Projet Interface Homme Machine

Groupe :

TRAING Hei-Hong
GUILLEBAUD Vincent

Objectif : Réaliser un gestionnaire de contact en utilisant le framework Spring MVC.

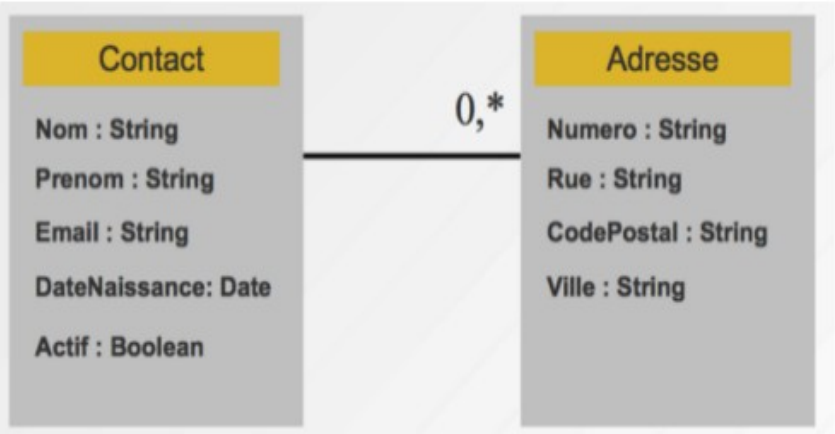
1) Architecture du projet

Le modèle métier de l'application est assez simple et est composé de deux class :

Un contact possède les attributs suivants :
- Nom
- Prénom
- Email
- Téléphone
- Date de naissance
- Actif
- Adresses

Une adresse possède les attributs suivants :
- Identifiant de l'adresse (facturation, domicile...)
- Numéro
- Rue
- Code postale
- Ville

Nous pouvons ainsi résumer l'architecture avec l'UML suivant :



Afin de trier les contacts par ordre alphabétique nous avons réalisé un AbstrctComparator qui est implémenté par deux Comparator l'un par nom l'autre par prénom.

L'utilisation de ce Comparator nous permet d'afficher les contacts contenus dans la Map par ordre alphabétique.

2) La simulation du DAO :

Pour simuler la couche persistante nous avons utilisé un singleton comprenant une Map qui contient les données des contacts.

La Map fonctionne avec un système de clé / valeur.
Map<Integer, Contact>

ici la clé utilisée est un entier de type Integer pour identifier un contact. La valeur est l'objet contact. Afin de pouvoir tester l'application nous avons ajouté à l'initialisation des données.

3) L'interface coté client

Utilisation de Bootstrap :

Nous souhaitions avoir une mise en page responsive pour que l'application soit utilisable sur ordinateur et sur smartphone. Afin de conserver une bonne mise en page suivant la taille et la résolution des écran nous avons utiliser le framework Bootstrap qui permet de faire de la mise en page responsive.

Nous avons essayé de réaliser une interface la plus intuitive possible afin que l'utilisateur comprenne rapidement le fonctionnement de l'application.

Utilisation d'AJAX :

L'ajout et la suppression des adresses se fait sur la même page que la modification d'un contact. Afin que lors de la suppression d'une adresse la page ne se rafraîchisse pas (afin d'éviter une perte des données du contact non enregistrées) nous avons utiliser de l'AJAX. Cela permet à l'utilisateur d'effectuer les modifications sur les adresses sans rechargement de la page.

Utilisation d'Angular :

Pour la fonctionnalité de recherche d'un contact nous avons utiliser le framework Angular. L'utilisation de ce framework nous permet de rechercher dynamiquement dans les contact et de rafraîchir les résultats de la recherche au fur et a mesure de la saisie des informations.

La recherche d'un contact se fait automatiquement sur trois critères : le nom, le prénom et le numéro de téléphone. Les résultats sont automatiquement filtrés.

Il est aussi possible en sélectionnant une lettre sur le haut de la page de filtrer les contacts et de ne conserver que ceux dont le nom commence par la lettre sélectionnée.

4) Difficultés rencontrées et points d'améliorations :

La première difficulté rencontrée à été l'utilisation de spring. En effet nous ne connaissions pas ce framework et il nous a fallu apprendre à l'utiliser avant de pouvoir se lancer sur la réalisation du projet.

La seconde difficulté à été pour le déploiement de l'application sur Google AppEngine. Après de nombreux essais nous n'avons pas réussi à déployer notre application sur cette plateforme. L'erreur rencontrée était au moment de la publication du projet. Les fichier .jsp ne pouvaient pas être compilés, nous avons l'erreur suivant : « Unable to update app : Failed to compile jsp files. »

La troisième difficulté à été la configuration du fichier .classpath
Ayant des configurations différentes le fichier .classpath n'était pas le même sur nos deux environnements de travail. Il nous a fallu alors parfois le modifier suivant notre environnement de travail après avoir effectué un git pull. Il fallait alors ajouter ou retirer une librairie dans le « java buid path » d'éclipse.

Il existe plusieurs points d'amélioration pour cette application. La première serait d'ajouter des tests unitaires pour pouvoir tester la robustesse du système.

Un autre point d'amélioration possible serait l'implémentation de la couche de persistance pour sauvegarder en base de données les contacts.

Les sources du projet sont disponibles sur GitHub :
https://github.com/heihong/IHM_projet/