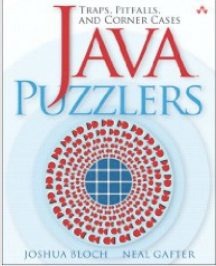




zenika  
ARCHITECTURE INFORMATIQUE

www.zenika.com

Repartez avec le livre



Durée  
4 jours

Répartition  
50% théorie  
50% pratique

Pré-requis  
Connaissance de Java

Public  
Architecte  
Développeur  
Chef de projet

Tarif (déjeuners inclus)  
1990 € (HT)  
Tarif (1 mois avant)  
1790 € (HT)

Lieu  
Paris 9ème

Sessions 2009  
10 au 13 février  
21 au 24 avril  
2 au 5 juin  
5 au 8 octobre  
3 au 6 novembre

Intra-entreprise sur  
demande

Inscription et  
renseignements  
+33(0)1.45.26.19.15  
training@zenika.com  
www.zenika.com

# Java avancé

## Approfondir ses connaissances en Java

### Objectifs

- Pousser plus avant la maîtrise du langage Java
- Comprendre et maîtriser les architectures des applications concurrentes
- Maîtriser les API de communication entre applications Java

### Contenu

#### Types paramétrés (« Generics »)

- Présentation et avantages
- Notion d'effacement de type
- Types bornés et indéfinis
- Comprendre les API et collections paramétrées
- Développer des classes et méthodes paramétrées
- Limitations

#### Références

- Comprendre le Garbage Collector
- Référencement des objets et consommation mémoire
- Hard, Soft, Weak et Phantom References
- En pratique : WeakHashMap

#### Réflexion

- Analyser une classe
- Accéder aux champs et méthodes
- Créer une nouvelle instance
- Cas particuliers : génériques, tableaux et enums
- Proxies dynamiques

#### Classloaders

- Définition et hiérarchie des classloaders
- Quand une classe est-elle chargée ?
- Fonctionnement interne : findclass(), defineClass() et loadClasses()
- Implémenter un classloader personnalisé

#### Maîtriser les Threads

- Notion de Thread et cycle de vie
- Lancer un thread : Thread et Runnable
- Arrêter proprement un thread
- Interruption prématurée du traitement
- Threads démons, priorité, groupes de threads
- Variables Threadlocal

#### Thread Safety

- Notion de classe « thread-safe », immutabilité
- Synchronisation, volatilité et visibilité mémoire
- Variables atomiques et collections synchronisées

#### Architectures concurrentes

- Synchronisation des threads avec wait, notify et join
- Le framework Executor
- Queues, Latches, Barrières et Sémaphores
- Design patterns concurrents

#### Sérialisation

- Rappels sur la sérialisation
- Serializable et Externalizable
- Sérialisation et désérialisation personnalisées

#### NIO

- Comparaison avec java.io
- Comprendre les Buffers
- Manipuler des données volumineuses avec le mapping en mémoire
- Opérations asynchrones avec les Channels et les Selectors

#### RMI

- Architecture générale
- Définir une interface de service
- Développer un service
- Le registre RMI