C# - FICHE PRATIQUE N° 7

Une petite gestion de personnel
Grille de données, chargement partiel d'un dataset.

A/ Grille de données

1. Principe

Les enregistrements d'un DataTable peuvent être édités d'une manière rapide à l'aide d'un composant DataGrid (grille de données).

> Du point de vue des données, ce composant possède deux propriétés importantes :

- DataSource :

La valeur de cette propriété peut être une DataTable ou un DataView. Dans ce cas, la grille affichera l'ensemble des colonnes de la table ou de la vue. La valeur de la propriété DataMember doit être une chaîne vide.

La valeur peut également être un DataSet ou un DataViewManager. Dans ce cas, la valeur de la propriété DataMember doit correspondre au nom de l'une des tables du DataSet ou du DataViewManager. Si DataMember est une chaîne vide, l'ensemble des tables peuvent être éditées à l'aide de la grille (le mieux est d'essayer…).

- DataMember :

Si la propriété DataSource est une DataTable ou un DataView, cette propriété ne doit pas être renseignée.

Si la propriété DataSource est un DataSet ou un DataViewManager, cette propriété doit être le nom de l'un des tables du DataSource (ou une chaîne vide pour permettre l'édition de toutes les tables).

Ces deux propriétés peuvent être renseignées graphiquement si les composants d'accès aux données sont situés sur le même formulaire. Dans le cas contraire, il faut le faire par programme. Ceci peut par exemple être fait dans le constructeur du formulaire contenant la grille.

> Paramétrage de la présentation :

La présentation d'un DataGrid peut être paramétrée de manière graphique en ce qui concerne les couleurs, la possibilité de trier les enregistrements, … Bien sûr, il est également possible de le faire par programme.

En ce qui concerne les colonnes de la grille, il est possible de les paramétrer de manière graphique si les composants d'accès aux données sont présents sur le formulaire. Dans le cas contraire, il faut le faire par programme et cette opération est assez lourde.

2. Mise en place du DataGrid

- Créez un nouveau formulaire Fm_grille, mettez en place la possibilité de l'ouvrir depuis le formulaire principal de la même manière que les autres.

- Créez sur ce formulaire deux TextBox destinées à éditer le code et la désignation des services.

- Ajoutez un dbNavigateur pour les services.

- Ajoutez le code nécessaire pour lier tous ces éléments à la table service à l'aide d'un DataViewManager.

- Paramétrer la propriété « ApplyDefaultSort » à vrai pour la table « employé ».

- Ajoutez un composant DataGrid et paramétrez-le par programme pour qu'il affiche les employés du service en cours d'édition (DataSource : votre DataViewManager, DataMember : tp1_service.serviceemploye).

- Testez le tout et essayez les diverses possibilités de paramétrage de la grille (l'option « mise en forme automatique » du menu contextuel de la grille permet de gagner du temps).

3. Paramétrage des colonnes

Il est possible de gérer individuellement chaque colonne affichée par la grille. Le principe est d'associer une collection d'objet DataGridStyle à la grille de données. Chacun de ces styles est utilisé pour afficher les données de l'une des tables du DataSet.

Un DataGridStyle est lié à une table du DataSource de la grille par sa propriété MappingName. Un DataGridStyle possède une collection de DataGridColumnStyle, chacun étant lié à une colonne de la table par l'intermédiaire de sa propriété MappingName. Les objets DataGridStyle et DataGridColumnStyle possèdent des propriétés qui permettent de paramétrer l'affichage de manière assez fine.

Créer ces objets par programme est un peu fastidieux (mais peut parfois être nécessaire). Pour le faire en mode graphique, les composants d'accès aux données doivent se situer sur le même formulaire, ce qui n'est pas le cas dans notre application. Voilà une solution pour « tricher un peu » :

- Copiez votre DataSet depuis le formulaire principal et collez-le sur le formulaire fm_grille. Ce DataSet a la même structure que le DataViewManager créé par programme, nous allons l'utiliser pour le paramétrage de la grille.

- Affectez ce nouveau DataSet à la propriété DataSource de la grille, et la relation service.serviceemploye à la propriété DataMember (ces valeurs seront remplacées lors de l'exécution de votre constructeur).

- La propriété TableStyle de la grille représente la collection de DataGridStyle. Ajoutez en un en le reliant à la table employé. Attention, il semblerait logique de relier le style à la relation service.serviceemploye, mais cela ne fonctionne pas correctement. Renommez le style créé correctement.

- La propriété GridColumnStyles représente la collection des colonnes à afficher. Ajoutez celles que vous souhaitez voir apparaître dans la grille, renommez-les et paramétrez leur présentation (la propriété « HeaderText » permet de donner un titre à une colonne). Attention, leur propriété MappingName doit être liée aux champs de la table employé et non à ceux de la relation serviceemploye même si cela semblerait logique. Déroulez la liste accolée au bouton Ajouter, par curiosité…

- Testez votre grille à l'exécution. Attention, remarquez bien que ses propriétés DataSource et DataMember sont redéfinies par programme et ne correspondent donc pas en réalité à la copie du DataSet qui à servi au paramétrage de la présentation (celui-ci n'est d'ailleurs jamais rempli !).

- Lorsque vous êtes satisfait du résultat, vous pourriez supprimer la copie du DataSet (elle pourra être recréée en cas de besoin). Ceci dit, cet objet ne prend aucune place puisqu'il n'est jamais rempli… Autant le laisser, il peut servir.

4. Modification des données à l'aide de la grille

Assurez-vous que votre grille affiche bien toutes les colonnes de la table, hormis le champ calculé recherche (la colonne « cadre » peut être affichée sous la forme d'une case à cocher) et testez la modification des données à l'aide de la grille. Quittez votre application puis relancez la : les modifications effectuées ont été répercutées dans la base de données SQLServer, toujours à l'aide des méthodes du formulaire principal gérant les événements « RowChanged » et « RowDeleted ».

De la même manière, lors de l'ajout d'un enregistrement, le numéro de l'employé est fixé par défaut à la valeur « auto » (définie dans la structure du DataSet), puis obtenu automatiquement à l'aide de la

procédure stockée. Le mieux est de paramétrer cette colonne en lecture seule (propriété « ReadOnly »).

5. Ajout d'un navigateur

Ajoutez un dbNavigateur et initialisez-le de manière à ce qu'il soit lié à la relation service.serviceemploye de votre DataViewManager. Exécutez l'application : le navigateur et la grille de données sont liés.

Il serait souhaitable de sélectionner la colonne « nom » lors de l'ajout d'un employé par un clic sur le bouton « plus » du navigateur.

Il suffit pour cela d'initialiser le navigateur en lui passant en paramètre la méthode surAjout dont le code est le suivant :

```
private void surAjout()
{
    dbGr_employe.Focus() ;
    dbGr_employe.CurrentCell = new DataGridCell(dbGr_employe.CurrentCell.RowNumber,1);
}
```

Cette méthode active la seconde cellule de la ligne en cours d'édition.

B/ Chargement partiel d'un dataSet

Dans le cas d'une base de données importante, le chargement de l'ensemble des données dans le dataSet peut être coûteux. Il est possible de l'éviter en ne chargeant que quelques enregistrements d'une table (ou du résultat d'une requête) :

> Une première solution consiste à passer deux paramètres en plus à la méthode Fill du DataAdapter :

```
dbAd_employe.Fill(dbDs_empSce1,0,10,"tp1_employe");
```

Cet appel charge 10 enregistrements à partir du premier (rang 0). Il est ainsi possible de balayer toute la table par lots de 10 enregistrements en travaillant avec une variable entière « pos » modifiée à l'aide de deux boutons « enregistrements précédents » et « enregistrements suivants ».

> Une seconde solution consiste à filtrer les enregistrements à charger à l'aide de la requête select du DataAdapter. Voici l'exemple d'un DataAdapter (dbAd_employeFiltre) qui permet de ne charger que les employés d'un certain sexe :

. La commande select est la suivante :

```
SELECT    numero, nom, prenom, sexe, cadre, salaire, sce
FROM      tp1_employe
WHERE    (sexe = @p_sexe)
```

. Le paramètre doit être renseigné avant l'appel de la méthode Fill :

```
dbAd_employeFiltre.SelectCommand.Parameters["@p_sexe"].Value="m";
dbAd_employeFiltre.Fill(dbDs_empSce1,"tp1_employe");
```

Ici, seuls les employés de sexe masculin seront chargés depuis la base de données.

Evidemment, quelque soit la solution envisagée, les choses sont un peu plus complexes…

Remarque : ce problème ne survient qu'avec des tables réellement volumineuses ou de mauvaises conditions réseau. A titre d'exemple, une table de 137 000 lignes pesant 43 Mo met 7 secondes à se charger sur ma machine depuis le SGDB local, et 8 secondes à travers un réseau à 100 Mbits normalement chargé.