



Travaux Dirigés de JEE n°3

Cours de Master 2 Informatique

Spring et Hibernate

L'ensemble du code, des noms de classes, des champs, des méthodes doivent être en anglais. Le code doit répondre aux normes SUN et être obligatoirement testé unitairement.

Le rendu du TD doit être fait par mail à l'adresse suivante : loyaute@univ-mlv.fr. Il s'agit de rendre un fichier zip constitué d'un rapport au format pdf et du code source dans un jar non exécutable

► Exercice 1. Intégration de Spring

Dans cet exercice, nous allons réaliser l'intégration de Spring afin de rendre exploitable ce que vous avez fait durant le TD précédent ! Pour utiliser Spring vous devez ajouter les dépendances adéquates dans le pom du projet/module qui l'utilisera.

Spring va utiliser des fichiers de configuration pour, par exemple :

- Instancier des objets en réalisant, si nécessaire, de l'injection de dépendances ;
- Gérer des connecteurs vers d'autres frameworks (quartz, hibernate, etc.) ;
- Gérer la sécurité (habilitations, accréditation) au sein de votre projet ;
- et d'autres choses...

Nous allons dans un premier temps, rajouter les répertoires `src/main/resources` et `src/test/resources` qui sont les répertoires par défaut utilisés par Maven pour les fichiers de ressources de votre projet/module. Par commodité, rajoutez les comme répertoire de sources dans Eclipse.

Dans le répertoire `src/main/resources`, rajoutez un sous-répertoire `spring` qui contiendra les fichiers de configuration utilisés par Spring. Créez un fichier `XXX-components.xml` (le triple X étant le nom de votre projet/module). Ce fichier va contenir le ou les chemins indiquant à Spring de détecter automatiquement les composants annotés (annotations `Service`, `Components`, `Repository`, `Controller`). Regardez le fichier `tutorial-components.xml` fournis en exemple.

- Faites les modifications adéquates et expliquez les ;
- Expliquez l'intégralité du mécanisme mis en jeu ;

Vous allez maintenant rajoutez le sous-répertoire **spring** dans le répertoire **src/test/resources**. Dans ce répertoire vous allez créer le fichier **XXX-test-context.xml** (triple X correspondant au nom de votre projet/module). Ce fichier de contexte va contenir les objets qui vont être instanciés à l'aide de Spring (*i.e.*, dont le cycle de vie est géré par Spring). Ces objets peuvent être une connexion à une base de données, un connecteur à quartz, *etc.* Récupérez le fichier **tutorial-context.xml** fournis en exemple.

- Faites les modifications adéquates et expliquez les ;
- Vous remarquerez que nous utilisons la base de données h2 (*in memory database*) en mode Oracle ;
- Donnez les différents modes possibles pour la base de données h2 ;
- A quoi sert l'option `DB_CLOSE_ON_EXIT` ?

Il est important de noter que nous créons ce fichier de contexte dans l'environnement de test car nous considérons actuellement que le module de persistance ne peut pas être fournis comme une application *standalone* ! Dans le cas contraire, nous aurions en plus fournis un fichier de contexte dans le répertoire **src/main/resources/spring**.

► Exercice 2. Configuration de la partie Hibernate

Lors du TD précédent, vous avez spécifié les Entités **Author** et **Book** à l'aide des annotations JPA et Hibernate. Nous allons maintenant spécifier la configuration du fournisseur d'entité (implantation JPA) ainsi que la spécification de différentes propriétés Hibernate.

Dans le répertoire **src/test/resources** rajouter le répertoire **META-INF**. Dans ce nouveau répertoire créez le fichier **XXX-test-persistence.xml**. Prenez exemple sur le fichier **tutorial-persistence.xml**.

- Faites les modifications adéquates et expliquez les ;
- Expliquez chaque propriété Hibernate que vous définissez.

► Exercice 3. Configuration de vos tests unitaires

La semaine dernière vous avez développé des DAOs ainsi que les tests unitaires associés. Vous allez maintenant configurer vos tests unitaires, pour exploiter Spring, en prenant exemple sur le test unitaire disponible.

- Faites les modifications adéquates et expliquez les ;

Lancez vos tests unitaires avec Maven.

- Que pourriez vous faire pour alimenter une population de données de test ?
- Hibernate charge, par défaut, le fichier `import.sql` comme mode bouchon ;

- *En prenant exemple sur le fichier disponible, faites les modifications adéquates et expliquez les ;*
- *Que se passe t-il lorsque vous relancez vos tests ?*
- *Que faudrait-il faire en base de données si vous réalisiez une recherche en fonction du nom d'un auteur ou d'un livre (ou en fonction d'un nom et d'une date de naissance) ?*
- *Expliquez le gain de cet ajout ;*
- *Donnez la commande SQL pour faire cette opération.*