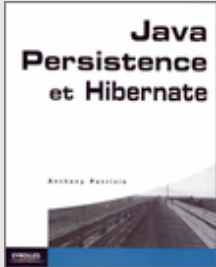




zenika
ARCHITECTURE INFORMATIQUE

www.zenika.com

Repartez avec le livre



Durée
3 jours

Répartition
50% théorie
50% pratique

Pré-requis
Connaissance de Java

Public
Architecte
Développeur
Chef de projet

Tarif (déjeuners inclus)
1490 € (HT)
Tarif (1 mois avant)
1290 € (HT)

Lieu
Paris 9ème

Sessions 2009
12 au 14 janvier
16 au 18 février
16 au 18 mars
27 au 29 avril
18 au 20 mai
22 au 24 juin
15 au 17 juillet
9 au 11 septembre
12 au 14 octobre
2 au 4 novembre
7 au 9 décembre

Intra-entreprise sur
demande

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

Hibernate

Implémenter la couche de persistance d'une application Java/JEE avec Hibernate

Contexte

L'intervention réussie sur plusieurs projets ayant souffert de problèmes de performance procure aux consultants Zenika une expérience précieuse sur ce sujet, concrétisée par la mise à disposition de la librairie [ZenTracker](#) sous licence LGPL permettant de mesurer l'activité sous-jacente d'Hibernate.

Objectifs

- Assimiler les concepts du mapping objet / relationnel
- Savoir maîtriser les principales fonctionnalités d'Hibernate
- Réaliser la couche de persistance d'une application JEE
- Acquérir les bonnes pratiques de développement et d'architecture

Contenu

Introduction

- La persistance transparente grâce aux outils de Mapping Objet/Relationnel (ORM)
- Comparaison avec les techniques JDBC standard en terme de coûts et de fiabilité

Mise en place d'Hibernate

- Description du packaging et des dépendances vers d'autres jars
- Présentation de l'installation et des différentes techniques de configuration
- Configuration en environnement JEE
- Mise en place de l'outil Hibernate Tool

La configuration

- Les fichiers hbm.xml, ou le pont entre le monde objet et le monde relationnel
- Mise en place des annotations
- L'identité : garant de la correspondance entre instances et enregistrements en base de données
- Importance et utilisation des méthodes equals() et hashCode()
- Quelle stratégie de génération d'identifiants

Manipulation de POJO

- Il ne s'agit plus d'INSERT ou d'UPDATE, mais de cycle de vie d'un objet
- Présentation de la SessionFactory et de la Session
- Transiant, Persistant, Détaché, Entité, Valeur ou le vocabulaire Hibernate
- La démarcation transactionnelle ou la garantie d'opérations ACID
- Les opérations CRUD, les premières interactions simples avec la base de données

Relations entre entités

- Il ne s'agit plus d'un objet, mais d'un graphe d'objets liés par des relations
- Les 3 relations du monde objet : 1-n, n-m et 1-1 et leur correspondance en terme relationnel
- Présentation des différents types de collection (set, bag, map, list, array) et des critères permettant de choisir
- Choix du sens de la relation et du type (Uni-directionnel VS bi-directionnel)
 - Correspondance avec le code Java
- L'attribut cascade et les précautions à prendre

Héritage

- Présentation des stratégies de mapping pour la gestion de l'héritage
- Avantages et inconvénients de chaque solution

Les composants

- Notion de composants en opposition aux entités
- Les composants simples
- La collection de valeurs et de composants

Les requêtes

- Présentation des API de requêtes (Criteria, SQL, Exemple, HQL)
- Pertinence et choix de l'API la plus adéquate
- Ajout de contraintes, de tris, d'ordres
- Parcours de relations, jointures implicites et jointures ouvertes
- Les requêtes scalaires pour optimiser les performances
- Externaliser une requête et choix d'une technique de binding de paramètres
- Comment utiliser les spécificités d'une base de données (cas du « connect by prior » d'Oracle)

Optimisation

- Les mises à jour groupées
- Initialisation paresseuse ou le chargement à la demande
- Modification dynamique de la stratégie de fetching définie au niveau du mapping
- Utilisation des caches de niveau 1, de niveau 2 et de requêtes
- Quel cache est mis à jour et par quelle action ?

Architecture

- La couche de persistance et le pattern DAO
- Intégration dans une architecture Web
- Gestion des sessions Hibernate et transactions
 - Le pattern OpenSessionInViewFilter
 - Intégration avec Spring