# 

|  |  |
| --- | --- |
| Projet | GSB Backoffice Commercial  Produits |
| Document | Classes et méthodes de la Partie Produit |
| Rédacteur | Marvyn FRANCESCONI |
| Date | 12/03/2015 |

## Classe DAO Factory :

Cette classe a pour but de DAO, elle va faire le lien entre la base de données et l’application en permettant la connexion et l’exécution des requêtes.

* public void connexion()

**Fonction connexion() : Permet la connexion à la BDD.**

* public void deconnection()

**Fonction deconnection() : Permet la déconnection à la BDD.**

* public SqlDataReader execSql(string requete)

**Fonction execSql(Chaine de caractère requete) : Récupère une requête SQL en chaine de caractère et retourne un objet SQLDataReader.**

Exécute les requêtes SQL passé en paramètre

## Classe DAO Produit :

Cette classe va être la continuité du DAOFactory en faisant appel à elle pour récupérer les Objets de la base de données et envoyer des informations. Elle va stocker les requêtes de suppressions, modifications, création et va récupérer les objets produit pour les mettre dans une liste.

* public static List<Produit> listeProduit()

**Fonction listeProduit() : Retourne une Liste de Produit**

Récupère l'intégralité des produits dans la BDD, crée les objets et les stocke dans une liste.

* public static ArrayList listeFamille()

**Fonction listeFamille() : Retourne une Liste de Famille**

Récupère l'intégralité des familles dans la BDD, crée les objets et les stocke dans une liste.

* public static string creerProduit(Produit leProduit)

**Fonction creerProduit(Produit leProduit) : Récupère un produit et retourne le message d’échec ou de réussite.**

Ajoute le produit passé en paramètre à la BDD par une requête.

* public static string supprimerProduit(int num)

**Fonction supprimerProduit(entier num) : Récupère le numéro d’un produit et retourne un message d’échec ou de réussite.**

Supprime le produit correspondant au numéro de la BDD par une requête.

* public static string modifierProduit(Produit leProduit)

**Fonction modifierProduit(Produit le Produit) : Récupère un objet produit et retourne un message d’échec ou de réussite.**

Modifie le produit de la BDD correspondant au numéro du produit entré en paramètre, modifie tous les champs contenus dans produit.

* public static string getNomFamille(int idFamille)

**Fonction getNomFamille(entier idFamille) : Récupère un id d’objet Famille et retourne le nom correspondant à cet id.**

Permet la récupération d'un nom de famille de produit avec son id en passant par la BDD.

* public static int getIDFamille(string nomFamille)

**Fonction getIDFamille(Chaine de caractères nomFamille) : Récupère le nom d’une Famille de produit et retourne l’ID de l’objet famille correspondant à ce nom.**

Permet la récupération d'un ID de famille d'un produit par rapport à son nom par le biais d'une requête dans la BDD.

## Classe Produit :

Cette classe va regrouper le constructeur de l’objet Produit, ces accesseurs et ses méthodes, elle va faire appel à DAOProduit (et en aucun cas à DAOFactory) pour récupérer et faire passer la liste des Produits. Elle va aussi contenir des méthodes concernant les produits et les vérifications qui vont être utilisés dans la classe principale.

* public Produit(int unNum, string unNom, string unEffet, string desCI, string unePresentation, string unDosage, int unIDFamille, double unPrixHT, double unPrixEch, string uneInteraction)

**Constructeur de la Classe Produit.**

Récupère le Numéro de produit, son nom, son effet, ses contre-indications, sa présentation (Gélules, poudre…), son dosage (une fois par jour…), son IDFamille (sa famille de produit), son Prix Hors-taxes, son prix à l’échantillon, et son interactions (Ne pas conduire, éviter le paracétamol…)

* public static Boolean validerNumNom(int num, string nom)

**Fonction validerNumNom(entier num, chaine de caractères nom) : Récupère le numéro et le nom d’un objet produit et retourne un Booléen.**

Valide lors de la création / Modification si le Numéro et le nom de produit ne sont pas déjà utilisés dans un produit de la Liste lesProduits. Retourne False si le Nom ou le NUM sont déjà utilisés, sinon return True

* public static String creerProduit(Produit unProduit)

**Fonction creerProduit(Produit unProduit) : Récupère un objet produit et retourne un message de réussite ou d’échec quant à sa création dans la BDD suite à creerProduit() de DAOProduit.**

Après avoir vérifié que le Nom et le Num n'étaient pas déjà utilisés (return true à vérification), appel à la méthode DAOProduit pour en crée un nouveau.

* public static string supprimerProduit(Produit unProduit)

**Fonction supprimerProduit(Produit unProduit) : Récupére un objet produit et retourne un message de réussite ou d’échec quand à sa suppression dans la BDD suite à supprimerProduit() de DAOProduit.**

Suppression d'un produit et retourne le message de validation ou d'erreur.

* public static List<Produit> listeProduit()

**Fonction listeProduit() : Retourne une liste des Produits.**

Création de la liste Produit en provenance du DAOProduit où ils ont été récupérés par la méthode listeProduit().

* public static string modifierProduit(Produit unProduit)

**Fonction modifierProduit(Produit unProduit) : Récupère un objet produit et retourne un message de réussite ou d’échec quant à sa modification dans la BDD suite à modifierProduit() de DAOProduit.**

Modification d'un produit et retourne le message de validation ou d'erreur.

* public static ArrayList listeFamille()

**Fonction listeFamille() : Retourne une ArrayList de nom de Famille.**

Création de la list Famille en provenance du DAOProduit où ils ont été récupérés.

## Classe Principale

Création des listes de Produit et de Famille en début de page, initialisation des fonctionnalités de la page (ce qui doit apparaitre ou nom) et chargement des données.

Cette page définit l’interface graphique et ses actions, elle va être en lien seulement avec la Classe produit et utiliser ses méthodes, elle va juste permettre l’affichage et l’utilisation de celle-ci sans faire beaucoup de calculs.

### Procédure Evénementielle :

* private void comboBoxSelectionProd\_SelectedIndexChanged(object sender, EventArgs e)

Evénement au changement d’objet produit dans la ComboBox de choix de produit.

Charge ses éléments, le fait apparaitre dans les zones de textes appropriés et fait apparaitre les boutons Modifier et Supprimer.

* private void buttonCreerProd\_Click(object sender, EventArgs e)

Événement au click du bouton Créer, création du produit.

* private void buttonValiderProd\_Click(object sender, EventArgs e)

Evénement qui au click permet à la fois la validation d’un produit en cours de création mais aussi la validation lors de la modification d’un produit.

* private void buttonSupprimerProd\_Click(object sender, EventArgs e)

Événement au click du bouton supprimer, permet la suppression d’un produit sélectionné.

* private void buttonAnnulerProd\_Click(object sender, EventArgs e)

Au click, annule les actions en cours et vide les zones de textes.

* private void buttonModifierProd\_Click(object sender, EventArgs e)

Au click définit les cases non modifiable et les grise, bloque aussi certaines boutons.

* private void dataGridViewProd\_CellContentClick(object sender, DataGridViewCellEventArgs e)

Au click sur une cellule, va afficher l’interaction dans la zone de texte correspond au produit de celle-ci en allant la chercher dans la liste des produits.

* private void tabPage1\_Click(object sender, EventArgs e)

Remplit la ComboBox à l’ouverture de l’onglet.

### Procédure et Fonctions :

* public Boolean caseRemplies()

**Fonction caseRemplies() : Retourne un Booléen.**

Vérifie que toutes les cases sont bien remplis (notamment au niveau de la création d’un produit), retourne false si ce n’est pas le cas.

* public void majListeProduit()

**Procédure majListeProduit() : Vide et reremplit la ComboBox et le DataGriedView.**

* public void majListeFamilleProduit()

**Procédure majListeFamilleProduit() : Vide et reremplit la ComboBox concernant les familles de produits.**

* public void textBoxVide()

**Procédure textBoxVide() : Vide les zones de textes du formulaire.**