

## Hibernate Avancé

Maîtriser les subtilités d'Hibernate pour un mapping objet/relationnel adapté et maintenable

La mise en place d'une couche de mapping objet-relationnel entre un modèle métier « objet » et une base de données relationnelle facilite considérablement la problématique de persistance d'une application.

Cependant, les deux modèles de données et de programmation sous-jacente n'ont rien à voir, ce qui occasionne des difficultés connues sous le terme d'« impedance mismatch ». Ces difficultés sont visibles lorsqu'une application Java devient peu performante ou instable.

Avant d'en arriver là, la connaissance précise de l'outil de mapping et le respect de ses règles de développement sont impératives. Réaliser un mapping correctement adapté à ses besoins reste une étape très délicate qui peut nécessiter plusieurs ajustements de sa stratégie de persistance afin de trouver le bon compromis entre performances, stabilité et impédance faible, en fonction du contexte applicatif.

Hibernate offre un large panel d'options de configuration et de représentation ainsi que des fonctionnalités spécifiques qui s'inscrivent dans cette recherche d'optimisation.

Ce cours permettra aux stagiaires de bien comprendre et d'expérimenter les avantages et inconvénients de chaque solution offerte pour choisir le compromis adapté face à certaines situations récurrentes.

### Détails

- Code : JP-HIB2
- Durée : 2 jours ( 14 heures )

#### Public

- Architectes
- Chefs de projets
- Ingénieurs

#### Pré-requis

- Bonne pratique de Java

### Objectifs

- Connaître et assimiler les problématiques classiques de performance liées au mapping objet-relationnel
- Maîtriser les concepts avancés d'Hibernate liés à la performance
- Maîtriser les bonnes pratiques de développement Hibernate
- Savoir utiliser les différents caches Hibernate.

### Programme

#### Objectifs

- Mapping Objet-Relationnel et « impedance mismatch »
- Problématiques liées au chargement des données
- Le lazy loading ou chargement par nécessité
- Notion et utilisation de Proxy
- Les stratégies de fetch

#### Utilisation des caches d'Hibernate

- Le cache de session
- Le cache de second niveau
- Le cache mapping
- Les stratégies de cache
- Avantages et inconvénients des différentes implémentations
- Le cache de requête

#### Partage des données

- Problématiques liées à la concurrence d'accès

- Verrouillage optimiste/pessimiste
- Clustering
- JBossTrecache

#### Optimisation des associations

- Cas des associations bi-directionnelles
- Gestion de l'attribut inverse
- Associations polymorphes

#### Problématiques liées à l'héritage

- Une table par hiérarchie
- Une table par sous-classe
- Une table par classe concrète

#### Monitoring des performances

- Suivi d'une SessionFactory
- Métriques

## Modalités

- **Type d'action** :Acquisition des connaissances
- **Moyens de la formation** :Formation présentielle - 1 poste par stagiaire - 1 vidéo projecteur - Support de cours fourni à chaque stagiaire
- **Modalités pédagogiques** :Exposés - Cas pratiques - Synthèse
- **Validation** :Exercices de validation - Attestation de stages