



Vos contacts :
Marie-Jeanne ou Marjorie
au : 05 61 34 39 80

JAVAOPT - Optimisations de code Java

Objectif	A l'issue de ce stage, vous serez apte à évaluer la consommation des ressources engendrées par une modélisation et par l'implémentation de cette dernière. L'étude des mécanismes de bas niveaux de la JVM permettra de mieux ressentir les différents concepts présentés. Des outils graphiques de monitoring et de profiling d'applications Java seront de plus présentés
Pré requis	Pour suivre ce stage, il est nécessaire de maîtriser les bases du langage Java....
Durée	3 jours

Contenu

Module 1 : Quelques rappels fondamentaux

- L'environnement J2SE (Java 2 Standard Edition)
- Concepts fondamentaux : ClassLoader, JIT, GC, Gestionnaire d'exceptions, ...
- JVMPI (Java Virtual Machine Profiling Interface)
- Quelques outils du J2SE : javap, ...
- Notion de byte code
- Utilisation du désassembleur javap
- Notions de pile et de tas (Stack & Heap)
- Mécanisme de déclarations : variables locales et attributs
- Appels de méthodes en assembleur
- Quelques considérations sur les performances
- Temps d'exécution VS taille de l'exécutable
- Génie logiciel VS Performances

Module 2 : Développement d'agents JVMPI

- Concepts fondamentaux de JVMPI
- Les différentes versions l'interface JVMPI
- Les événements JVMPI
- Prises de mesure de temps et applications multithreadées
- Codage d'agents JVMPI
- Traquer l'activité du ClassLoader
- Traquer l'activité en termes d'allocations dynamiques
- Traquer l'activité en termes d'appels de méthodes

Module 3 : "Guide lines" de programmation

- Encapsulation VS performance
- Coût d'invocation de getter/setter
- Optimisation des temps de développement
- Fonctionnement de l'instruction switch
- Limitation d'instances temporaires
- Utilisation des chaînes de caractères
- Utilisation des collections Java
- Choisir les bons algorithmes
- Collection synchronisées ou non (Vector vs ArrayList, ...)
- Les différentes techniques de parcours d'une collection
- Utilisation de tableaux Java typés
- Accès indexés aux valeurs
- Traitements des valeurs de types primitifs

Module 4 : Le garbage collector (GC)

- Aspects fondamentaux
- Tâches du garbage collector : libération et défragmentation
- Les différents algorithmes utilisés
- Fonctionnement du Garbage Collector de la JVM Hotspot (Sun Microsystems)
- Monitorer l'activité du GC
- Collectes mineures et collectes majeures
- Paramétrage du Garbage Collector de la JVM HotSpot
- Gestion de la taille des Heap (-Xmx, -Xms, -XX:NewRatio, -XX:SurvivorRatio, ...)
- Libération incrémentale des ressources
- Utilisation multi-threadées du garbage collector

Module 5 : Recyclons les ressources de la JVM

- Utilisation d'un pool d'instances
- Implémentation d'un pool d'instances
- Utilisation d'un pool d'instances
- Etude du comportement du GC via un agent JVMPI
- Application multithreadées
- Monitoring via JVMPI sur l'utilisation des threads
- Le package java.util.concurrent
- Utilisation de pool de threads
- SoftReferences et WeakReferences

Module 6 : Outils graphiques de monitoring et de profiling

- La JConsole
- Présentation du modèle JMX (Java Monitoring eXtensions)
- Démarrage d'une JVM supportant le monitoring
- Lancement et attachement de la JConsole
- Les différentes possibilités de la JConsole
- TPTP (Test and Performance Tool Platform)
- Rappel sur l'utilisation de plugin Eclipse
- Présentation du plugin TPTP
- Test de détection de fuite de mémoire
- Mesure de performances d'une application Java
- Taux de couverture de code