MIN15114 - Programmation, GL, preuve
Examen (jan. 2017)

Stéphane Lopes Zoubida Kedad

Durée : 2h - Document autorisé: 1 feuille A4 recto-verso

Exercice 1 (Questions de cours)

1. Quelle différence faites-vous entre principes, patterns et idiomes ?
2. Expliquez le fonctionnement du pattern OBSERVER en vous appuyant sur son utilisation dans la bibliothèque standard de Java.
3. Expliquez le lien entre un SGBD (MySQL par exemple), JDBC, Hibernate et JPA.
4. À quoi sert l'élément `persistence-unit` du fichier XML `persistence.xml` ?
5. Avec JPA, quelles annotations utilise-t-on pour marquer une entité et sa clé primaire ?
6. Avec un ORM, comment gère-t-on le cas des associations *n-n* ?

Exercice 2 (Design patterns)

Dans cet exercice, vous proposerez une implémentation Java pour la génération de document structuré (par exemple *HTML*).

Un document est un arbre où chaque nœud est un élément. Chaque élément possède un nom (`p`, `h1`, `span`, `div`, ...) et des attributs et englobe un contenu. Le contenu est soit un texte simple (élément terminal), soit d'autres éléments (élément composé).

Pour chaque élément, on veut pouvoir générer sa représentation. Un élément peut être représenté comme une chaîne de caractères (*<nom>contenu</nom>* en HTML). La méthode `toString` devra produire la représentation HTML de l'élément.

Pour chaque implémentation ci-dessous, vous donnerez un extrait de code utilisant votre proposition.

1. Donnez un diagramme de classes UML qui modélise votre proposition.
2. Donner le code Java de la classe `ElementTerminal`. Vous utiliserez le pattern BUILDER pour leur création.
3. Donner le code Java de la classe `ElementCompose`. Vous vous appuierez sur le pattern COMPOSITE pour cela.
4. Ajoutez la possibilité de parcourir la hiérarchie (parcours en profondeur). Vous vous appuierez sur le pattern ITERATOR pour cela.

Exercice 3 (Persistance avec JDBC et le pattern DAO)

Dans cet exercice, vous allez développer, avec JDBC et le pattern DAO, la couche de persistance d'une application gérant un ensemble d'ouvrages.

Un *livre* est identifié par son *numéro ISBN*, possède un *titre* et est écrit par un ou plusieurs *auteurs*. Un *auteur* possède un *nom* et une *adresse email*. Un auteur peut écrire plusieurs livres (zéro ou plus) et chaque livre peut être écrit par plusieurs auteurs (au moins un).

L'application doit permettre :

— d'afficher les caractéristiques des livres (dont les auteurs),

— pour chaque auteur, d'afficher sa bibliographie.

1. Donnez un diagramme de classes UML qui modélise cet énoncé (uniquement le domaine sans la couche de persistance).
2. Donnez l'implémentation Java de la classe `Auteur`. L'affichage des caractéristiques de l'étudiant se fera avec la méthode `toString`.
3. Faites de même pour la classe `Livre`. Dans la suite, on suppose l'existence de la classe abstraite `DAO<T>` vue en cours. En particulier, vous supposerez que les tables sont présentes dans le SGBD.

4. Proposez une implémentation Java pour une classe *Connexion* gérant la connexion à un SGBD. Pour cela, vous vous appuierez sur le pattern SINGLETON. Vous réaliserez cette classe pour une connexion à *MySQL* sur *localhost* et la BD *exam* (utilisateur *user*, mot de passe *passwd*).
5. Donnez le squelette (déclaration et signature des méthodes) des classes DAO nécessaires.
6. Donnez l'implémentation de la méthode `AuteurDAO.create` qui rend persistant un auteur. **En particulier, proposez une solution pour la persistance de l'association auteur-livre**.
7. Donnez l'implémentation de la méthode `AuteurDAO.find` qui recherche un auteur à partir de son nom.
   Dans la suite, on suppose que les classes DAO sont totalement implémentées.
8. Donnez l'extrait de code qui crée deux auteurs, les deux livres qu'ils ont écrits et rend les objets persistants.
9. Donnez l'extrait de code qui récupère un auteur et affiche sa bibliographie.
10. Donnez le code de la classe `DaoJdbcFactory` qui implémente le pattern FABRIQUE pour la création des DAO.
11. Donnez le code de la classe `DaoAbstractFactory` qui implémente le pattern FABRIQUE ABSTRAITE pour la création des DAO.
12. Quels changements faut-il apporter au code qui utilise ces DAO ?
13. Donnez un diagramme de classes UML qui reprend l'ensemble des classes créées et leurs relations.