Une proposition de correction pour le projet Java

# Ressources

Vous trouverez dans le zip « Projet JAVA QCM.zip » une proposition d’ébauche de correction pour le projet java QCM.

Dedans, vous trouverez :

* Le backup de la base de données utilisée pour le dev (qcm.bak)
* Le script de création des tables (bdd.sql)
* Le projet Eclipse correspondant
* La navigation entre les écrans (IHM.pdf) qui n’est plus forcément à jour

# Mise en route

Pour tester la solution :

* Mettez en place la base de données avec le backup
* Chargez le projet Eclipse
* Modifiez le fichier WebContent/WEB-INF/lib/web.xml
  + repertoireStockage : doit pointer vers le répertoire « upload » de l’application
  + repertoireArchivage : doit pointer vers un répertoire où écrire les résultats d’un test en cas de contestation. Regarder le contenu du répertoire « archives » du zip pour voir ce qu’il en est.
* Modifiez le fichier context.xml (dans le répertoire META-INF) pour modifier la chaîne de connexion

La base de données n’est pas très riche.

L’application n’est pas robuste à toutes les utilisations non classiques

Voici les logins à utiliser :

* Login formateur : trichard/mdp
* Login stagiaire : tharel/mdp

Au niveau interface, je ne suis pas très bon comme vous allez pouvoir le constater… J’ai utilisé JQuery pour gérer tout ça mais… il m’a envoyé dans des réalisations un peu complexes à mon goût. Si quelqu’un a l’envie de refaire l’interface, qu’il n’hésite pas…

# Architecture générale

L’architecture utilisée est classique :

Action utilisateur 🡺 Servlet 🡺 Traitement métier 🡺 Affichage résultat via JSP

## L’IHM

L’IHM est composée de 3 principaux répertoires :

1. Administration
   1. Ce répertoire contient l’ensemble des interfaces permettant à un formateur de gérer des tests : créations/inscriptions
   2. Un filtre d’accès est mis en place (fr.eni\_ecole.qcm.web.**filters**.FiltreFormateur)
2. Authentification
   1. Ce répertoire contient la page d’authentification
3. Passage
   1. Ce répertoire contient l’ensemble des interfaces permettant le passage d’un test pour un stagiaire
   2. Un filtre d’accès est mis en place (fr.eni\_ecole.qcm.web.**filters**.FiltreStagiaire)

L’accès est donc soumis à une authentification préalable

## Les servlets

Les servlets permettent de recevoir les actions utilisateur.

5 servlets ont été mises en place pour gérer :

1. Les pages d’administration
2. La page d’authentification
3. Les pages de passage d’un test
4. Le calcul du temps restant
5. La gestion de l’upload d’image

Les servlet principales sont :

* fr.eni\_ecole.qcm.web.**servlets**.Administration
* fr.eni\_ecole.qcm.web.**servlets**.Passage

Leur rôle est assez simple :

* Détecter le type de la demande et l’action correspondante
* En fonction de ces informations, elle renvoie le traitement vers le bon contrôleur métier

## Les contrôleurs

Les contrôleurs ont pour rôle d’effectuer le traitement métier

Les classes utilisées se trouvent dans les packages suivants :

* fr.eni\_ecole.qcm.web.**controleur**.administration
* fr.eni\_ecole.qcm.web.**controleur**.authentification
* fr.eni\_ecole.qcm.web.**controleur**.passage

Je vous laisse découvrir le contenu.

Les contrôleurs manipulent les données des packages ci-dessous.

## Le modèle métier

Les classes utilisées pour manipulées les informations sont décrites dans les packages suivants :

* fr.eni\_ecole.qcm.web.**modele**.administration
* fr.eni\_ecole.qcm.web.**modele**.passage

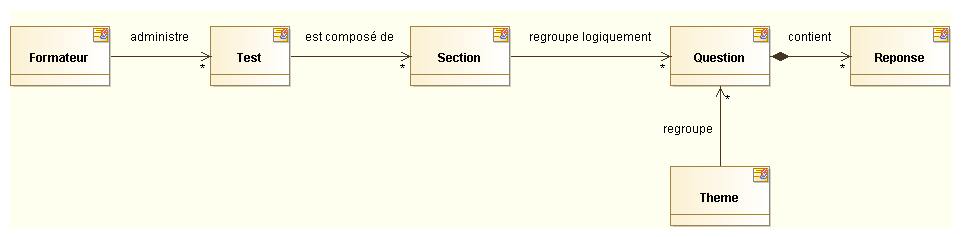
## La persistance des données

Les clases utilisées pour rendre persistant les données sont dans les packages suivants :

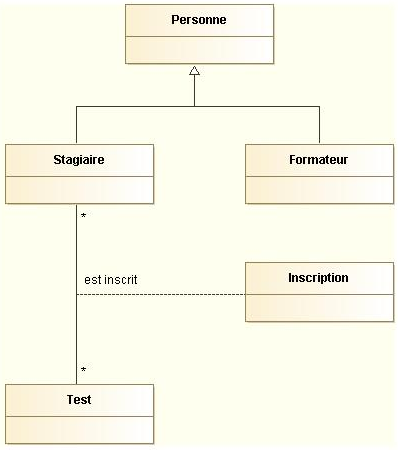
* fr.eni\_ecole.qcm.web.**basededonnees**
* fr.eni\_ecole.qcm.web.**basededonnees**.administration
* fr.eni\_ecole.qcm.web. **basededonnees**.authentification
* fr.eni\_ecole.qcm.web. **basededonnees**.passage

# Les diagrammes de classes

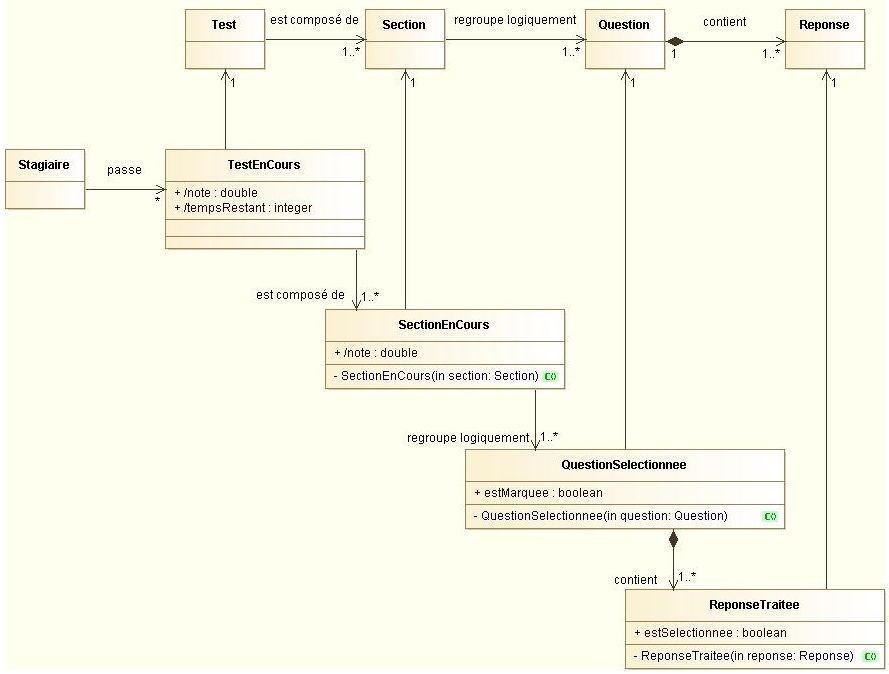
Créer un test :



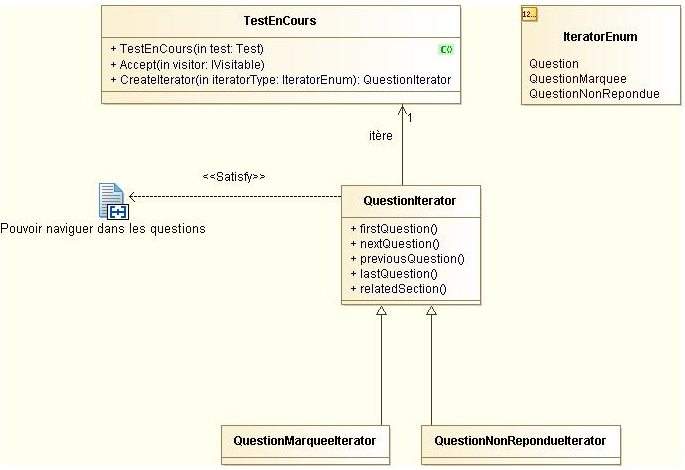
Inscrire à un test :



Passer un test :



Itérer les questions :



Noter et archiver un test :

