PROJET

PROGRAMMATION ORIENTEE OBJECT :
Gestion d'une base de Données de Bières

L2S4
SCHEIDEL ARTHUR
BOLTENHAGEN MATHILDE

1. Introduction

Comme l'indique le titre, nous avons réalisé un petit logiciel de gestion de base de données sur les bières. L'utilisateur doit pouvoir afficher sa base de données ainsi que la manipuler. Nous avons choisit l'affichage le plus courant pour ce type d'informations : un tableau.

Lors de la première ouverture du logiciel, l'utilisateur n'a pas encore ajouté de bières dans sa base de données mais nous avons décidé de mettre une bouteille en exemple. Ainsi, la compréhension du logiciel est plus rapide et l'utilisateur peut avoir une version de son affichage. Si la bière en exemple ne lui plait pas, il pourra la supprimer par la suite.

En effet, ce logiciel est equipée de multiple actions :

“ajouter” : un bouton ajouter ouvre une nouvelle fenêtre que l'utilisateur peut remplir completementou non s'il n'a pas toutes les informations nécessaires.

“supprimer”: en selectionnant une bouteille (une ligne du tableau) et en cliquant sur le bouton supprimer, l'utilisateur l'efface

“modifier” : l'utilisation de ce bouton est similaire à celui du bouton supprimer. Lorsque l'utilisateur a choisit la ligne à modifier, une nouvelle fenêtre s'ouvre avec les anciennes valeurs de la bouteille, cela lui permet une bonne visibilité sur les informations qu'il a et qu'il veut changer. L'utilisateur peut aussi double-cliquer sur une cellule qu'il souhaite modifier directement à l'intérieur du tableau.

“filtrer” : en cliquant sur ce bouton, une fenêtre s'ouvre et l'utilisateur peut entrer un mot. Lors de la
validation de ce dernier, la fenêtre n'affichera plus que les bouteilles contenant ce mot dans l'une des colones.
“quitter filtrage” : ce bouton sert simplement à revenir à la base de données complete après un filtrage

“sauvegarder” : les boutons “ajouter”,”supprimer” et “modifier” activent chacun deux actions : une modification de l'affichage et une modification des données en mémoires. Nous avons choisi d'utiliser la sérialisation très pratique de Java qui permet de garder un objet crée dans un programme dans un fichier “.ser”. Si l'utilisateur ne sauvegarde pas, ses modifications seront perdues.

“exporter” : le point faible du fichier de sauvegarde est que l'utilisateur ne peut pas le lire directement. Pour cela, un bouton exporter lui permet de réaliser une copie de sa base de données dans un fichier “.txt”

“afficher/masquer les images” : cette option peut être pratique pour les utilisateurs ayant de petits écrans d'ordinateurs

2. Description du code

La structure du breuvage est très inspirée du sujet de l'ent où la base des données est la bouteille qui contient un breuvage qui lui contient une brasserie qui elle contient un lieu d'origine. Notre tableau affiche les données du vecteur_Bouteilles de Vecteurs.java grâce à Model qui les rangent dans une ArrayList.

Différentes Classes :

-**Bouteille.java** est composée d'une Icon et de trois String pour la taille, le bouchon et le type et d'un Breuvage

-**Breuvage.java** est composée de six String pour les mêmes informations que le sujet mais aussi pour deux commentaires : un commentaire de dégustation et un commentaire libre, une liste de fermentation (String[]) et d'une Brasserie.

-**Brasserie.java** est composée d'un String, son nom et d'un LieuO

-**LieuO.java** est composée de deux String, le nom du lieu et le pays.

-**Vecteurs.java** est composée d'un vecteur de Bouteilles.

-**Model.java** est composée de la liste des Bouteilles et de l'entête du tableau, c'est l'interieur du tableau.

-**Data.java** est composée du Vecteurs à sauvegarder et des méthodes de sauvegarde dans le fichier “.ser”

-**Tableau.java** est composée des données de sauvegardes, du model, du tableau et deux autres variables pour l'affichage (printimage : pour les images ; sorter : pour le cadre)

-**Ajout.java** et **Modif.java** sont composées de leurs boutons et leur nombreuses variables de sauvegardes des données entrées.

3. Repartition du travail et avancement

Nous avons d'abord déterminé nos classes principales des 4 objets. Puis Mathilde a commencé à faire la lecture de quatre fichiers textes, un pour chaque objet, mais cette méthode était peu efficasse puisque nous lisions et ouvrons quatres fichiers. Elle a donc refait la lecture pour un seul fichier lorsque nous avons découvert l'existance des fichiers “.ser”. Pendant ce temps, Arthur commençait à regarder le fonctionnement d'une interface utilisateur. C'est comme qu'on décida, tout d'abord, de se diviser :

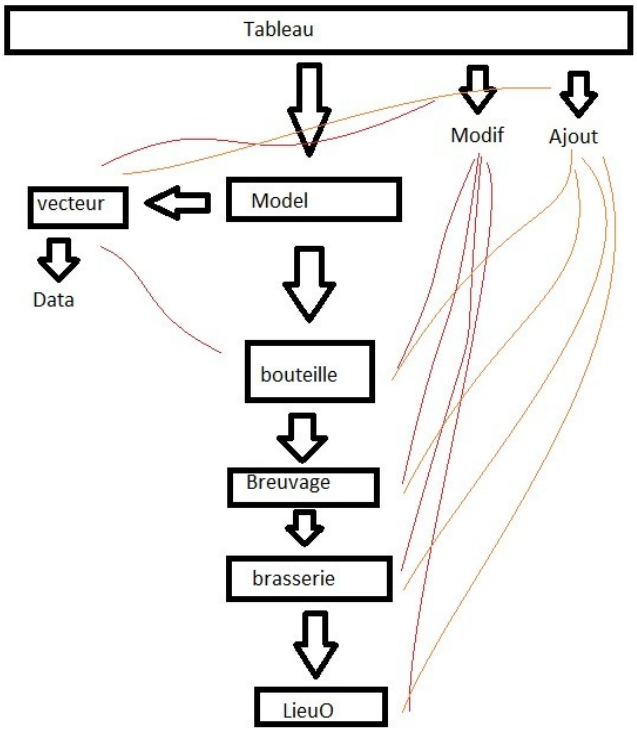
- Arthur s'occuperait de l'affichage
- Mathilde s'occuperait des l'algorithmes internes

Grâce à nos nombreuses recherches, nous trouvons facilement des exemples et des tutoriels bien expliqués. A cause de ces recherches, nous avons changés à plusieurs reprises, le principe de gestion que l'on comptait mettre en place.

Finalement, nous avons décidé d'utiliser un vecteur contenant les bouteilles que l'on affiche dans le tableau. L'affichage et le vecteur subissent les mêmes modifications auw mêmes moments.

Au fur et à mesure, on constata que l'interface utilisateur était bien plus complexe et nous confondions donc nos deux parties. Notre organisation se passait sur une plateforme appelée “c9io” qui nous permettait de voir les différentes versions de nos travaux. Ainsi, nous avons plus ou moins suivit les deux parties de notre code (affichage et algorithme).

4. Digramme UML



5. Amélioration

Nous avons rencontrés quelques difficultés lors de la suppression et de la modification (qui supprime aussi). Notre dernière version est quasiement bonne, seul le dernier (premier dans le tableau) élément ne peut être ni modifié, ni supprimé. Peut-être cela sera-t-il résolu pour demain lors de notre présentation ?

Le programme serait plus agréable si la fenetre se redimensionnerait en fonction de la taille du tableau, on a réussi a faire cela pour les cases dans le tableau, pas le tableau dans la fenetre.

Notre fonction de filtrage permet de faire toutes les recherches que l'on souhaite mais serait-il pas mieux de choisir une colone puis de recherche dedans, par exemple si nous souhaite une certaine brasserie.

La modification de toute les bouteilles ratacher a une brasserie aurait été bien cependant il se pourrait qu'une bouteille soit racheter et change de brasserie, sans que les autres ne doivent changer. lolilol

L'Ajout d'un menu déroulant de petits icons pour les types de fermeture et de bouteilles plutot que du texte. manque de temps pour le faire

Une case remplit de la couleur de la biere pour la case couleur de biere (dailleur trop les nerfs on pouvait trop le faire apres jy ai plus pensé)