Planete-Kids

Vente en ligne de vêtements pour enfants

Dossier de conception système

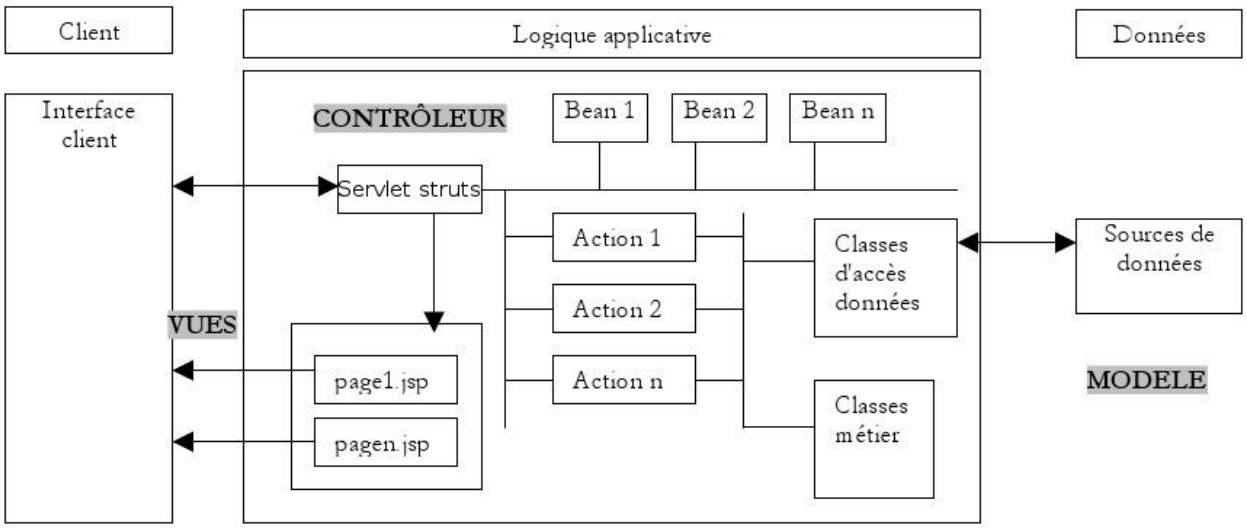
Guillaume Missionnier
Clément Plantier
Guillaume Renault
Marc Schaller

Table des matières

1. Schéma d'architecture système

1. Shéma général

Nous avons fait le choix d'utiliser le framework Struts pour développer notre site de commerce en ligne J2EE. Voici un schéma du modèle MVC utilisé par Struts :



Dans le détail nous avons :

Client :

- Accès admin en ligne de commande à travers un shell.
- Accès via un site WEB

Logique applicative :

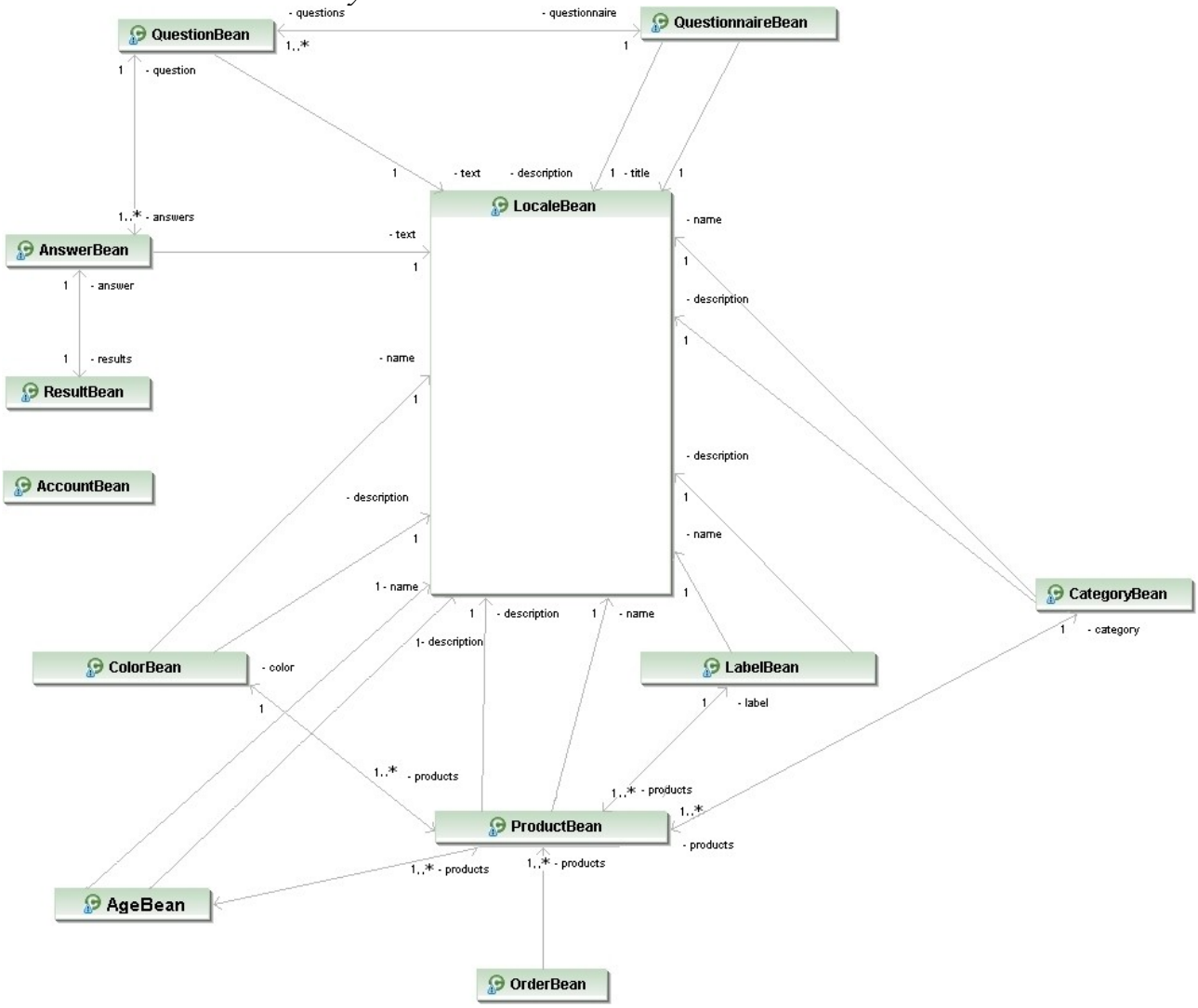
Cette partie est découpée en 3:

- Le contrôleur principal, une servlet générique fournie par Struts dans notre cas ; l'ensemble des informations dont elle a besoin sont configurées dans un fichier : *struts.xml*
- Les classes d'actions, qui permettent de traiter l'ensemble des requêtes.
- Les beans, de deux catégories dans notre cas : les beans entity, et les beans stateful. Suivant la requête reçue, c'est le servlet qui décide si il y a un bean à créer ou non.

Données :

- Nous avons choisi une base MySQL pour la base de donnée. Cependant, modulo quelques modifications de configuration et de driver SGBD, une autre base pourrait être utilisée : le site fonctionne également sur une base PostgreSQL.

2. *Beans Entity*



3. *Beans Stateful*

Les beans stateful sont volontairement détachés des beans entity dans la présentation afin de rendre les choses plus lisibles.

Le *CustumerBean* est en relation avec les *ProductBean*, *ColorBean*, *LabelBean*, *CategoryBean* et le *QuestionnaireBean*.

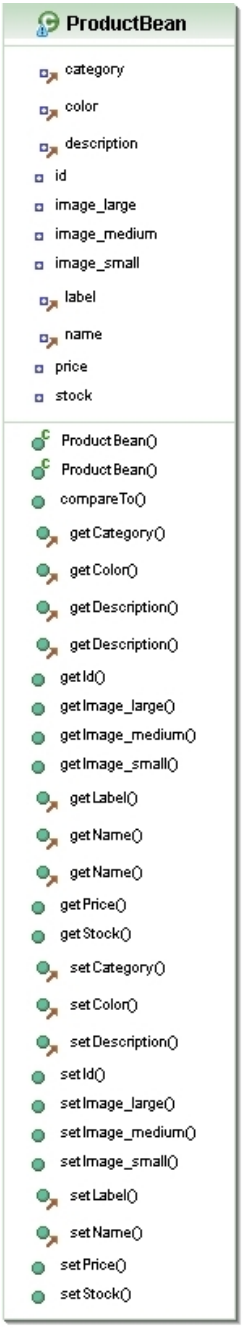
L'AdminBean est en relation avec tous les beans entity, dans la mesure où il s'agit du bean qui permet de gérer l'ensemble du site.

Le *CartBean* est quant à lui chargé de gérer le panier du client. Il est principalement en relation avec l'*OrderBean* et le *ProductBean*.

2. Description fonctionnelle

1. Beans Entity

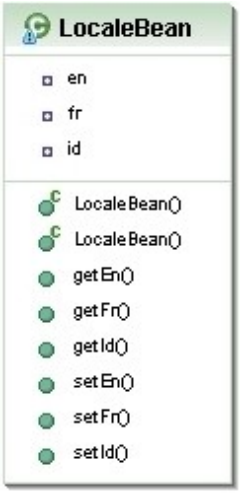
Tous les Beans entity sont construits sur le même modèle : Des attributs qui le composent, deux constructeurs (un vide, et un autre prenant en paramètre l'ensemble de tous ses attributs), des « getters » et des « setters ». Voyons les un par un, en occultant progressivement les attributs et méthodes qui reviennent ou qui ont un nom explicite et sans réel besoin d'être développés.



On abordera dans un premier temps les beans entity qui font référence aux produits du site. Dans un second temps, on détaillera ceux relatifs au questionnaire.

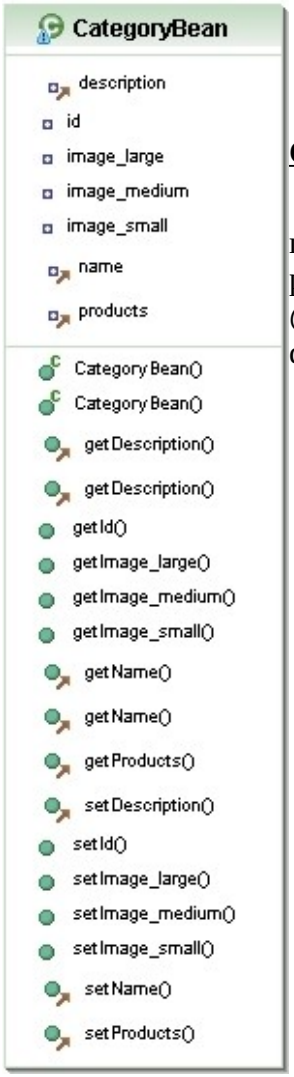
ProductBean:

Un produit est identifié de manière unique par son Id. Il est associé à une catégorie (categoryBean), une couleur (colorBean), une marque (LabelBean), un nom et une description (LocaleBean), ainsi qu'une tranche d'âge associée (AgeBean) qui n'est pas encore présent au moment de l'écriture du dossier, d'où son absence. L'ensemble des chemins permettant d'accéder aux différentes images sont également présentent. On y retrouve également le prix et le stock. Les différentes méthodes permettent d'accéder à l'ensemble des attributs.



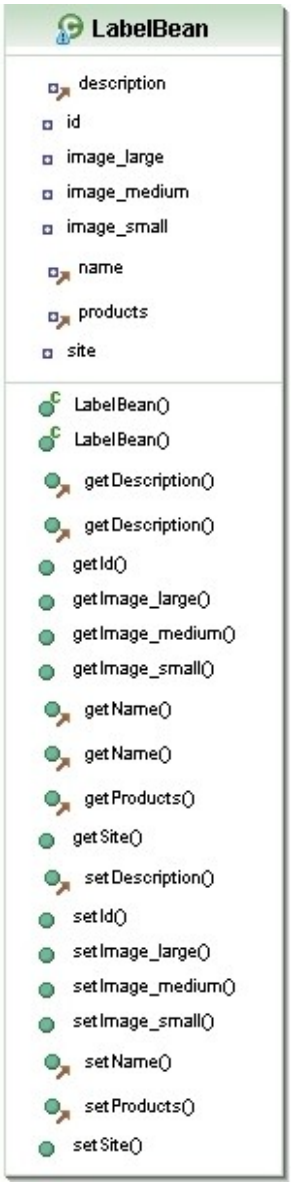
LocaleBean :

Il s'agit d'un bean essentiel, car c'est lui qui gère les différentes langues du site. Il est utilisé à chaque fois qu'une chaîne de caractères doit être connue en plusieurs langues. On gère ici le français et l'anglais.



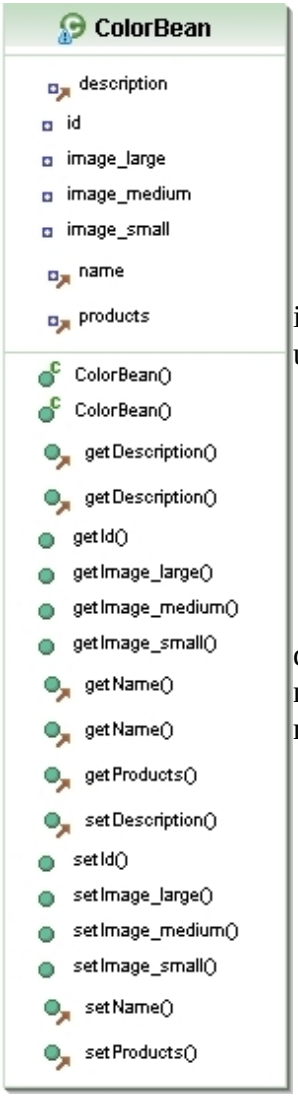
CategoryBean :

Une catégorie permet de classer un produit. Elle est identifiée de manière unique par son *Id*. L'attribut *products* est une liste de tous les produits appartenant à la catégorie. Il est associé à une relation @OneToMany. L'ensemble des méthodes copie le fonctionnement de celles de productBean, et les noms sont explicites.



LabelBean :

Le LabelBean, comme son nom l'indique, orchestre les marques relatives à chaque produits. Un nouvel attribut comportant l'url du site est apparu en plus de ceux déjà abordés. Il est identifié de manière unique par un *Id*. Les méthodes qu'il contient ont un nom explicite, on ne s'attardera pas dessus.



ColorBean :

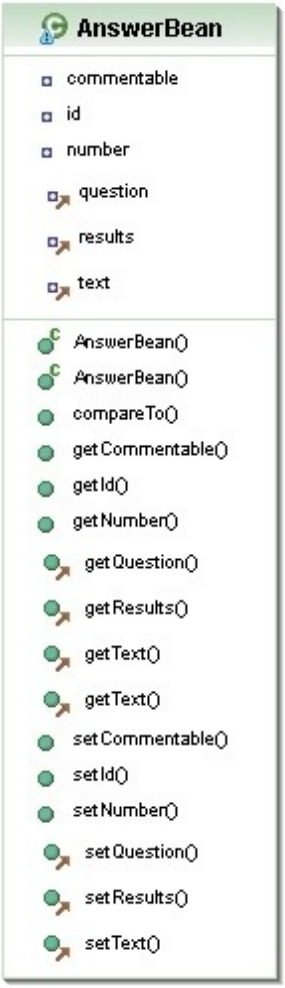
C'est l'avant dernier bean utilisé pour la description d'un produit. Il est identité de manière unique par son *Id*. Le reste des attributs et méthodes ont un nom explicite ou bien ont déjà été abordés.

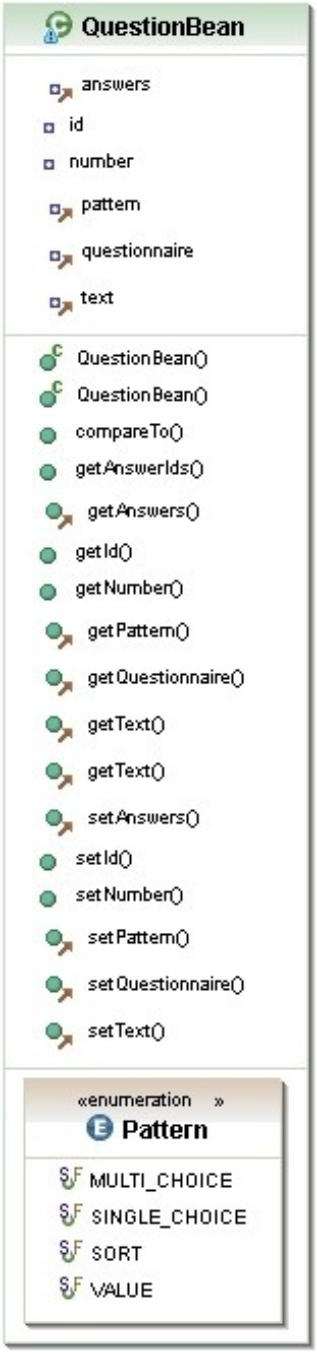
AgeBean :

Ce bean est chargé de faire correspondre un produit à une tranche d'âge. Il est donc en relation avec les produits (ProductBean) et comprend un nom (LocaleBean) nommant la tranche d'âge associée, ainsi que des méthodes permettant d'agir sur ses attributs au niveau administrateur.

OrderBean :

C'est les bean qui est chargé des commandes. Il a comme attribut la liste des *produits* (ProductBean), et pour chaque produit son prix et sa quantité. On y retrouve également un attribut spécifiant l'état (type énuméré interne à la au bean). Sont présentes les méthodes qui permettent la mise à jour de l'état de la commande.





Partie relative à l'environnement Questionnaire

QuestionnaireBean :

Un questionnaire est constitué d'une *description* et d'un *titre* (LocaleBean), ainsi que d'un ensemble de *questions* (QuestionBean) avec une relation @OnetoMany. Il est identifié de manière unique par son *Id*. Les « getters » et « setters » sont sur le même modèle que ce qui a déjà été vu.

QuestionBean :

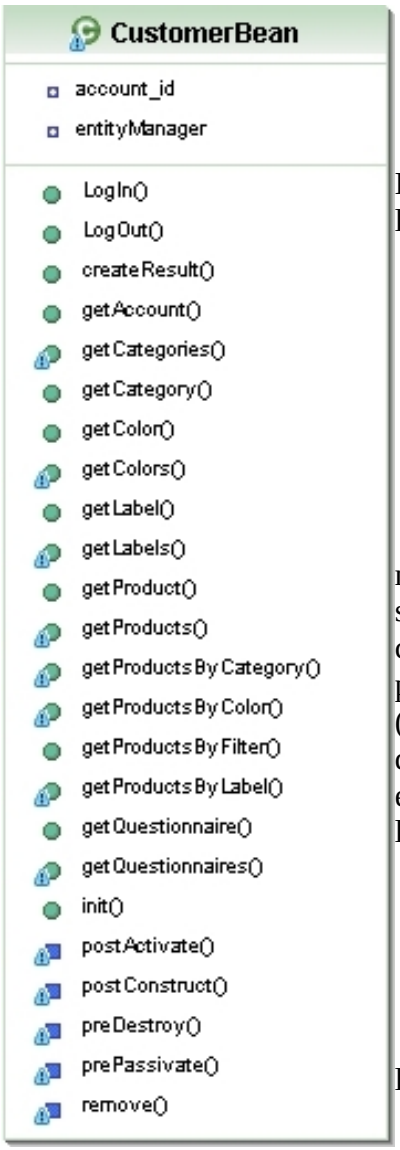
Chaque question est identifiée de manière unique par son Id. On retrouve une relation @ManyToOne au niveau de l'attribut *questionnaire*. Une question contient à cause de sa nature un ensemble de réponses possibles, lesquelles sont stockées dans des AnswerBean. L'attribut *answers* se voit donc attribuer une relation @OneToMany. Enfin, l'attribut *pattern* est un type énuméré permettant de décrire le genre de la question.

AnswerBean :

C'est le bean contenant la réponse à la question auquel il est associé. Son fonctionnement est sur le même modèle que les autres beans : un Id unique, des « getters » et des « setters » permettant d'accéder aux attributs.

2. *Beans Stateful*

Les beans session ont une courte durée de vie. Ils agissent principalement pour réaliser une opération, et n'ont pas pour rôle de stocker des informations. Les beans stateful permettent de conserver l'état du bean entre plusieurs appels.



CustomerBean :

C'est le bean qui permet de gérer une session client. L'ensemble des méthodes qui le compose sont des « getters », dans la mesure où un client ne peut que récupérer des informations.

AdminBean :

Ce Bean n'est pas exposé ici, un trop grand nombre de méthodes rendrait la lecture impossible. Il est bon de savoir qu'il s'agit du Bean permettant l'administration du site. A l'inverse du customerBean, l'adminBean contient aussi les « setters » pour les produits (ProductBean), les marques (LabelBean), les couleurs (ColorBean), les différentes tranches d'age (AgeBean), les questionnaires (QuestionnaireBean), les questions (QuestionBean), et les réponses (AnswersBean). On aura aussi la possibilité de gérer les commandes (OrderBean).

CartBean :

C'est le bean qui constituera le panier. Il sera constitué d'une liste de produits (ProductBean) associés à une quantité.