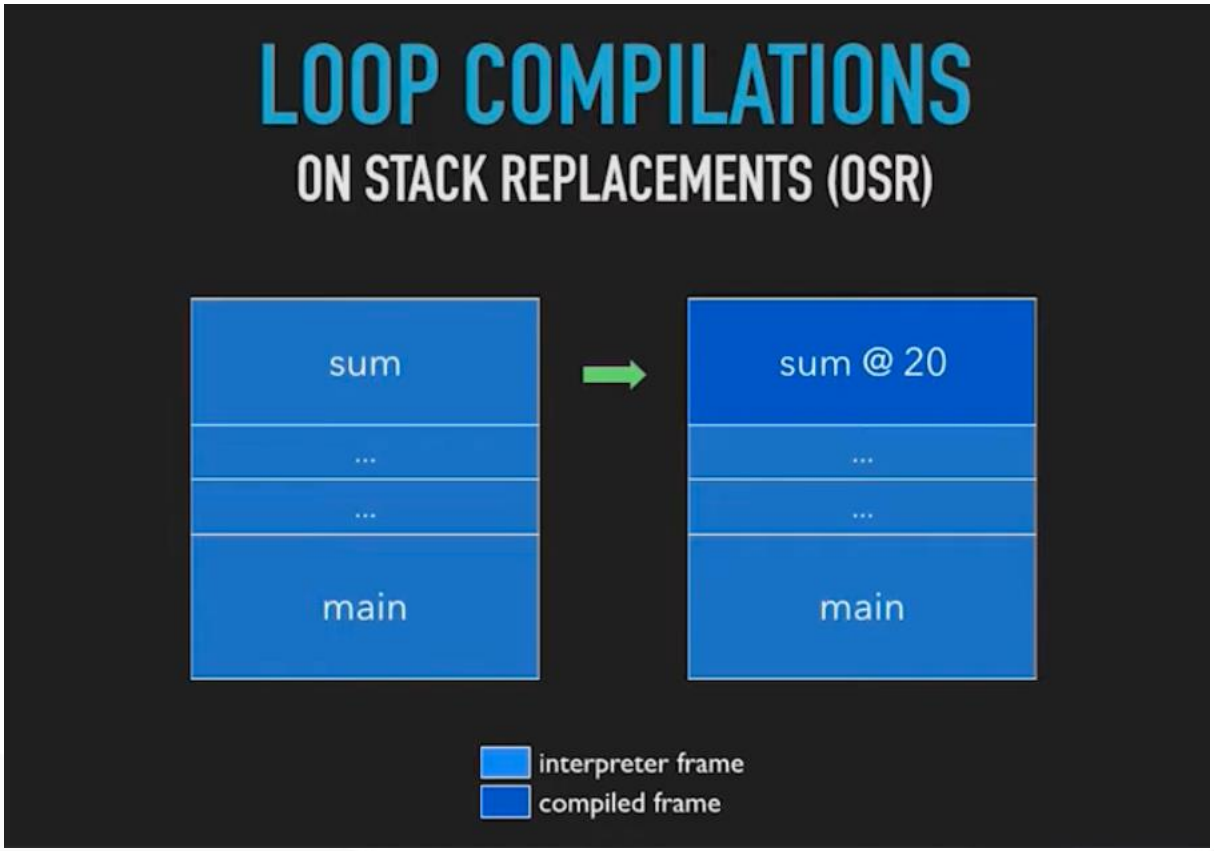
JIT Compiler Types



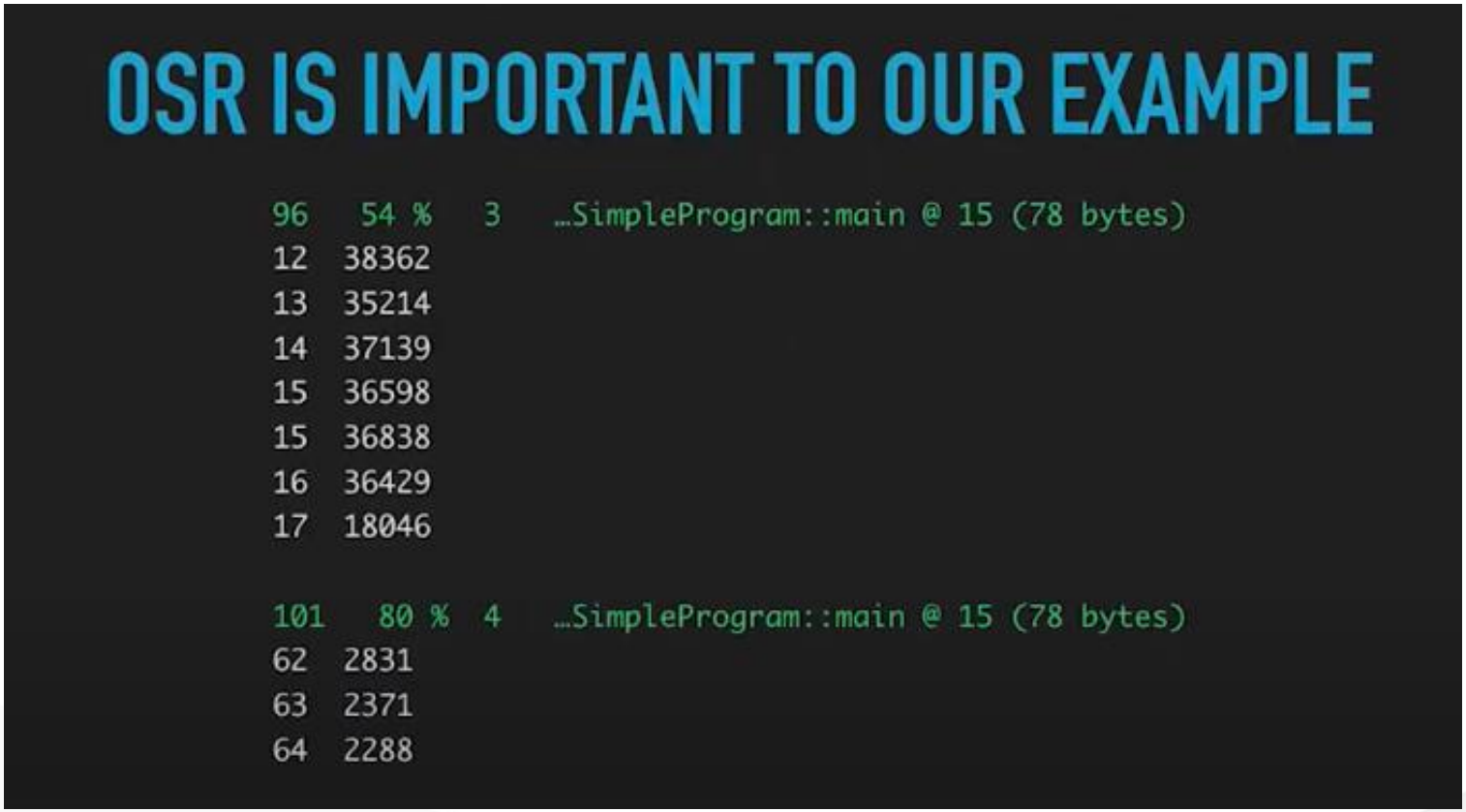
Tiered Compilation



Loop OSR


LOOP COMPILATIONS
ON STACK REPLACEMENTS (OSR)
sum
...
...
main
sum @ 20
...
...
main
interpreter frame
compiled frame

PrintCompilation



When is JIT called

Tune Code Cache

- java –XX:+PrintCodeCache main.<classname>
- The maximum size of the code cache is dependent on which version of Java you are using.
    - If you are using Java 7 or below, then it was either 32MB(32 bit JVM) or 48MB(64 bit JVM)
    - If you're using Java eight or above and you're using the 64 bit JVM, then the code cache can be up to 240MB.
- So we can change the code cache size with three different flags with bytes, KB or MB

  InitialCodeCacheSize
  ReservedCodeCacheSize
  CodeCacheExpansionSize

- Use jconsole to monitor code cache usage over time

CICompilerCount

- Used to control the number of threads dedicated to the Just-In-Time (JIT) compilation process related to the C1 and C2 compilers.
- Can specify threads the JVM should allocate for these compilation tasks to JIT compilers.
- Considerations:
    - **Performance Tuning:**
      Adjusting CICompilerCount can impact application startup time, overall performance, especially in scenarios with high compilation activity.
    - **Resource Consumption:**
      Increasing the number of compiler threads can lead to higher CPU consumption, particularly during the initial phases of application execution or when code is frequently being recompiled.
    - **Default Values:**
      The JVM has default values for CICompilerCount based on the architecture and available resources. It is generally recommended to use the default unless specific performance issues related to compilation are observed.

Java tools and JVM flags

- java -XX:+PrintFlagsFinal -version | grep CICompilerCount
  OR
  jinfo -flag CICompilerCount 45081

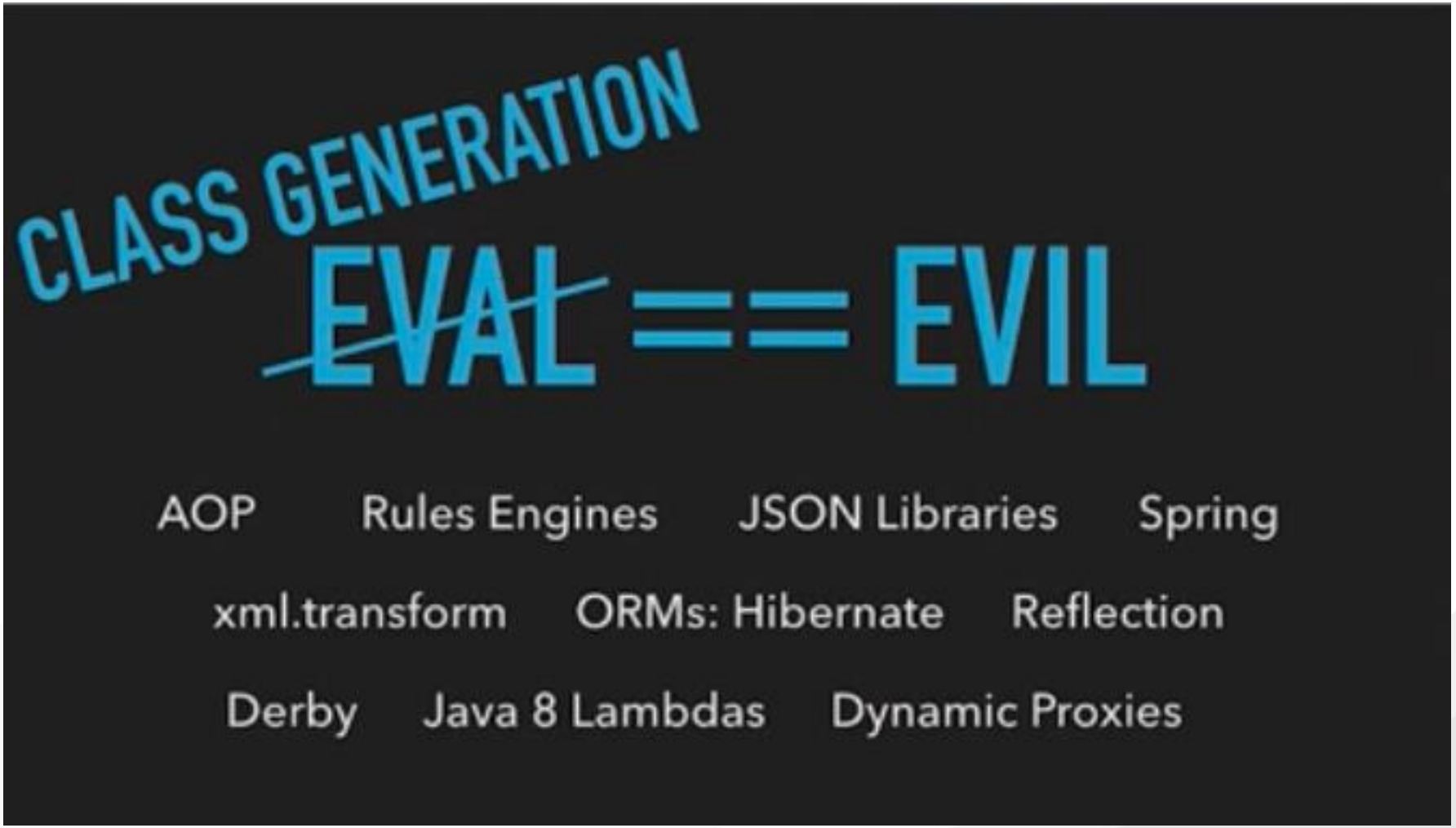
```
[(base) Manishs-MacBook-Pro:bin Shalini$ jps
 47988 Jps
 45081 Eclipse
[(base) Manishs-MacBook-Pro:bin Shalini$ jinfo -flag CICompilerCount 45081
 -XX:CICompilerCount=4
```

- java -XX:+PrintCompilation main.Main 150000
- java -XX:CICompilerCount=6 -XX:+PrintCompilation main.Main 150000
- jinfo -flag CompileThreshold 45081

```
[(base) Manishs-MacBook-Pro:bin Shalini$ jinfo -flag CompileThreshold 45081
 -XX:CompileThreshold=10000
```

GOAL?


WHAT'S YOUR GOAL?
STARTUP TIME
PEAK PERFORMANCE
FIRST RESPONSE PERFORMANCE
PREDICTABLE PERFORMANCE


CLASS GENERATION
EVAL == EVIL
AOP
Rules Engines
JSON Libraries
Spring
xml.transform
ORMs: Hibernate
Reflection
Derby
Java 8 Lambdas
Dynamic Proxies

JVM Versions

32 bit

Might be faster if heap < 3GB

Max heap size = 4GB

Client compiler only

64 bit

Might be faster if using long / double

Necessary if heap > 4GB

Max heap size – OS dependent

Client & server compilers

ThankYou

shalini06mittal@gmail.com

7738460004