

Formation Développement d'EJB 3 en Java EE



La technologie Enterprise JavaBeans (EJB) est une architecture de composants logiciels côté serveur pour la plateforme de développement Java EE. La nouvelle spécification EJB3 simplifie la mise en oeuvre des composants distribués Java : en effet depuis la version 3.0, le modèle EJB utilise le principe d'annotation java (meta-données) pour spécifier toute la configuration et les propriétés transactionnelles de l'objet

Objectifs

- Découvrir les bases de la technologie EJB3
- Connaître les différents types d'EJB (entité, session et MDB)
- Comprendre quels sont les apports des EJB 3.x
- Être capable de gérer des transactions
- Disposer des meilleures pratiques de migration des EJB 2.x vers les EJB3.x

Public concerné

- Développeurs Java

Pré requis

- Avoir suivi la formation : "Découvrir l'objet et programmer en Java" (OB300) ou connaissances équivalentes

Une formation de 4 jours

Caractéristiques
Tarif : 1960 € HT par personne
Numéro de formateur : 11753687675
Nombre d'heures : 28
Référence : OB377
Contact : Patrick LE GOFF
Telephone : 01.76.60.66.10
Email : contact@kaptive.com

Paris
28/03/2011
04/07/2011
26/09/2011
19/12/2011

Description des modules

num	Module
1	Introduction
Détails	<ul style="list-style-type: none"> - Nouveautés de Java 1.5 (annotations...) - Architectures à objets distribués, architectures n-tiers, modèle en couches - Les offres de marchés (JBoss, WebSphere...) - Présentation de la plate-forme Java EE
2	L'architecture EJB 3
Détails	<ul style="list-style-type: none"> - La spécification EJB 3 - Architecture : la classe Bean, les interfaces Home et Remote, le descripteur de déploiement XML - Les différents types d'EJB (Session, entité et message-driven-bean) - Le rôle du conteneur d'EJB - Présentation de RMI/IIOP, protocole d'accès aux EJB - Principe d'un service de nommage : exemple avec la registry RMI - JNDI (Java Naming and Directory Interface): fonctionnement, applications, API - Localisation des EJB avec JNDI - Principe et rôle des intercepteurs
3	Persistance de données avec les EJB "entités"
Détails	<ul style="list-style-type: none"> - Les framework ORM (Object Relational mapping) - Les bases du modèle de mapping objet-relationnel - Les identifiants : clé primaire simple ou composite - Le nouveau modèle de persistance EJB3 et ses API JPA - Le gestionnaire de cycle de vie des objets : l'Entity Manager - Opérations sur les EJB entités - Cycle de vie d'un EJB entité - Méthodes " callback " et listeners - Les accès concurrents sur les EJB entités - Le langage EJB-QL : syntaxe et utilisation
4	Création de services avec les EJB "sessions"
Détails	<ul style="list-style-type: none"> - Comparaison EJB sessions stateless et stateful - L'EJB session du type stateless, son cycle de vie - LEJB session du type stateful, son cycle de vie : mécanismes de passivation et d'activation - Les accès concurrents sur les EJB session - Rôle du contexte d'un EJB session - Rôle des objets "Handle" sur les EJB session stateful - Le bon usage des EJB du type session
5	Traitement de messages avec les EJB "message-driven"
Détails	<ul style="list-style-type: none"> - Principe des MOM (Middleware Orientés Message) - L'API JMS (Java Message Service) - Modèles de messages Abonnement/Diffusion et Point-A-Point - Les destinations : Queue et Topic - Principe, mise en oeuvre et cycle de vie d'un EJB message-driven - Le client d'un EJB message-driven - Configuration des objets JMS administrés - La garantie d'acheminement (Transaction et stockage des messages)
6	Sécuriser l'accès aux EJB
Détails	<ul style="list-style-type: none"> - Principe de JAAS (Java Authentification and Autorisation Service) - Les rôles, les groupes de permissions et les permissions sur les méthodes - Mise en oeuvre de la sécurité sur l'accès aux méthodes d'un EJB - Sécurité déclarative ou programmée
7	Les meilleures pratiques EJB

- Détails** - Exposé des bonnes pratiques pour la mise en oeuvre des EJB
- Présentation de quelques design patterns : Transfer Object, Data Access Object, Session Façade, Composite Entity