

Pratique des Techniques Informatiques

BTI IG Développeur d'Application

Session 2010-2012

Romain BELLINA

FICHE DE SYNTHÈSE n°1

« Création de cocktail »

OBJECTIF DE L'ACTIVITÉ

- ♦ Création de classes à partir d'un diagramme de classe,
- ♦ Application des concepts d'héritage,
- ♦ Développer une application utilisant des collections d'objets,
- ♦ Chargement d'une interface graphique en JAVA.

SUPPORT DE L'ACTIVITÉ

Lieux de réalisation	Outils utilisés
♦ Mac OS X	♦ Java ♦ Eclipse

COMPÉTENCES CONCERNÉES

- C32 ☒ Développer à l'aide d'un langage de programmation procédural.
 C33 ☒ Maquetter une application, la développer à l'aide d'un langage de programmation événementielle.
 C34 ☒ Développer à l'aide d'un langage de programmation objet.
 C35 ☒ Développer autour d'une base de données relationnelle.
 C36 ☒ Développer dans le cadre d'une architecture client-serveur.

Présentation de l'activité :

« Cocktail 3000 » est une application codée en Java avec une interface graphique. Elle permet la création de boisson, alcoolisé ou non, pour ensuite créer des cocktails. L'application permet aussi la modification et la suppression des boissons ou cocktails. Les boissons et les cocktails sont ensuite stockés sur un serveur SQL.

Au lancement, l'application se connecte au serveur et charge la totalité des boissons et cocktails dans des collections de données.

Lors de la création d'une boisson ou d'un cocktail, il a été choisi de l'insérer dans la base de données puis par la suite de recharger l'intégralité des données pour mettre à jour les collections

Modèle conceptuel des Données

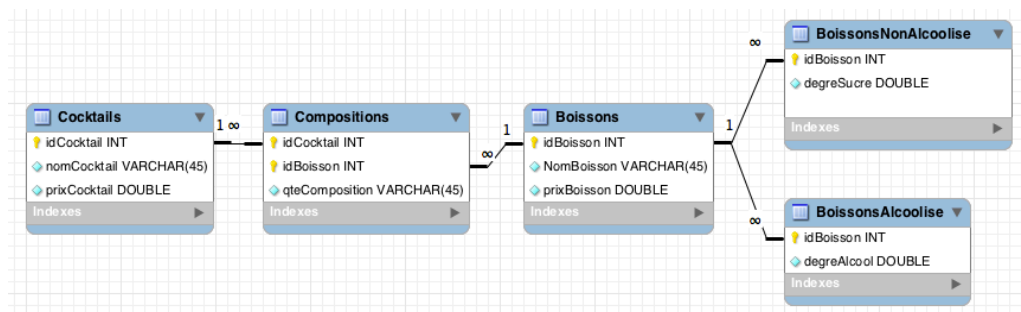
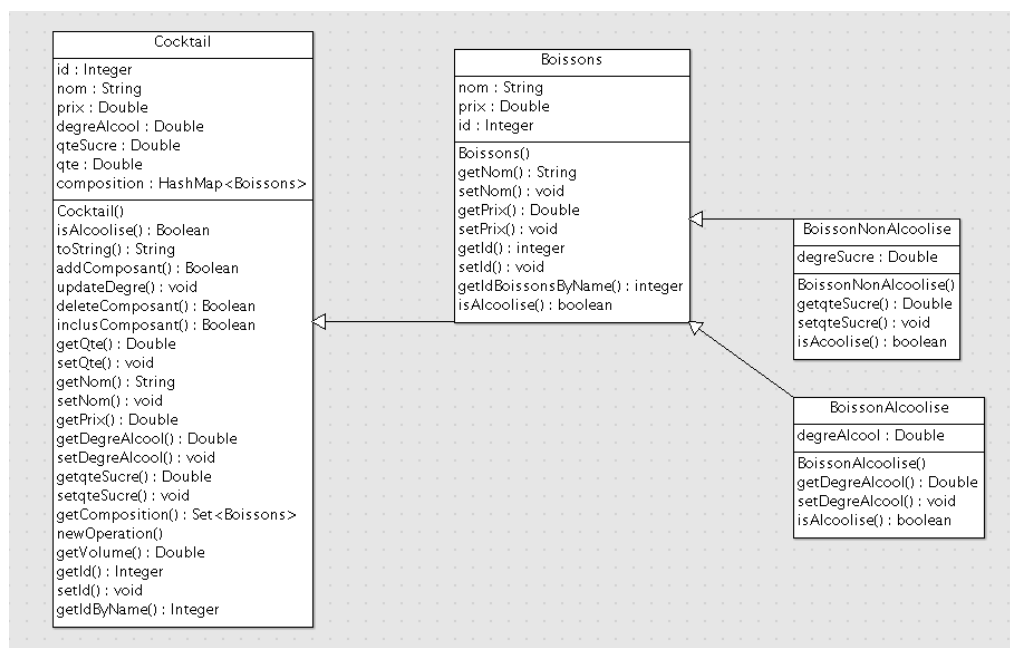


Diagramme de classe :



Héritage et Polymorphisme :

L'un des aspects important de ce logiciel réside dans l'héritage d'une classe par rapport à une autre.

Ici nous avons 3 classes :

- 'Boissons'
- 'BoissonAlcoolise'
- 'BoissonNonAlcoolise'

C'est trois classes sont lié entre elle. En effet il existe un héritage entre ces classes. C'est à dire que les attributs et méthodes de la classe mère 'Boissons' sont transférés aux classes filles 'BoissonNonAlcoolise' et 'BoissonElcoolise'.

Ainsi, la méthode 'getNom()' sur un objet 'BoissonAlcoolise', retournera le nom de la boisson.

```
BoissonNonAlcoolise Coca = new BoissonNonAlcoolise('Coca', 2,1,30) ;  
String nom = Coca.getNom() ;
```

Avec l'héritage, une autre notion est importante, il s'agit du polymorphisme. Il consiste à faire appel au constructeur de la classe mère dans la classe fille.

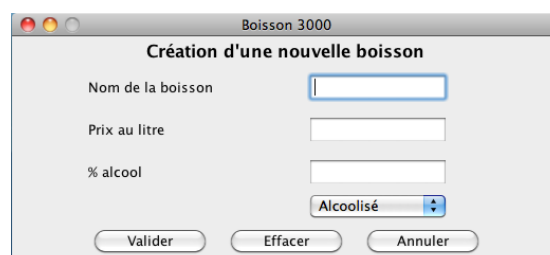
Comme dans l'exemple précédent, j'ai appelé le constructeur de la classe 'BoissonsNonAlcoolise' et passé des arguments qui sont le nom, le prix, l'id et le pourcentage de sucre.

```
public BoissonNonAlcoolise(String nom,double prix,int id, double degreSucre) {  
    super(nom,prix,id);  
    this.degreSucre = degreSucre;  
}
```

Le polymorphisme va appeler le constructeur de la classe 'Boissons' grâce à la fonction `super()`; et passera les argument nécessaire pour créer l'objet 'Boissons'.

```
public Boissons(String Nom,double Prix,int id) {  
    this(Nom,Prix);  
    this.Prix=Prix;  
    this.id=id;  
}
```

Aspect fonctionnel et technique :



L'ajout d'une boisson ce fait par l'intermédiaire d'une interface graphique. L'utilisateur saisi les informations nécessaires à la création d'une boisson et click sur le bouton 'Valider'. Lors de la création de l'affichage, nous avons défini un 'listener' qui permet de réceptionner les actions comme des clicks sur certains endroits. Ici, l'action sur le bouton valider va lancer

la récupération des valeurs des champs pour compléter les requêtes SQL d'insertion dans la base de données, puis va l'envoyer au serveur.

```
INSERT INTO Boissons(NomBoisson,prixBoisson) VALUE('nom',prix) ;
```

```
INSERT INTO BoissonsAlcoolise(idBoisson,degreAlcool) VALUE(id,degre);
```

Une fois les boissons ajoutées, nous pouvons créer un cocktail.

Dans cette interface, l'utilisateur choisie les composants de son cocktail. A l'aide des flèches il ajoute le composant sélectionné dans la liste à son cocktail. Lors de l'appui, un pop-up apparaît pour lui demander la quantité de boisson à ajouter au cocktail. Si le composant est déjà présent dans la liste, la quantité sera ajoutée à la quantité déjà présente.

L'évènement sur le bouton 'Valider' crée les requêtes d'insertion dans la base de données à partir des informations présentes.

```
INSERT INTO Cocktails(nomCocktail,prixCocktail) VALUES('nom',prix) ;
```

Une fois le cocktail créé dans la base, nous parcourons le HashMap qui contient la composition du cocktail pour l'insérer dans la table 'Compositions'

```
HashMap<Boissons,Double> compoCocktail = cocktailtmp.getCompositionFull();
Iterator<Boissons> ite = compoCocktail.keySet().iterator();
while(ite.hasNext()){
    boissonCompo = ite.next();
    qteBoisson = compoCocktail.get(boissonCompo);
    idBoisson = boissonCompo.getId();
    requeteCompo = "INSERT INTO Compositions
(idCocktail,idBoisson,qteComposition) VALUES (" + idCocktail + "," + idBoisson
+ "," + qteBoisson + ")";
}
```

Quand toute la collection de boisson est insérée dans la table, l'affichage se ferme. Le cocktail est donc en mémoire.

La page d'accueil récupère tous les cocktails présents dans la base et affiche leurs informations. L'affiche se fait par récupération des données dans des collections puis affichage de ces collections dans les JList et JText correspondant.

Bar 3000

Fichier Cocktails Boissons

Cocktails

Mojito
Martini Gin
Black Jack

Rafraichir

Composition

Boissons	Volume
Badoit	20.0 cl
Rhum	5.0 cl

Nouvelle
Modifier

Action sur les boissons

Information sur le cocktail

Prix 2,65 €
Volume 25,00
% Alcool 8,00
% Sucre 2,40

Action sur les cocktails

Nouveau Modifier Supprimer

Conclusion :

L'objet a toujours été mon talon d'Achille, mais grâce au développement de cette application, j'ai compris les fondements de la programmation orienté objet comme l'héritage, et le polymorphisme. L'utilisation des méthodes de classe permet de faire des actions relativement facilement tout en gardant un code clair et simple.

En plus de l'utilisation des méthodes spécifique à chaque classe, j'ai découvert aussi les collections qui permettent l'accès rapides à une banque de données.

Cette application m'a permis aussi de découvrir les interfaces graphiques et leur interaction avec le code.