Conception d'applications internet

Introduction

Olivier Cailloux

LAMSADE, Université Paris-Dauphine

12 novembre 2015

LAMSADE
UMR CNRS 7243
laboratoire d'analyse et modélisation de systèmes pour l'aide à la décision

DAUPHINE
UNIVERSITÉ PARIS

PSL
RESEARCH
UNIVERSITY

L'enseignant

- Olivier Cailloux
- olivier.cailloux@dauphine.fr
- Coordonnées : cf. annuaire de Dauphine

Le terme Java

Terme *Java* adopté en 1995 (“as an example of yet another name that would never work”) (source: Java World)

Le terme Java

Terme *Java* adopté en 1995 (“as an example of yet another name that would never work”) (source: Java World)

A jar full of Java beans, please

- JAR File : introduits à la version 1.1. Une collection de fichiers `.class`.
- Java Bean (aussi version 1.1). (specs) : un composant logiciel pour assemblage (par exemple, un bouton AWT, une feuille de calcul à placer dans un document).

Fun fact

Voyons le nombre magique des fichiers `.class`...

Java EE

- Java EE ?

Java EE

- Java EE ? Java Platform, Enterprise Edition

Java EE

- Java EE ? Java Platform, Enterprise Edition
- JCP ?

Java EE

- Java EE ? Java Platform, Enterprise Edition
- JCP ? Java Community Process

Java EE

- Java EE ? Java Platform, Enterprise Edition
- JCP ? Java Community Process
- API ?

Java EE

- Java EE ? Java Platform, Enterprise Edition
- JCP ? Java Community Process
- API ? Application Programming Interface

Java EE

- Java EE ? Java Platform, Enterprise Edition
- JCP ? Java Community Process
- API ? Application Programming Interface

Java EE

- technologies
- Spécifications, dont API
- Implémentation de référence
- Version actuelle : 7

Objectifs pédagogiques, 1

Mise en œuvre des patterns

- Mise en œuvre des patterns dans les spécifications Java EE
- Quand les mettre en œuvre ?
- Applications dans programmes propres

Objectifs pédagogiques, 2

Lecture de spécifications

- Savoir extraire l'information utile des spécifications !
- Standards W3C, spécifications Java EE...

Objectifs pédagogiques, 3

Modélisation

- Réponse à des besoins exprimés vaguement
- Appui sur les standards du web actuels
- Dosage du réalisme et de l'intérêt des fonctionalités

Quels sont *vos* objectifs ?

Intérêt pratique

- Qu'on soit programmeur, qu'on discute avec des programmeurs
- Rendre le travail plus difficile conceptuellement
- Prendre de la hauteur, éviter les tâches répétitives et se concentrer sur le conceptuel
- Respect et compréhension des standards (aperçu de la façon dont ils sont construits) : compétence essentielle
- ... dans de multiples domaines
- Importance des patterns dans de multiples domaines
- Technologie en vogue

Par cours

À l'issue de chaque (?) cours :

- compréhension des bases théoriques d'une technologie (patterns invoqués, liens avec autres technologies)
- capacité de réponse à l'aide de la technologie vue à (au moins) un besoin simple

Projet

- Réponse à un besoin *réel*
- Par groupe
- Pair programming encouragé
- Rapport final
- Recommandé : code en anglais

Forum

- Sur My Course
- Marquer les posts utiles !

Évaluation

Résultat : 0,7 × note projet finale + 0,3 × note concept.

Projet

- Évaluation projet final pondérée par code individuel
- Indiquez (qui commet et) qui est support
- Le rapport peut servir à expliquer des déséquilibres
- Un faible pilotage ne sera pas compensé par un grand support
- Si vous avez testé le pair programming : mieux

Concept

Résumé, tuto, note critique, document explicatif, vidéo, correction de wikipedia, réponses sur stackexchange, sur le forum, code...

Vote pour le meilleur pédagogue et pour le meilleur projet

Prérequis

Programmation Java théorique et appliquée ; ingénieurie logicielle théorique

- Exceptions
- Héritage
- Concept d'API
- Programmation par contrat
- XML
- Maven
- Git

Java EE : Processus

- Java EE fortement appuyée sur standards ouverts
- Standards du W3C / IETF ?

Java EE : Processus

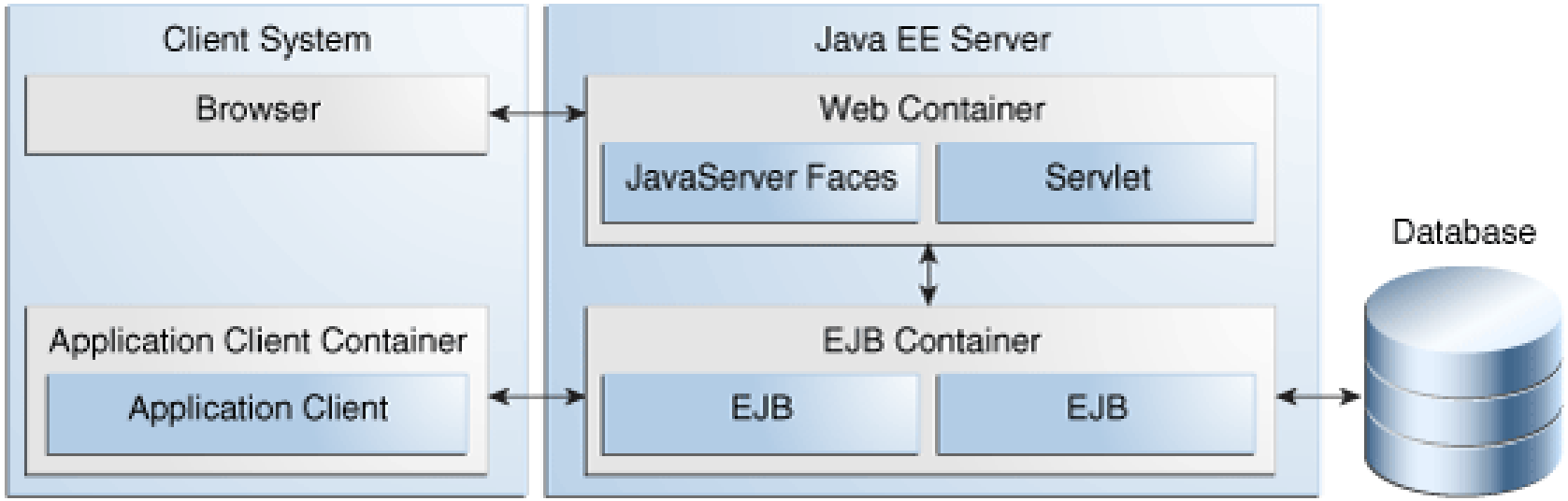
- Java EE fortement appuyée sur standards ouverts
- Standards du W3C / IETF ? HTTP, HTML, XML, WSDL, ...

Java EE : Processus

- Java EE fortement appuyée sur standards ouverts
- Standards du W3C / IETF ? HTTP, HTML, XML, WSDL, ...
- JCP : implication de « la communauté » pour standards Java
- Tentions entre standard ouvert et contrôle ! (2010, Apache quitte le comité JCP ; Doug Lea également, en faveur de OpenJDK...)

Conteneurs

- Un produit conforme Java EE fournit trois *conteneurs*
  - Conteneur EJB
  - Conteneur web
  - Conteneur application client
- Contenant des *composants* (du type adéquat)
- Chacun fournit des services pour le développeur
- Fournit l'accès aux API (différents conteneurs, différentes API)



Composants

- *Composant* : une unité logicielle assemblée dans une application Java EE avec ses classes et fichiers liés et communiquant avec d'autres composants.
- Code Java compilé normalement
- Assemblé dans une application Java EE : peut utiliser les services ; doit se conformer aux spécifications
- Exécution gérée par le conteneur (pas de `main`, par exemple)

Composant EJB

EJB

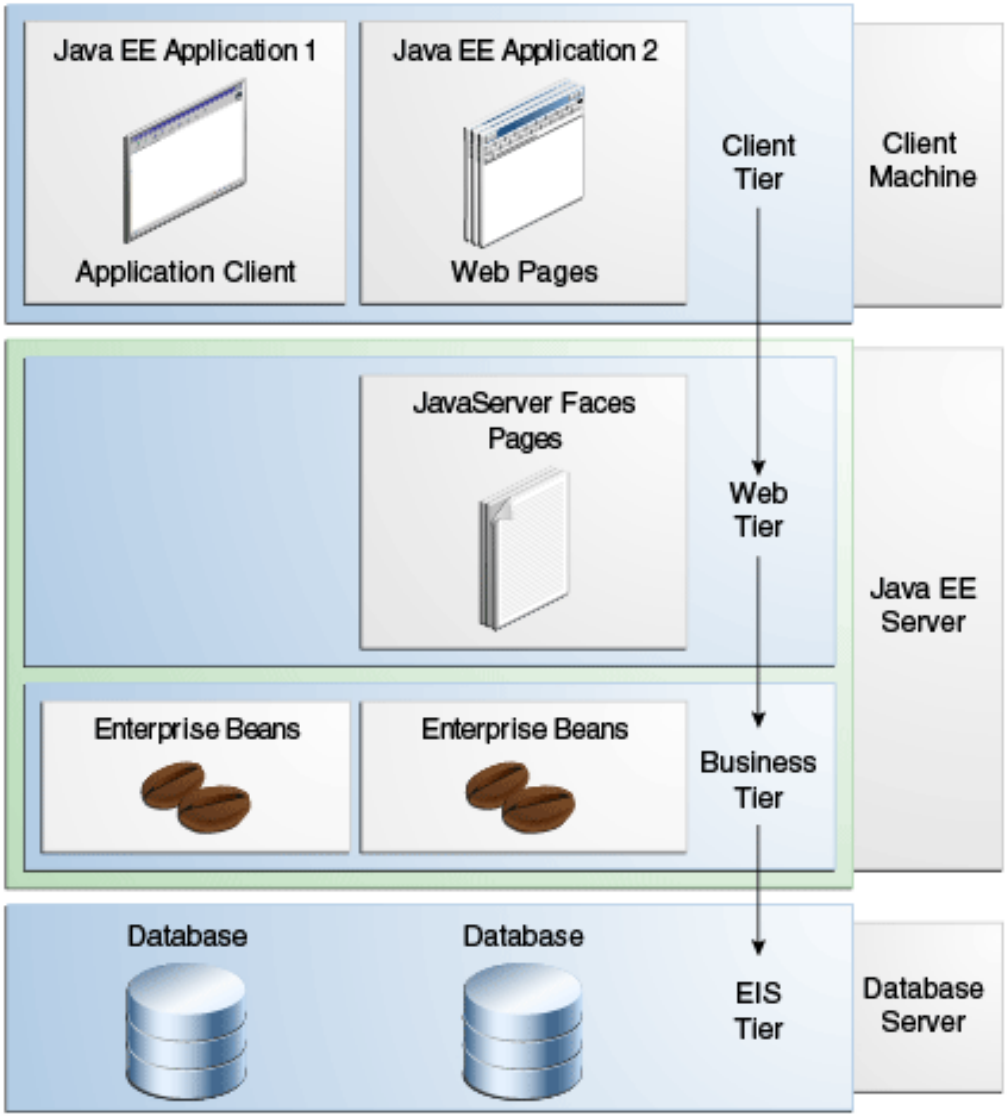
- *Enterprise* Java Bean
- Composant « business », sur le serveur
- Service pouvant être appelé localement ou à distance
- Deux types : session bean, message-driven bean

- Le conteneur rend l'EJB accessible de l'extérieur
- Permet le Remote Method Invocation, sorte de RPC
- Le conteneur instancie, facilite la sérialisation, ...

Composant Web

- Java Servlet
- JavaServer Faces

Couches (« tier »)



- Ajout d'une couche multithread entre le client et le serveur classique
- Souvent : presentation, logic, data tier
- Couche web : peut être également appelé un client web (pourquoi ?).

Deux applications Java EE

BDD ⇔ EJB ⇔ client

- entreprise A : niveau de stock calculable d'après BDD
- EJB : requête pour obtenir le niveau de stock
- composant client (fournisseur de A) : contacte l'EJB

BDD ⇔ EJB ⇔ Servlet

- entreprise A : niveau de stock calculable d'après BDD
- EJB : requête pour obtenir le niveau de stock
- Web : servlet répondant à HTTP GET (client léger)
- Et puis ? Quel client final ?

Options pour client non-java ?

Modules

- *Module* Java EE : fichier archive compressé
- Ensemble de composants pour un même conteneur (typiquement)
- Éventuellement : un descripteur de déploiement (`.xml`) pour ce type de conteneur (standard Java EE ou par produit)
- Éventuellement : des pages HTML statiques ; des classes utilité, ...
- Les descripteurs surchargent les annotations

Module Web

Module Web

- Fichier `.war`
- Fichiers `.class` servlets et autres dans WEB-INF/lib ou WEB-INF/classes
- Fichiers web statiques (.html, images, ...) dans root
- `WEB-INF/web.xml` : descripteur pour conteneur Web (JNDI)
- `META-INF/glassfish-web.xml` : descripteur pour glassfish
- `META-INF/MANIFEST.MF`

Assemblage et déploiement

- Application Java EE composée d'un ou plusieurs modules
- On peut déployer un module seul (`.war`, `.jar`)
- Ou assembler les modules dans un fichier Enterprise Archive (`.ear`)
- EAR : plusieurs modules et év. descripteur d'application (`META-INF/application.xml`, `META-INF/glassfish-application.xml`)

Déploiement

- Procédure dépend du serveur d'application Java EE
- Typiquement : déplacer l'archive (`.war`, `.ear`, `.jar`) dans un répertoire du serveur
- Accès depuis l'environnement de développement via plug-ins

Services

Exemples :

- Managed beans
- CDI
- RestFul
- JSF
- Bean validation
- JAXB, JAX-WS, JNDI (aussi dans Java SE)

GlassFish Server Tools

- Démarrer, arrêter le serveur
- Déployer des paquets
- Application : console d'administration
- Base de données
- wsimport : artéfacts JAX-WS depuis WSDL (etc.)

À vous

- « Installez » GlassFish (copie depuis `/usr/local/glassfish-4.1/glassfish`)
- Démarrez votre serveur (cf. `bin/`, `http://localhost:8080`, `http://localhost:4848`)
- Désactivez l'écoute extérieure
- Lisez les logs

Accès environnement Java EE via Eclipse

- Accès aux bibliothèques Java EE requis
- En Eclipse : possible d'ajouter un environnement Java EE rudimentaire, « J2EE Preview » (Preferences / Server / Runtime Environments)
- Quand un projet vise ce runtime, eclipse ajoute les bibliothèques en dépendances
- Ces bibliothèques ne seront cependant pas exportées dans l'archive (pourquoi ?)

Notions d'HTTP

HTTP ?

Notions d'HTTP

HTTP ?

- Protocole de communication principal du net
- Échange de requêtes HTTP et réponses HTTP
- Accès à une ressource HTTP via URI
- Requête GET (par exemple)
  - Paramètres possibles (dans « query »)
  - `https://www.google.com/maps?q=paris-dauphine&sourceid=Mozilla-search`
- Réponse ?

Notions d'HTTP

HTTP ?

- Protocole de communication principal du net
- Échange de requêtes HTTP et réponses HTTP
- Accès à une ressource HTTP via URI
- Requête GET (par exemple)
  - Paramètres possibles (dans « query »)
  - `https://www.google.com/maps?q=paris-dauphine&sourceid=Mozilla-search`
- Réponse ?
  - En-tête : media-type (liste), encodage, code de statut…
  - Corps : HTML par exemple
- Cf. RFC 723X

Programmation web « bas » niveau

- Ouverture d'un socket pour écriture réseau
- Définition de l'encodage binaire
- Client envoie requête de connexion
- Serveur écoute sur un socket
- Établissement de la connexion
- Gestion des threads
- La communication peut commencer
- Dispatch à la bonne classe
- Gestion des time-outs
- ...

Servlet

En Java EE, le conteneur effectue une partie du travail pour nous

- Le programmeur (côté serveur) indique ce qu'il faut répondre
- Servlet (ici, HTTP) : classe qui traite les requêtes
- Annoter la classe (`javax.servlet.annotation.WebServlet`) et étendre `HttpServlet` ; préciser `urlPatterns` (ou `value`)
- Requête associée à un servlet par le conteneur
  - `http://server/context-root/servlet-path`
  - context-root : associé à un module web en fonction de son nom d'archive (ou dans descripteur non-standard ou dans descripteur standard d'une application Java EE)
  - servlet-path : associé à un servlet en fonction de `urlPatterns`
- Le conteneur gère le cycle de vie du servlet, lui envoie les objets requête et réponse
- Exemple, `doGet` : récupérer une sortie (`getWriter` ; `getOutputStream`) ; écrire les en-têtes ; écrire le corps

À vous : calculons sur le web

- Créer un module web (dynamique ou statique ?)
- Créer un servlet
- Programmer réponse : "ça fait 0"
- Exporter le module dans une archive déployable (extension ?)
- Déployer le module sur le serveur à la main
- Envoyer une requête GET (comment ?) et voir "ça fait 0"
- Se féliciter

En plus

- Installer Glassfish Tools (depuis Eclipse Marketplace)
- Accepter deux paramètres `add1` et `add2`
- Renvoyer "ça fait " et l'addition des paramètres
- Renvoyer une erreur s'il manque un paramètre

Avant-première

Comment faciliter le développement de servlets ?

Avant-première

Comment faciliter le développement de servlets ?

- Décodage facile de paramètres
- Réponse HTML
- Réponse XML

Avant-première

Comment faciliter le développement de servlets ?

- Décodage facile de paramètres
- Réponse HTML
- Réponse XML

Et plus !

- Injection de références
- Accès à des classes (distantes) pour service (EJB)
- Définition de services web Restful, Soap
- Accès aux données, transactions

Projets I

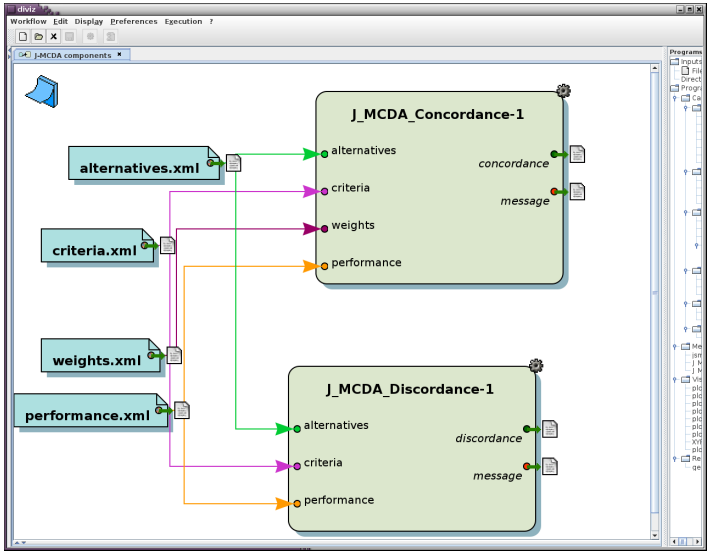
Objectif : un projet utile et non redondant. Voici quelques pistes.

- Gestion musique ou bibliographie collective : votes ; images ...
- Suivi alimentaire (extensions)
- Données de votes et de préférences (whale, Pref Lib) : édition, visualisation, agrégation
- Rendez-vous grâce à synchronisation de calendriers
- Planning des cours : GUI, liens avec calendriers en ligne
- Mise en forme d'articles de blog : imprimer en deux colonnes, transférer sur une liseuse
- Météo : collecte de différentes prédictions ; collecte manuelle ; comparaisons
- Recherche d'emploi / d'appartement fct distance réelle

Projets II

- Centralisation des dons et objets en vente (collecte listes, post nouveau, log réputation, tri par distance)
- Définition de contraintes linéaires en ligne collaborativement
- Refaire site Pôle info 3
- Parcours multi-modal (http://velib.io/) ou statistiques vélib

Façade à Diviz : plate-forme d'agrégation de préférences



À faire

- Lisez (ou redirigez) vos e-mails @ Dauphine (pour les annonces)
- Installer les outils sur votre machine : Eclipse Mars Java EE, Java EE 7 (et Glassfish 4), Java 8 (OpenJDK)
- Choix d'un projet sur MyCourse
- Présentation (5 à 10 min) de vos idées
- Compte GitHub ou Bitbucket et commit initial : diapos

Sondage !

- Avez-vous une machine à amener en cours ?
- Y a-t-il des prérequis qui vont vous manquer ?
- Sur quoi le cours devrait-il porter ?

Licence