Java EE et Architecture applicative

Plan du cours

1. Présentation de la plateforme J2EE
2. Les API de J2EE
3. Java EE 7.0
4. Les implémentations de Java EE 7.0
5. EJB
6. Les Annotations
7. JPA
8. Spring
9. Les Design Pattern : Singleton, Facade, IOC, AOP et le Value Objet.

La présentation de J2EE (1/3)

- J2EE est une plate-forme **fortement orientée serveur** pour le développement et l'exécution d'applications distribuées. Elle est composée de deux parties essentielles :
  - un ensemble de spécifications pour une infrastructure dans laquelle s'exécutent les composants écrits en Java : un tel environnement se nomme **serveur d'applications**.
  - un ensemble d'API qui peuvent être obtenues et utilisées séparément. Pour être utilisées, certaines nécessitent une implémentation de la part d'un fournisseur tiers.

La présentation de J2EE (2/3)

L'utilisation de J2EE pour développer et exécuter une application offre plusieurs avantages :

- une architecture d'applications basée sur **les composants** qui permet un découpage de l'application et donc **une séparation des rôles** lors du développement
- la possibilité **de s'interfacer** avec le système d'information existant grâce à de nombreuses API : JDBC, JNDI, JMS, JCA ...
- la possibilité **de choisir** les outils de développement et le ou les serveurs d'applications utilisés qu'ils soient commerciaux ou libres.

La présentation de J2EE (3/3)

J2EE permet **une grande flexibilité** dans **le choix de l'architecture** de l'application en combinant les différents composants. Ce choix dépend des besoins auxquels doit répondre l'application mais aussi des compétences dans les différentes API de J2EE. L'architecture d'une application se découpe idéalement en au moins trois tiers :

- **la partie cliente (couche Présentation)** : c'est la partie qui permet le dialogue avec l'utilisateur. Elle peut être composée d'une application standalone, d'une application web ou d'applets.
- **la partie métier (couche Métier ou Service)** : c'est la partie qui encapsule les traitements (dans des EJB ou des JavaBeans)
- **la partie données (couche DAO)** : c'est la partie qui stocke les données

Les API de J2EE

API	version de l'API dans					
	1.2	1.3	1.4	5	6	7
Entreprise Java Bean (EJB)	1.1	2.0	2.1	3.0	3.1	3.2
Remote Method Invocation (RMI) et RMI-IIOP	1.0					
Java Naming and Directory Interface (JNDI)	1.2	1.2	1.2.1			
Java Database Connectivity (JDBC)	2.0	2.0	3.0			
Java Transaction API (JTA) Java Transaction Service (JTS)	1.0	1.0	1.0	1.1	1.1	1.2
Java Messaging service (JMS)	1.0	1.0	1.1	1.1	1.1	2.0
Servlets	2.2	2.3	2.4	2.5	3.0	3.1
Java Server Pages (JSP)	1.1	1.2	2.0	2.1	2.2	2.2
Java Server Faces (JSF)				1.2	2.0	2.2
Expression Language (EL)					2.2	3.0
Java Server Pages Standard Tag Libray (JSTL)				1.2	1.2	1.2

Les API de J2EE

API	version de l'API dans					
	1.2	1.3	1.4	5	6	7
JavaMail	1.1	1.2	1.3	1.4	1.4	1.4
J2EE Connector Architecture (JCA)		1.0	1.5	1.5	1.6	1.6
Java API for XML Parsing (JAXP)		1.1	1.2			
Java Authentication and Authorization Service (JAAS)		1.0				
JavaBeans Activation Framework		1.0.2	1.0.2			
Java API for XML-based RPC (JAXP-RPC)			1.1	1.1	1.1	1.1
SOAP with Attachments API for Java (SAAJ)			1.2	1.3		
Java API for XML Registries (JAXR)			1.0	1.0	1.0	1.0
Java Management Extensions (JMX)			1.2			
Java Authorization Service Provider Contract for Containers (JACC)			1.0	1.1	1.4	1.4
Java API for XML-Based Web Services (JAX-WS)				2.0	2.2	2.2

Les API de J2EE

API	version de l'API dans					
	1.2	1.3	1.4	5	6	7
Java Architecture for XML Binding (JAXB)				2.0	2.2	
Streaming API for XML (StAX)				1.0		
Java Persistence API (JPA)				1.0	2.0	2.1
Java API for RESTful Web Services (JAX-RS)					1.1	2.0
Web Services				1.2	1.3	1.3
Web Services Metadata for the Java Platform					2.1	2.1
Java APIs for XML Messaging (JAXM)					1.3	
Context and Dependency Injection (CDI)					1.0	1.1
Dependancy Injection (DI)					1.0	
Bean Validation					1.0	1.1
Managed beans					1.0	1.0

Les API de J2EE

API	version de l'API dans					
	1.2	1.3	1.4	5	6	7
Interceptors					1.1	1.1
Common Annotations					1.1	1.1
Java API for WebSocket						1.0
Java API for JSON						1.0
Concurrency Utilities for Java EE						1.0
Batch Applications for the Java Platform						1.0

- Ces API peuvent être regroupées en trois grandes catégories :
  - les composants : Servlet, JSP, JSF, EJB
  - les services : JDBC, JTA/JTS, JNDI, JCA, JAAS
  - la communication : RMI-IIOP, JMS, Java Mail

L'environnement d'exécution des applications J2EE

- J2EE propose des spécifications pour une infrastructure dans laquelle s'exécutent les composants. Ces spécifications décrivent les rôles de chaque élément et précisent un ensemble d'interfaces pour permettre à chacun de ces éléments de communiquer.

- Ceci permet de séparer les applications et l'environnement dans lequel elles s'exécutent. Les spécifications précisent à l'aide des API un certain nombre de fonctionnalités que doit implémenter l'environnement d'exécution. Ces fonctionnalités permettent aux développeurs de se concentrer sur la logique métier.

- Pour exécuter ces composants de natures différentes, J2EE définit des conteneurs pour chacun d'eux. Il définit pour chaque composant des interfaces qui leur permettront de dialoguer avec les composants lors de leur exécution. Les conteneurs permettent aux applications d'accéder aux ressources et aux services en utilisant les API.

Les conteneurs (1/2)

- Les conteneurs assurent la gestion du cycle de vie des composants qui s'exécutent en eux. Les conteneurs fournissent des services qui peuvent être utilisés par les applications lors de leur exécution.

- Il existe trois (03) conteneurs définis par J2EE:
  - **conteneur web** : pour exécuter les servlets, les JSP et les JSF.
  - **conteneur d'EJB** : pour exécuter les EJB.
  - **conteneur client** : pour exécuter des applications standalone sur les postes qui utilisent des composants J2EE.

- Pour déployer une application dans un conteneur, il faut lui fournir deux éléments :
  - l'application avec tous les composants (classes compilées, ressources ...) regroupées dans une archive ou module. Chaque conteneur possède son propre format d'archive.
  - un fichier descripteur de déploiement contenu dans le module qui précise au conteneur des options pour exécuter l'application.

Les conteneurs (2/2)

- Il existe trois types d'archives :

Archive / module	Contenu	Extension	Descripteur de déploiement
bibliothèque	Regroupe des classes	jar (Java ARchive)	
application client	Regroupe les ressources nécessaires à leur exécution (classes, bibliothèques, images, ...)	jar	application-client.jar
web	Regroupe les Servlets, les JSP et les JSF ainsi que les ressources nécessaires à leur exécution (classes, bibliothèques de balises, images, ...)	War (Web ARchive)	web.xml
EJB	Regroupe les EJB et leurs composants (classes)	jar	ejb-jar.xml
Application J2EE	Regroupe un ou plusieurs modules	ear (Entreprise ARchive)	application.xml

Les services proposés par la plate-forme J2EE

- Une plate-forme d'exécution J2EE complète implémentée dans un serveur d'applications propose les services suivants :
  - service de nommage (naming service)
  - service de déploiement (deployment service)
  - service de gestion des transactions (transaction service)
  - service de sécurité (security service)

- Ces services sont utilisés directement ou indirectement par les conteneurs mais aussi par les composants qui s'exécutent dans les conteneurs grâce à leurs API respectives.

J2EE historique

J2EE 1.4 (Diffusée en novembre 2003)

- Le support des services web (nouveau)
- J2EE deployment API 1.1 (nouveau)
- J2EE management API 1.0 (nouveau)
- Java ACC : Java Authorization Contract for Container (nouveau)
- EJB 2.1 (update)

Java EE 5.0 (Diffusée en 2007)

- Les annotations
- JSF
- JPA
- EJB 3.0
- Le support des dernières versions des API concernant les services web et permettant une mise en œuvre d'une architecture de type SOA.

Java EE 6.0 (Diffusée en décembre 2009)

- Plus riche : de nouvelles spécifications sont ajoutées et les spécifications majeures sont enrichies
- Plus facile : généralisation de l'utilisation des POJO et des annotations notamment dans le tiers web
- Plus légère : création de la notion de **profiles** et de **pruning**, EJB Lite
- Toujours aussi robuste après 10 ans d'existence.
- L'implémentation de référence est le projet open source GlassFish version 3.

La notion de Profile

- Java EE 6 définit la notion de profile qui est un sous-ensemble et/ou un sur-ensemble de Java EE 6 pour des besoins particuliers.

- Les profiles sont conçus pour proposer une solution technique particulière pour des besoins spécifiques. Cela permet à un fournisseur de n'avoir à proposer que le support des technologies incluses dans le profile plutôt que d'avoir à implémenter toutes celles de la plate-forme Java EE.

- Java EE 6 définit un premier profile : le web profile qui se concentre sur le développement d'applications de type web. Il doit pouvoir être exécuté dans un simple conteneur web car le packaging se fait dans une archive de type war.

- Le web profile est composé de plusieurs spécifications pour définir un sous-ensemble de Java EE : Servlet 3.0, JSP 2.2, EL 1.2, JSTL 1.2, JSF 2.0, EJB Lite 3.1, JTA 1.1, JPA 2.0, Bean Validation 1.0, DI 1.0, CDI 1.0 et Interceptors 1.1.

La notion de Pruning (1/2)

- La plate-forme Java EE est devenue au fil des versions imposante en terme de nombre de spécifications, de fonctionnalités et d'API. Aucunes des précédentes versions n'a supprimé de fonctionnalités même si certaines sont obsolètes car remplacées, rarement adoptées ou utilisées. Cela pose plusieurs problèmes :
  - Pour le développeur : la taille du SDK augmente avec le nombre d'API
  - Pour les fournisseurs de serveurs d'applications : l'obligation de support pour ces fonctionnalités avec un accroissement de la taille des serveurs, de leur consommation en ressources et de leur temps de démarrage.

- Certaines de ces fonctionnalités sont de plus obsolètes car remplacées par une plus récente ou ne sont que peu ou pas utilisées.

- Java EE 6 entame donc une cure d'amaigrissement de la plate-forme en définissant un ensemble d'API déclarées comme **pruned**.

La notion de Pruning (2/2)

- Les fonctionnalités déclarées **pruned** dans Java EE 6 sont :
  - Entity CMP 2.x : cette API est avantageusement remplacée par JPA
  - JAX-RPC : cette API est avantageusement remplacée par JAX-WS
  - JAX-R : cette API est très peu utilisée car l'utilisation de UDDI n'est pas très répandue
  - JSR 88 (Java EE Application Deployment) : cette API est très peu mise en œuvre par les fournisseurs de serveurs d'applications

- Elles doivent toujours être disponibles dans une implémentation de Java EE 6 mais elles seront amenées à disparaître dans les prochaines versions de la plate-forme Java EE et leur support ne sera plus obligatoire dans les implémentations des serveurs d'applications.

- Le concept de pruned est plus fort que le concept de **deprecated** de la plate-forme Java SE.

La présentation de Java EE 7.0

Java EE 7.0 (1/5)

- Les spécifications de Java EE 7 sont définies dans **la JSR 342**. Cette spécification ne définit pas directement d'API mais elle liste celles qui sont incluses dans la plate-forme et comment celles-ci interagissent entre-elles.

- Elle définit aussi des éléments précis de la plate-forme comme les transactions, la sécurité, l'assemblage, le déploiement, ...

- Java EE 7 a plusieurs objectifs :
  - poursuivre l'amélioration de la productivité et la simplification de l'utilisation de la plate-forme.
  - le support de HTML 5 (WebSocket, Json, HTML5 forms, ...).
  - l'ajout de nouvelles fonctionnalités : Batch API (modèle de programmation pour les applications batch) et Concurrency Utilities API (exécution de traitements asynchrones sans avoir à utiliser JMS).

- Les spécifications de Java EE 7 furent officiellement diffusées **en juin 2013**.

Java EE 7.0 (2/5)

- More annotated POJOs
- Less boilerplate code
- Cohesive integrated platform

- WebSockets
- JSON
- Servlet 3.1 NIO
- REST

- Batch
- Concurrency
- Simplified JMS

Java EE 7.0 (3/5)

- Java EE 7 inclut plusieurs **nouvelles API** :
  - Java API for JSON Processing 1.0 (JSR 353)
  - Java API for WebSocket 1.0 (JSR 356)
  - Batch Applications for the Java Platform 1.0 (JSR 352)
  - Concurrency utilities for Java EE 1.0 (JSR 236)

- Java EE 7 inclut des API ayant **une mise à jour majeure** :
  - Java Message Service 2.0 (JSR 343)
  - Expression Language 3.0 (JSR 341)
  - JAX-RS : Java API for Restful Web Services 2.0 (JSR 339)

- Java EE 7 inclut des API mises à jour :
  - JPA 2.1 (JSR 338)
  - EJB 3.2 (JSR 345)
  - Servlet 3.1 (JSR 340)
  - Java Server Faces 2.2 (JSR 344)
  - Contexts and Dependency Injection for Java EE 1.1 (JSR 346)
  - Bean Validation 1.1 (JSR 349)
  - Interceptors 1.2 (JSR 318)
  - Java EE Connector Architecture 1.7 (JSR 322)

Java EE 7.0 (4/5)

- Les JSR mises à jour (Maintenance Release) sont :
  - Web Services for Java EE 1.4 (JSR 109)
  - Java Authorization Service Provider Contract for Containers 1.5 (JACC 1.5) (JSR 115)
  - Java Authentication Service Provider Interface for Containers 1.1 (JASPIC 1.1) (JSR 196)
  - JavaServer Pages 2.3 (JSR 245)
  - Common Annotations for the Java Platform 1.2 (JSR 250)
  - Java Transaction API 1.2 (JSR 907)
  - JavaMail 1.5 (JSR 919)

- Initialement Java EE 7 devait intégrer la spécification Java Temporary Caching API 1.0 (JSR 107) et des fonctionnalités relatives au Cloud (par exemple State Management (JSR 350)) mais elles sont repoussées dans la prochaine version de Java EE.

Java EE 7.0 (5/5)

- Java EE 7 poursuit la simplification de l'utilisation de la plate-forme entamée depuis Java EE 5 notamment :
  - la nouvelle version 2.0 de l'API JMS
  - des changements dans la configuration : ressources de type fabrique de connexions par défaut pour la base de données et le broker JMS, amélioration dans la déclaration des permissions de sécurité, ...
  - les technologies déclarées pruned dans Java EE 6 sont optionnelles dans Java EE 7 : EJB Entity Beans, JAX-RPC 1.1, JAXR 1.0 et la JSR-88

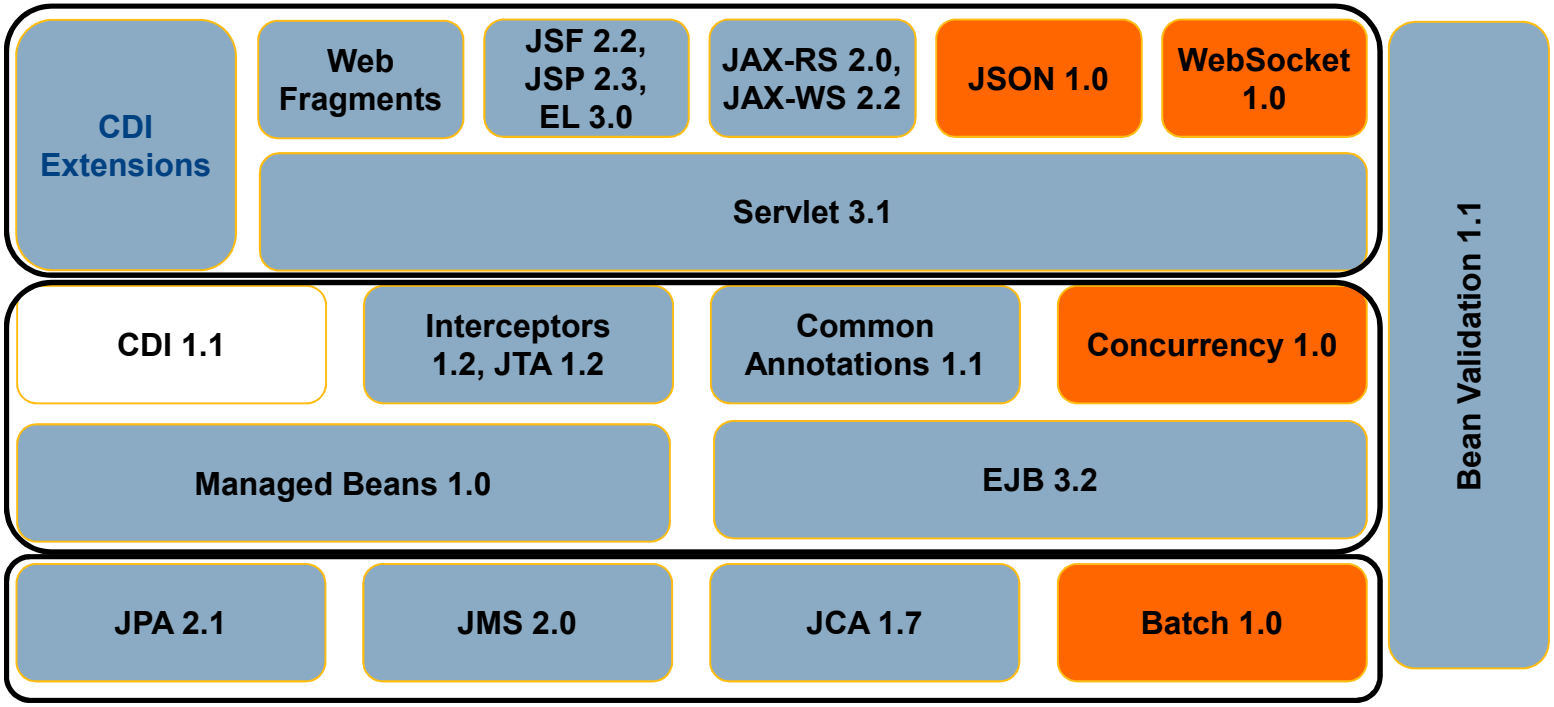
- JAX-RS 2.0 est ajoutée dans le Web Profile.

- L'implémentation de référence est **la version 4.0 du serveur d'applications Glassfish.**

Top Ten Features dans Java EE 7

1. WebSocket client/server endpoints
2. Batch Applications
3. JSON Processing
4. Concurrency Utilities
5. Simplified JMS API
6. @Transactional et @TransactionScoped
7. JAX-RS Client API
8. Default Resources
9. More annotated POJOs
10. Faces Flow

Java EE 7 JSRs



- Les spécifications (JSR) sont disponibles moyennant le lien suivant : https://java.net/projects/javaee-spec/pages/Home

Java EE 7.0

http://www.oracle.com/technetwork/java/javaee/overview/compatibility-jsp-136984.html

Java EE 7.0 : les serveurs

Java EE 7 Full Platform Compatible Implementations

GlassFish Server Open Source Edition 4.0

Tested Configuration

TMAX JEUS 8

Tested Configuration

Wildfly 8.x

Tested Configuration

Cosminexus: Hitachi Application Server v10.0

Tested Configuration

IBM WebSphere Application Server Version 8.5.5.6(Liberty Profile)

Tested Configuration

Oracle Weblogic Server !2.2.1

Tested Configuration

Red Hat JBoss Enterprise Application Platform 7.0

Tested Configuration

IBM WebSphere Application Server Version 9.x

Test Configuration

Java EE 7.0 Web Profile : les serveurs

Java EE 7 Web Profile Compatible Implementations

GlassFish Server Open Source Edition 4.0
Web Profile

Tested Configuration

IBM WebSphere Application Server
Version 8.5.5.6(Liberty Profile)

Tested Configuration

IBM WebSphere Application Server
Version 9.x

Test Configuration

Wildfly 8.x Web Profile

Tested Configuration

Red Hat JBoss Enterprise Application
Platform 7.0

Tested Configuration

Entreprise Java Bean (EJB)