Tél : (+33)1 55 60 76 72
Email : formation@oxiane.com

Java Optimisation

Améliorer la performance de ses applications Java

Les applications Java, comme toute autre application informatique, souffrent généralement de problèmes de performance, souvent détectés tardivement.

Il n'est pas toujours possible de se reposer sur les exploitants en augmentant les capacités de l'infrastructure (bien que ce soit parfois une solution plus rapide et économique).

Il faut alors demander aux développeurs de replonger dans leur code pour l'optimiser.

Encore faut-il avoir une démarche et des outils pour comprendre ce qu'il faut optimiser et savoir comment le faire.

Détails

- Code : JA-OPT
- Durée : 3 jours (21 heures)

Public
- Architectes
- Développeurs
- Ingénieurs
- Consultants

Pré-requis
- Bonne pratique de Java

Objectifs
- Appliquer une démarche d'optimisation
- Comprendre le fonctionnement de la machine virtuelle (JVM)
- Connaître et savoir utiliser les outils d'audit et de mesure
- Repérer les fuites mémoire
- Comprendre la problématique des applications multi-thread, détecter (ou éviter) les erreurs

Programme

La performance
- Définition
- La performance pour tous
- Les éléments composant la performance
- La performance perçue
- La performance au démarrage
- Impact dans les couches d'une application
- Performance et architecture
- La notation « grand O »

Le processus d'optimisation
- Analyser
- Coder
- Mesurer pour vérifier

Benchmarking
- Mesure de temps
- Les pièges
- Le benchmarking
- Les microbenchmarks
- JMH
- Recommandations

Les outils
- Profiling
- JVisualVM
- Eclipse MAT
- SoapUI
- JMeter

Les principaux soucis de performance
- Les bottlenecks
- Les origines

L'utilisation efficace des API
- API Collections
- L'utilisation des wrappers
- La méthode hashCode()

Gérer les entrées sorties
- L'utilisation des classes à bon escient
- La sérialisation
- La sérialisation personnalisée
- Les échanges réseau

Les accès à la base de données
- Nombre excessif de requêtes
- Mauvaise configuration de la persistance
- Lectures de données importantes et/ou superflues
- Configuration du pool de connexions

La gestion de la mémoire
- L'organisation de la mémoire
- Le ramasse-miettes
- Le cycle de vie des objets
- Configurer la mémoire et le GC
- Mesure de la mémoire
- Off Heap
- La gestion de la mémoire et les performances
- Les options de la JVM HotSpot pour le suivi de l'activité du GC
- Fuite de mémoire
- Fuite de ressources

Optimiser l'utilisation de la mémoire
- Optimiser l'occupation mémoire
- Optimiser les instanciations

Parallélisation
- Les apports de la parallélisation
- La mise en œuvre
- Les API
- Le framework Fork/Join
- Race condition et contention
- Les deadlocks
- Obtenir et exploiter un threaddump
- Les streams parallèles

Les légendes urbaines
- Java est lent
- La concaténation de chaînes
- Augmenter la taille du heap
- Le cache comme LA solution
- 64 bits vs 32 bits

L'importance de la JVM et de la version de Java
- Choisir la JVM
- Configurer la JVM
- La version de Java utilisée

La performance n'est pas que technique
- Les raisons non techniques des mauvaises performances
- Prise en compte dans le SDLC (Software Development Live cycle)
- Mesurer la performance en continue
- Monitoring

Modalité
- Stage pratique en présentiel
- Stage pratique en distanciel
- Nombre de stagiaires minimum : 4
- Nombre de stagiaires maximum : 10

Méthodes pédagogiques
- Exposés
- Cas pratiques
- Echanges d'expérience

Profils des intervenants
- Toutes nos formations sont animées par des consultants-formateurs expérimentés et reconnus par leurs pairs.

Modalités d'évaluation
- Evaluation des acquis de la formation par le biais de cas pratiques et/ou mises en situation.
- Attestation de formation remise à chaque participant.

Démarche qualité

- Questionnaire d'évaluation de satisfaction à chaud complété par chaque participant à l'issue de la formation.

Moyens pédagogiques

- Salle équipée de PC (1 poste par stagiaire), vidéo-projecteur.
- Espace de pause.

Dernière mise à jour le 23/10/2020