TD 1 Java EE

Mise en place des objets métiers & Outils de qualimétrie

A rendre pour le 22 octobre 2010 à 18h.

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.*

1. **Des auteurs...**
   Dans un premier temps, nous allons définir un auteur d'un livre par l'ensemble des caractéristiques suivantes:

   - Un numéro de sécurité social (ssid);
   - Civilité (Mlle, Mr, Mme, Dr, Pr);
   - Un nom de famille;
   - Des prénoms;
   - Une date de naissance;
   - Des adresses (domicile, professionnelle);
   - Des numéros de téléphone (portable, domicile, professionnel);

   (a) Définissez les classes dont vous avez besoin afin de représenter un auteur;
   (b) Modélisez par un schéma UML les classes et leurs interactions (composition, aggrégation);
   (c) Implantez maintenant votre modélisation UML;
   (d) Quelles sont les opérations possibles sur ces champs?
   (e) Rajoutez les méthodes adéquates.

2. **...et des livres**
   Nous allons maintenant définir un livre par les caractéristiques suivantes:

   - Un numéro isbn;
   - Un titre;
   - Un nombre de page;
   - Une date d'impression;
   - Un nombre de page;
   - Des auteurs;
   - Une société d'édition;

   De même que précédemment:

   (a) Définissez les classes dont vous avez besoin afin de représenter un livre;
   (b) Modélisez par un schéma UML les classes et leurs interactions (composition, aggrégation);
   (c) Implantez maintenant votre modélisation UML;

(d) Quelles sont les opérations possibles sur ces champs?

(e) Rajoutez les méthodes adéquates.

3. **Structurer son code source**
   Nous allons maintenant structurer le code source en le mettant dans un répertoire particulier, packager le tout, *etc.*.

   (a) Le nom de votre projet va être `tutorial-persistance`;

   (b) Le code que vous avez précédemment écrit doit être placé dans le répertoire `src/main`;

   (c) Vous allez packager votre code de la facon suivante:
   - `fr.umlv.jee.tutorial.persistance.author`
   - `fr.umlv.jee.tutorial.persistance.book`

4. **Ecriture des tests unitaires**
   L'ensemble du code que vous écrivez doit être obligatoirement testé unitairement! Pour se faire nous allons tester notre code en utilisant `JUnit`.

   - Créez le répertoire de source suivant: `src/test`;
   - Créez les packages suivants:
     - `fr.umlv.jee.tutorial.persistance.author`
     - `fr.umlv.jee.tutorial.persistance.book`
   - Ecrivez maintenant vos tests unitaires pour toutes vos classes!

   Vos tests unitaires couvrent-ils correctement votre code?

   Pour le savoir:

   - Ajoutez un plugin de couverture de tests, par exemple le plugin eclemma d'Eclipse (`http://eclemma.org`).
   - Installez le (site update);
   - Lancez vos tests unitaires via la commande *Coverage As*.
   - Vous verrez alors le taux de couverture de vos tests unitaires mais, pas leur pertinence!

5. **Qualimétrie de votre code source**
   Des outils sont disponibles afin de vérifier, par exemple:

   - L'absence de copier-coller;
   - Le respect des normes de codage;
   - L'absence de code mort (variable non utilisée ...);
   - Présence de Javadoc et de commentaires dans le code;
   - *etc.*.

   Nous allons produire maintenant des rapports avec de tels outils.

   - Intégrez dans votre IDE les plugins des outils **PMD**, **Checkstyle**, **JavaNCSS** et **Findbugs**;
   - Lancez maintenant ces outils sur votre projet actuel, sur un projet que vous avez réalisé l'année dernière.
   - Etudiez les rapports produits et les suggestions de corrections;
   - Que pouvez-vous en déduire sur la mise en place de tels outils en entreprise?

- A quoi sert l'intégration continue? Peut-on intégrer les outils cités ci-dessus dans un tel environnement?
- Y'a t-il un intérêt d'utiliser l'intégration continue en entreprise?
- A quoi sert l'outil Sonar?
- A quoi sert l'outil Japex? Réaliser un exemple d'utilisation de cet outil sur les différentes listes de l'API Java (*java.util* et *java.util.concurrent*).
- Que signifie le concept de *continuous performance testing*?
- Explicitez le patron de conception **Decorator** puis, citez un package dans lequel le patron de conception **Decorator** est utilisé.