Exercices Winforms, ADO.Net et XML

Table des matières

[1 Construction d’une interface visuelle 2](#_Toc481485534)

[2 Récupération et affichage de données 3](#_Toc481485535)

[3 Création d’une saisie complète 4](#_Toc481485536)

[4 Requêtes de masse 7](#_Toc481485537)

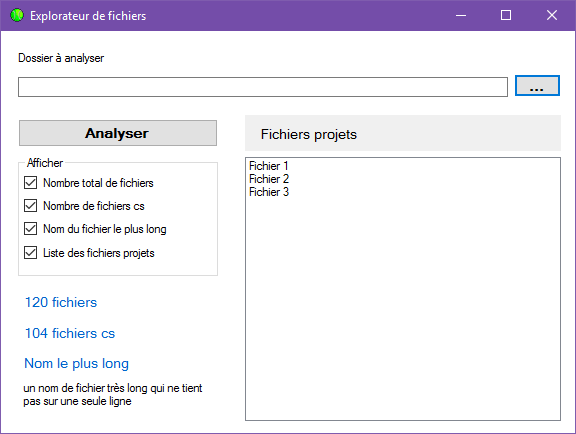
[5 XML 8](#_Toc481485538)

[5.1 Ecriture de fichiers xml 8](#_Toc481485539)

[5.2 Sérialisation 8](#_Toc481485540)

# Construction d’une interface visuelle

**Etape 1 :** Construire l’interface visuelle suivante :



**Etape 2** : Branchement de gestionnaires d’évènements

Lors du clic sur le bouton « … », afficher une boîte de dialogue de sélection de dossier, et affecter le fichier sélectionné à la zone de saisie (utiliser la classe FolderBrowserDialog)

Lors du clic sur les cases à cocher, afficher ou masquer les éléments visuels correspondants

**Etape 3** : Utilisation de paramètres d’application

Lors de la fermeture de l’application, faire en sorte que le dossier sélectionné, et l’état des cases à cocher soient mémorisé dans les paramètres utilisateur

**Etape 4** : message de confirmation

Lors de la fermeture de la fenêtre, afficher un message demandant si on veut enregistrer le chemin sélectionné et l’état des cases à cocher.

**Etape 5** : Utilisation de ressources

Rendre la fenêtre localisable en anglais. Tester en forçant la langue anglaise dans la méthode Main

Ajouter une icône dans la barre de titre de la fenêtre

**Etape 6** : logique applicative

Implémenter l’analyse du dossier sélectionner en utilisant le code de l’exercice de la veille (la séparation du visuel et de la logique applicative permet normalement de réutiliser les classes Analyseur et Explorateur sans modification)

# Récupération et affichage de données

Cet exercice utilise la base de données Northwind

**Etape 1** : Exécution d’une première requête

* Dans cotre solution Visual Studio, ajouter le projet ADO fourni
* Dans une classe DAL, créer une méthode pour récupérer la liste des régions. On se contentera dans un premier temps d’afficher les valeurs dans la fenêtre de sortie.
* Exécuter la méthode en debug pour vérifier les valeurs.

**Etape 2 :** Remplissage d’une DataGridView

* Créer une fenêtre fille nommée FormFournisseurs qui sera ouverte par clic sur un menu.
* Au chargement de cette fenêtre, exécuter une requête pour récupérer la liste des fournisseurs (tous les champs, sauf les 3 derniers)
* Afficher le résultat dans une DataGridView.

**Etape 3** : Remplissage d’une liste déroulante

Dans la même fenêtre, ajouter une liste déroulante Pays, et la charger à partir du résultat d’une requête qui récupère les différents pays des fournisseurs triés par ordre alphabétique

**Etape 4** : Requête paramétrable

Rendre la requête de récupération des fournisseurs paramétrable selon le pays

Dans la fenêtre précédente, lors de la sélection d’un pays dans la liste déroulante, exécuter cette requête en lui passant la valeur sélectionnée, puis afficher son résultat dans la DataGridView

NB/ La DataGridView doit afficher une liste filtrée dès l’ouverture de la fenêtre.

**Etape 5** : Requête scalaire

Dans la fenêtre précédente, ajouter un libellé qui affiche le nombre de produits fournis par l’ensemble des fournisseurs du pays sélectionné

**Etape 6** : Exécution d’une fonction

Dans la base de données, créer une fonction qui renvoie la liste des commandes passées par un client. Cette fonction prend l’Id client en paramètre, et renvoie pour chaque commande :   
Id, Date, Date d’envoi, nombre d’articles, Montant total, Frais d’envoi (tri selon la date)

Dans l’application, créer une nouvelle fenêtre FormCommandes, ouverte par clic sur un nouveau menu, et ajouter une zone de saisie d’un code client, un bouton « Voir les commandes » et une DataGridview.

Faire en sorte que lorsqu’on clique sur le bouton, la DatagridView affiche la liste des commandes du client dont on a saisi le code

**Etape 7** : Maître / détail

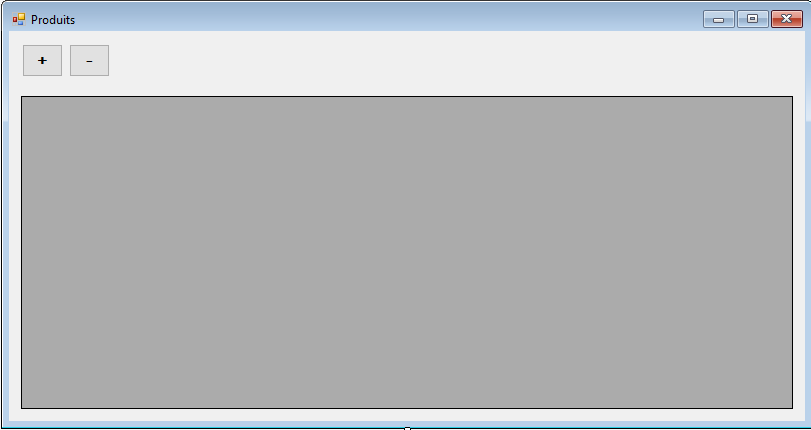
Supprimer le bouton et remplacer la zone de saisie par une ListBox affichant la liste complète des clients (code, nom et prénom).

Lorsqu’on double-clique sur un client, afficher sa liste de commandes.

# Création d’une saisie complète

Pour cet exercice, continuer à utiliser la base de données Northwind et le projet ADO créé dans l’exercice précédent.

L’objectif de cet exercice est de créer une saisie complète de produits, avec une fenêtre affichant la liste des produits, et une fenêtre de saisie modale.



* Le bouton + permettra d’ajouter un nouveau produit en le saisissant dans une fenêtre modale
* Le bouton – permettra de supprimer le produit sélectionné dans la liste

**Etape 1** : fenêtre modale de création d’un nouveau produit

Créer une fenêtre nommée FormSaisieProduit, ressemblant à ce qui suit :

|  |  |
| --- | --- |
|  | Affecter sa propriété ControlBox à False pour masquer les boutons et l’icône de la barre de titre  Affecter ses propriétés AcceptButton et CancelButton  Affecter également la propriété DialogResult des boutons OK et Annuler. Ceci permet de fermer la fenêtre en renvoyant le résultat.  Utiliser des contrôles de type MaskedTextBox pour es zones de saisie numérique |

Ajouter à la fenêtre une propriété ProduitSaisi de type Produit.

A la fermeture de la fenêtre (redéfinir OnClosing), si l’utilisateur a cliqué sur OK (tester la propriété DialogResult), initialiser cette propriété avec une nouvelle instance de Produit, dont les propriétés ont les valeurs saisies dans la fenêtre.

NB/ L’Id du produit sera déterminé au moment de l’insertion dans la base de données (champ Identity). On ne le fait donc pas saisir à l’utilisateur.

**Etape 2** : Fenêtre d’affichage des produits

Créer une fenêtre fille FormProduits, ouverte par clic sur un menu Produits. Cette fenêtre doit afficher dans une DataGridView la liste complète des produits, avec les mêmes informations que dans la fenêtre modale créée précédemment + l’id des produits.

**/!\** Dans la DAL, on utilisera une collection de type BindingList< > pour renvoyer la liste des produits.

**Etape 3** : Ajout d’un nouveau produit

Sur la fenêtre FormProduits, ajouter le bouton « + », permettant d’ouvrir la fenêtre modale créée précédemment.

A la fermeture de la fenêtre modale, récupérer le produit saisi et l’ajouter à la collection qui sert de source de données à la DataGridView. Pour cela, utiliser un code du type :

using (var form = new FormSaisieProduit())

{

   DialogResult dr = form.ShowDialog();

   if (dr == DialogResult.OK)

   {

      ...

   }

}

A l’exécution, vous constatez que le nouveau produit ajouté apparaît automatiquement dans la DataGridView. Ceci se fait grâce à l’utilisation de la BindingList<>.

**Etape 4** : Suppression du produit sélectionné

Sur la fenêtre FormProduits, ajouter le bouton « - », permettant de supprimer le produit sélectionné dans la grille.

Pour réaliser cette suppression, il suffit de supprimer le produit de la BindingList. Pour récupérer le produit sélectionné, utiliser la propriété CurrentRow.DataBoundItem de la DataGridView.

**Etape 5** : Enregistrement en base de données

Dans la DAL, créer les méthodes nécessaires pour l’ajout et la suppression d’un produit.

Appeler ces méthodes sur les clics des boutons + et – de la fenêtre des produits.

**Etape 6** : Gestion des erreurs

Lorsqu’on tente de supprimer un produit qui est référencé par une ligne de commande, l’application plante. Pour y remédier :

* Retrouver le numéro de l’erreur SQL correspondante.
* Intercepter cette erreur en affichant un message pour indiquer à l’utilisateur que le produit ne peut pas être supprimé.

/!\ Gérer uniquement ce numéro d’erreur, et non l’ensemble des erreurs de type System.Data.SqlClient.SqlException

**Etape 7** : Transaction

Dans la méthode de DAL qui ajoute un nouveau produit, ajouter une commande qui crée automatiquement une catégorie « Autres produits » si la catégorie du produit n’existe pas déjà.

Placer les deux commandes dans une transaction

Tenter d’ajouter un produit avec un id de catégorie et un id de fournisseur qui n’existent pas. Vérifier que la transaction est bien annulée et que la catégorie « Autre » n’est pas créée.

Tenter d’ajouter un produit avec un id de catégorie qui n’existe pas, et un id de fournisseur qui existe bien. Vérifier que la transaction est bien validée et que la catégorie « Autre » est créée.

**Etape 8** : Listes déroulantes

Dans la fenêtre modale de création d'un nouveau produit, remplacer la zone de saisie de l'Id de la catégorie par une liste déroulante affichant les libellés des catégories

De même, remplacer la zone de saisie de l'Id du fournisseur par une liste déroulante affichant les noms des fournisseurs.

**Etape 9** : Maître-détail

Dans la fenêtre des fournisseurs, ajouter une seconde DataGridView au-dessous de la première, pour afficher les produits du fournisseur sélectionné   
L'ensemble des produits de tous les fournisseurs devra être récupéré en une seule requête (utiliser la méthode de DAL déjà faite pour cela).   
Lorsqu'on sélectionne un fournisseur, faire en sorte d'afficher uniquement les produits de ce fournisseur.

# Requêtes de masse

Pour cet exercice, continuer à utiliser la base de données Northwind, et le projet ADO créé dans l’exercice précédent.

**Etape 1 : Mise en place de l’enregistrement global**

Pour optimiser les traitements, on ne souhaite plus enregistrer les ajouts et suppressions de produits à chaque clic sur les boutons + ou -, mais plutôt de façon globale quand on clique sur un bouton Enregistrer. Pour cela, dans la fenêtre FormProduits :

* Ajouter un bouton « Enregistrer » à côté des boutons + et –
* Créer 2 listes privées nommées \_produitsAjoutés et \_produitsSupprimés, destinées à stocker les produits à ajouter et à supprimer avant de faire l’enregistrement en base
* Dans les gestionnaires d’évènements clic des boutons + et -, remplacer les appels aux méthodes de DAL par l’ajout des produits dans les listes crées précédemment.   
  NB/ La DataGridView doit continuer à refléter l’ajout et la suppression des produits, même s’ils ne sont pas encore enregistrés en base.

**Etape 2 : Création du type table pour l’insertion de produits en masse**

Dans la base, créer un type table TypeTableProduit qui vous permettra de stocker en mémoire les produits à ajouter.

**Etape 3 : Création de la méthode d’insertion en masse**

Créer une méthode de DAL prenant en paramètre une liste de produits, et réalisant l’insertion de ces produits en masse dans la base. Cette méthode utilisera le type table créé précédemment et exécutera la requête au sein d’une transaction.

**Etape 4 : Appel de la méthode d’insertion**

Sur le clic du bouton Enregistrer, appeler la méthode de DAL que vous venez de créer, en lui passant en paramètre la liste des produits ajoutés.

Tester en créant plusieurs produits d’affilée, puis en cliquant sur Enregistrer. Vérifier dans la base que les produits ont bien été ajoutés.

**Etape 5 : Rafraîchissement de la liste**

Pour rafraîchir les Id des produits dans la DataGridView, après appel de la méthode d’insertion de masse, faire en sorte de recharger la liste complète des produits.

**Etape 6 : Création du type table pour la suppression de produits en masse**

Pour supprimer des produits, on n’a besoin que de leurs identifiants. Créer un type table TypeTableId avec une seule colonne de type entier.

**Etape 7 : Création de la méthode de suppression en masse**

Au lieu de supprimer les produits, nous allons simplement les rendre obsolètes, c’est-à-dire passer leur champ Discontinued à 1.

Créer une méthode de DAL prenant en paramètre une liste d’identifiants de produits, et réalisant leur mise à jour en masse dans la base. Cette méthode utilisera le type table créé précédemment, et exécutera la requête au sein d’une transaction.

**Etape 8 : Modification de la requête de liste des produits**

Dans la méthode de DAL qui récupère la liste des produits, ajouter une clause where à la requête, de façon à ne récupérer que les produits qui ne sont pas obsolètes

**Etape 9 : Appel de la méthode de suppression**

Sur le clic du bouton Enregistrer, appeler la méthode de DAL de suppression des produits, en lui passant en paramètre la liste des produits supprimés.

Tester en supprimant plusieurs produits d’affilée, puis en cliquant sur Enregistrer. La liste des produits étant rechargée complètement à chaque clic sur ce bouton, si la suppression s’est bien passée, les produits supprimés ne doivent pas réapparaître.

**Etape 10 : bravo, vous pouvez partir en week-end !!**

# XML

## Ecriture de fichiers xml

Ecrire un petit fichier xml dans Notepad++ pour représenter les clients, commandes et lignes de commandes du schéma Northwind.

Remarques :

* Inutile de mettre toutes les informations de chaque entité.
* A l’intérieur des éléments de type collection, mettre simplement 3 ou 4 lignes de données

Faire la même chose avec les employés, territoires et les associations entre les deux.

## Sérialisation

Sérialiser dans un fichier xml la liste des commandes avec le détail de leurs lignes de commandes, en utilisant un XmlSerializer.

Ecrire également la méthode de désérialisation

## Désérialisation avec Linq To XML

En repartant du fichier xml généré précédemment et des entités déjà créées pour les commandes et lignes de commandes, désérialiser le fichier en utilisant LinqToXml.

Conclusion ?

## Requêtes avec Linq To XML