mikael.dautrey@isitix.com
TDSpring2014N2.docx

TD n°2

Retour sur le corrigé du TD n°1

Récupérer le corrigé du TD1 (git clone https://github.com/mdautrey/ENSSAT2014.git)

Compiler + exécuter

Vérifier le résultat en ouvrant la page http://localhost:8080/greeting

Quelques travaux supplémentaires sur le TD n°1

La page greeting affiche Hello, Monsieur Sam Sick

Modifier le code pour que cette page pour qu'elle affiche Hello, Mister Sam Sick

Le bean PersistIndividu est injecté dans le contrôleur via l'interface PersistIndividu.

Il y a trois manières de réaliser une injection de dépendance avec Spring

1 – Configuration explicite en XML

2 – Configuration explicite en Java

3 – Configuration implicite avec découverte automatique des beans et autowiring

Dans cet exemple, nous sommes partis sur l'option 3, pour sa rapidité de mise en œuvre

En réalisant ce développement à la va vite, un tag qui n'est pas nécessaire a été ajouté dans le code

Trouver le tag en trop et le supprimer

Il y a également une méthode d'initialisation qui pourrait être supprimée.

Trouver cette méthode et la supprimer

Tester et valider le fonctionnement de l'application après ces deux modifications.

Le travail a été réalisé dans le répertoire « initial ». Il se trouve qu'il aurait dû être réalisé sur le répertoire complete qui propose une fonctionnalité supplémentaire de login/log off.

Déplacer les modifications réalisées vers complete, rebuilder l'application + tester

Dessiner l'architecture mise en place pour gérer un premier niveau de sécurité de l'application

Persistance avec Spring

Réaliser l'exemple http://spring.io/guides/gs/relational-data-access/ (https://github.com/spring-guides/gs-relational-data-access.git)

Dessiner l'architecture de l'application + se renseigner sur la technologie de la base h2

S'inspirer des éléments de l'exemple pour ajouter une vraie couche de persistance au projet du TD n°1 avec une table Individu comportant quatre colonnes : Civilité | Nom | Prénom | Age et une requête à cette table pour afficher le nom de l'individu (on peut, dans un premier temps : insérer les enregistrements dans la table au démarrage de l'application et afficher un Hello au premier individu sortant de la requête dans la page greeting.html)

Quelques pistes pour démarrer :

- Commencer par ajouter les dépendances du fichier Pom.xml de l'exemple data-access au fichier Pom.xml de l'application greeting
- Quels sont les contrats qui lient les objets entre eux ?
    - Quel est le contrat entre la base de données et la couche de persistance ?
    - Quel est le contrat entre la couche de persistance et le reste de l'application ?
- Simplifier l'individu en supprimant la notion d'âge
- Faut-il ajouter une colonne id et si oui pourquoi ?

Utiliser le mécanisme de programmation orientée aspect pour ajouter un log à l'entrée de la méthode qui lance le select sur la table des individus et à la sortie.