

Conception d'applications web en Java

Présentation du cours

Olivier Cailloux

LAMSADE, Université Paris-Dauphine

Version du 21 août 2017

L'enseignant

- Olivier Cailloux
- olivier.cailloux@dauphine.fr
- Coordonnées : cf. [annuaire](#) de Dauphine

Objectifs pédagogiques

- Programmer des applications d'envergure
- De qualité
- Portables
 - Appui sur API ouvertes
 - Programmer selon les specs
- Programmation Java (mais surtout principes généraux!)

Objectifs pédagogiques (plus précisément)

Architecture web

Appui sur quelques standards du web actuels

- Services Web REST
 - Formats complémentaires : XML et JSON
 - BD SQL et ORM
 - Réduction de la complexité par division
-
- Prise en main d'outils de dev avancés :
 - Eclipse ;
 - Maven ;
 - Git...
 - Beaucoup de paramètres \Rightarrow comprendre !

Objectifs pédagogiques (agilité)

Approche agile ? À moitié !


- Livraisons fréquentes
- Travail en binôme
- Réusinage fréquent

Un environnement imparfait

Log Viewer – Mozilla Firefox

servlet best pract... x Maven – Introduc... x http://loc...ityManager x PersistenceCont... x PersistenceCont... x hibernate - When... x

localhost:4848/common/logViewer/logViewer.jsf?instanceName=server&loglevel=INFO&viewResults=true



Log Viewer

View, search, and filter a server log file using basic and advanced options. Refer to the Log Levels page for information about log levels you can filter here.

[Advanced Search](#)

Search Criteria

Text search:

Only log entries containing the specified text will be displayed. Search is case sensitive.

Timestamp: ☒ Most Recent ☐ Specific Range:

Log Level: ☐ Do not include more severe messages

Log entries are limited to those stored in the log file. Set appropriate log level in the Log Level page to ensure data is logged.

[Modify Search](#)

Instance:

Log File:

Log Viewer Results (0)

Un environnement complexe

De <http://graphserver.github.io/graphserver/> :

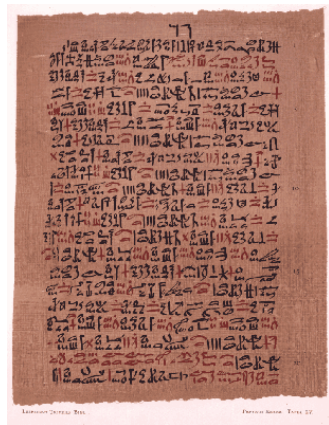
First, get Java 1.6. Or Java6. I think they're the same thing. Get the JDK, but not the JVM, the JRE, SDK, SDN, or the JCE. Note Java SE comes with Jave EE, which is apparently at version 5, and may or may not have anything to do with J2EE. I don't know what those have to do with anything. Google it or something. I don't know. They don't make it particularly easy for you. It's like, they've got more money than god and nothing pleases them better than pouring all that expertise into baffling the hell out of you. Honestly, I can't stand Java, but you need it to run Osmosis.

Intérêt pratique

- Qu'on soit programmeur, qu'on discute avec des programmeurs
- 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
- Technologie en vogue
 - Java EE VS Spring VS ... ?

Les bienfaits de la complexité

- Activité plus difficile : souvent plus attrayante
- Évite les activités répétitives : complexité amène diversité
- Récompenses plus grandes
- Recherche d'un accomplissement personnel
- Cercle vertueux : accès à activités plus complexes
- Pas un bien positionnel : accessible à tous



Écriture **hiératique**, égypte ancienne

Prérequis

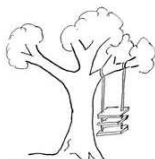
- Programmation en Java, manipulation d'un environnement de développement, compréhension des notions algorithmiques élémentaires.
- Capacité à comprendre des textes en anglais liés à l'informatique.
- HTTP, HTML, XML, SQL.

Mise en œuvre

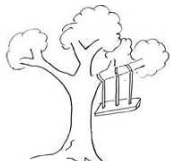
- Central : travail sur projet
- Objectif : un projet utile
- Un projet \neq par groupe
- Programmation en binôme, équipes tournantes :
diversification, pas spécialisation
- Si \neq niveaux : l'avancé aide le débutant
- Durant séance x : constituez un binôme \neq du précédent
- Travaillez sur une fonctionnalité de votre projet
- À remettre avant séance $x + 1$
- Livraisons exclusivement via git

Agilité

- Je joue le rôle du client : détail des fonctionnalités à implémenter
- Fonctionnalités de l'application décrites de manière vague : à vous de m'interroger



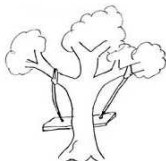
CE QUE DEMANDE L'UTILISATEUR.



CE QUI EST ECRIT DANS LE CAHIER DES CHARGES.



CE QUE L'ANALYSTE A COMPRIS.



CE QUE LE PROGRAMMEUR A REALISE.



APRES LA MISE AU POINT.



CE QU'IL FALLAIT...

Évaluation

Évaluation individuelle et collective !

- 50% CC
 - Une note par fonctionnalité
 - Obligation : ≥ 1 point de difficulté par séance pour chacun
 - *Vous êtes toujours censés comprendre votre code*
- 50% Projet (fin d'année)
 - Qualité générale du projet, correction collective des défauts relevés pendant l'année
 - Présentation collective de vos projets
 - Soutenance orale : comprenez-vous votre code ?
 - Nb points de difficultés
 - Diversité des aspects
- Fin d'année : Vote pour la meilleure application

Travail attendu

- $\{[(25 \text{ à } 30\text{h} / \text{ECTS}) \times 3 \text{ ECTS}] - 24 \text{ h}\} / 7$ inter-séances
- 9 heures de travail entre chaque séance
- Voir exigences sur le site [GitHub java-course](#)

Astuces importantes

- Éviter l'essai / erreur : *comprendre*
- Poser des questions !
- Commencer simple
- Si ça ne fonctionne pas : faire plus simple
- Exclure bugs possibles pas à pas
- Éviter débuggage pas à pas
- Se méfier du code auto-généré
- Choisir *une* approche cohérente. Mélange deux tutos : problèmes presque assurés (\neq versions ; \neq annotations...)

Attendu

- Vous êtes supposé lire les annonces publiées sur MyCourse
- Rediriger vos e-mails @ Dauphine si nécessaire pour vous assurer de recevoir les annonces
- Suivre *scrupuleusement* les instructions sur le site SVP

Licence

Cette présentation, et le code LaTeX associé, sont sous [licence MIT](#). Vous êtes libres de réutiliser des éléments de cette présentation, sous réserve de citer l'auteur.

Le travail réutilisé est à attribuer à [Olivier Cailloux](#), Université Paris-Dauphine.