Java Performance Tuning - COURSE CONTENT

Ch1: Introduction to Profiling, Monitoring & Tuning

- ☐ What is profiling & monitoring
- ☐ What to monitor : GC, Methods, CPU Usage, Object Creation, Threads, I/O
- ☐ How to Monitor : Profiling, Sampling, JVM TI / PI, Byte Code Instrumentation

Ch2: Understanding Garbage Collection in Java

- ☐ Why GC monitoring is important from performance perspective, Tracing Collectors / MarkSweep, Copy Collectors, Understanding the pros and cons of Copy & MarkSweep
- ☐ Heap fragmentation, heap expansion & compaction
- ☐ Generational collectors in HotSpot
- ☐ Serial GC, ParallelGC & Concurrent GC

Ch3: Monitoring & Tuning Garbage Collection in Java

- ☐ Monitoring Heap expansion & contraction
- ☐ Understanding reasons of OOME
- ☐ Generating & understanding the GC log files
- ☐ Monitoring frequency of GC & time it takes
- ☐ Finding pre-matured promotions
- ☐ Configuring rotating log files
- ☐ Sizing the heap & generations
- ☐ Selecting the right collector to tune GC

Ch4: Profiling the CPU (Method Profiling)

- ☐ What is CPU Profiling and why is it needed

- ☐ Understanding Hotspots in threads
- ☐ Tools to get the hotspots
- ☐ Understanding the hotspot - call tree

Ch5: Enhancing Performance of CPU Hungry Code

- ☐ Parallelizing loops for performance
- ☐ Do not reinvent the wheel - Using Cyclic Barrier, Phasers, Fork & Joins
- ☐ Optimizing usage of Strings
- ☐ Optimizing hashCode & equals for HashMaps
- ☐ Database Tuning

Ch6: Java Memory Profiling & Monitoring

- ☐ What is Memory Profiling & why is it needed
- ☐ Understanding heap occupancy & detecting Memory Leaks
- ☐ Understanding ClassLoader Leaks
- ☐ Generating heap dumps
- ☐ Understanding the concept of Shallow Size, Retained Size & Dominating Objects
- ☐ Using Eclipse MAT to detect memory leaks

Ch7: Object Creation Techniques

- ☐ Efficient working with Collection Data Structures
- ☐ Canonicalizing Objects emptySet, emptyList, emptyMap
- ☐ Working with Exception Objects
- ☐ Caching and its impact on performance
- ☐ Developing Canonicalized Mappings using WeakReferences
- ☐ Developing Memory-Sensitive Caches using SoftReferences
- ☐ Pooling, Singleton, Prototype
- ☐ Timing the object creation - eager, early

Ch8: Monitoring Threads

- ☐ Understanding Thread States
- ☐ Understanding DeadLock, LiveLock & Starvation
- ☐ What is thread contention
- ☐ What are entry-sets and wait-sets
- ☐ Understanding Memory Barriers

Ch9: Coding Concurrency for Performance

- ☐ General Techniques to improve performance of Concurrency code
- ☐ Using Volatile, Atomic Variables & Contended Locks
- ☐ Using Lock API - tryLock etc to avoid deadlocks
- ☐ Code Motion
- ☐ Code Fusion
- ☐ Concurrent Data Structures in java.util.concurrent

Ch10: HotSpot Tuning for 64 Bit & NUMA

- ☐ Memory Considerations on 64Bit Architectures
- ☐ Compressed Oops

- ☐ Escape Analysis
- ☐ NUMA Collector Enhancements