Programme détaillé
formation 'Architecture JAVA Entreprise (JEE)'- 5j

Objectifs : fournir connaissances théoriques et pratiques permettant d'être un leader technique Architecte Java

Audience : Développeurs expérimentés (4 ans +) .

Prérequis : pratique développement java/j2ee depuis au moins 3 ans.

Moyens pédagogiques : 1 ordinateur/stagiaire. Supports cours. Travaux pratiques. Vidéoprojecteur. Tests

Durée : 5 jours (total 35 h)

Vous allez apprendre à

- concevoir un socle technique Java/J2ee pour vos architectures en couche
- comprendre l'injection de dépendances et les produits associés (spring/Guice/Weld)
- comprendre la programmation aspects et les produits associés (SpringAOP/AspectJ)
- choisir entre une couche service synchrone (WS/EJB/Pojo) ou asynchrone (MOM)
- mettre en oeuvre une architecture asynchone via Spring JMS / EJB3
- créer votre couche d'accès aux données (SGBDR/NoSQL) basée sur meilleures pratiques
- faire le bon choix entre framework d'intégration (camel/spring integ) et ESB (Fuse/Mule)
- mettre en oeuvre des patterns de conception spécifiques JEE
- comprendre technologies JMX / MBeans pour développeur / admin
- sécuriser une application J2EE avec JAAS , fichier .policy et certificats
- comprendre les stratégies de monitoring et tuning performances
- comprendre les différentes briques d'une chaîne d'intégration continue
- mettre en oeuvre une stratégie de haute disponibilité (horizontal/vertical) .

Programme détaillé

Programmation Aspects

- Principes et valeur ajoutée
- Différences avec la POO
- Tisseurs : Spring AOP, AspectJ, Jboss AOP
- Tissage à la compilation
- Tissage à l'exécution
- Tissage et serveur d'application
- Mise en oeuvre Audit

Injection de dépendances

- Présentation du concept
- Problèmes du couplage fort
- Notion de 'Reverse JNDI'
- Role du conteneur léger
- Injection avec Spring
- Injection avec Google Guice
- Injection avec Weld
- @Autowire / @Inject

Architecture en couche
- Valeur ajoutée : conception, maintenance
- Interfaces entre les couches
- couche Service : synchrone ou asynchrone
- Mise en oeuvre avec Spring
- Quel choix ? pourquoi ?

Couche d'accès aux données (DAO)
- Interfaces générique et classes Abstraites
- Persistence relationnelle (sgbdr)
- Persistence non relationnelle (NoSQL)
- Les 4 types de bases NoSQL
- Spécification JPA et Implémentations
- Spring Data : CRUDRepository
- Bonne pratique : Spring Data jpa
- Quel choix ? pourquoi ?

Architecture JAVA JEE
- Spécifications JEE 6 et livrables
- Approche logicielle : architecture en couches
- Problématiques : sécurité + transactions
- Problématiques : haute disponibilité + performance
- Choix frameworks : innovations ou spécifications ?
- Choix d'intégration : synchrone ou asynchrone ?
- Choix présentation : Java ou javascript ?
- Rôle de l'architecte MOE / MOA
- Gérer la gouvernance

Couche Service synchrone
- Couche service synchrone avec POJO/RMI
- Couche service synchrone avec Hessian/Burlap
- Couche service avec EJB3 / Web Services
- Comprendre la différence WebService Soap/Rest
- Comprendre l'orchestration de Web Services
- Quel choix ? pourquoi ?

Couche Service asynchrone
- Pensez MOM !
- Spécification et API JMS
- Fournisseurs de messages JMS : standalone ou dans serveur Jee ?
- Couche service asynchrone avec EJB Message
- Couche service asynchrone avec Spring JMS
- Quel choix ? pourquoi ?

Couche Client
- Clients web Java : lequel choisir : Spring MVC, JSF, Struts2, Wicket ?
- Client JavaScript : lequel choisir : ExtJS, GWT, jQuery ?
- Les API de présentation (AWT, SWING, SWT)
- Problématiques déploiement d'un client riche
- Java WebStart ou Eclipse RCP

Web Services
- Web Service SOAP
- Web Services REST
- Analyse de trames
- Comparaison SOAP / REST
- Test de conformité d'un Web Service
- Mise en oeuvre framework Apache CXF
- Gouvernance et qualité de service
- Orchestration de WS avec BPEL

Les services techniques Java EE
- Annuaire JNDI : ressources DB, JMS, EJB,
- Middleware JMS
- Sécurité JAAS,
- Persistance JPA,
- Transaction JTA
- Transactions distribuées (2PC)

EJB 3
- Services offerts par le conteneur
- Opposition EJB / SPRING
- Bonnes pratiques EJB session
- EJB et Web Services

XML
- Définition et utilisations
- Complémentarité avec Java
- Les API standards comme JAXP
- Comprendre JAXB, JAXP
- SAX, DOM, STAX
- Intégration basée sur flux XML

Sécurité
- fichier java.policy
- API JAAS : Realms
- Sécuriser les composants EJB
- Sécuriser les applications Web
- @Interceptor EJB
- @Secured Spring
- Filtres Spring Security

Monitoring JMX
- Spécification JMX
- 3 couches
- Instrumentation
- MBean Server
- Client JMX
- jvisualvm
- Mise en oeuvre

Performances JAVA
- Méthodologie de tuning
- Analyse performances
- Outils : jconsole, jvisualvm
- Fonctionnement mémoire Java
- Garbage Collector
- paramètres JAVA_OPTS
- Tests performance avec JMeter

Tests
- Tests fonctionnel web avec SELENIUM
- Tests unitaires avec JUnit / TESTNG
- Tests d'intégration avec Spring

Haute disponibilité
- Load balancing avec mod_jk
- Reprise sur incident (Fail over)
- Réplication de session
- Affinité de session (sticky session)
- Cache niveau 2 des Entités JPA
- Multicast IP / Jgroups

Intégration continue

- Briques clés projet intégration continue
- Maven : Build projet multi modules
- Maven : les rapports qualité
- Déploiement dans Nexus (Snapshots/Releases)
- Installation et configuration Jenkins/Hudson
- Lancement Builds Jenkins
- Plugins Jenkins

Intégration Java / SI

- Présentation SOA / ESB
- Critères de choix d'un ESB
- Frameworks d'intégration = ESB 'light'
- EIP Patterns : Camel ou Spring Intégration ?
- Urbanisme et intégration
- SOA et Web Services
- Orchestration : BPEL
- Processus métiers et BAM

Osgi

- Contexte d'utilisation
- Spécifications
- Implémentation : Felix
- Moteur de services
- Création composant OSGI
- Fichier MANIFEST.MF
- Serveur Glassfish
- Apache Karaf