P6

Sacha COLBERT, Dylan BEROUD, Xaver FERBER, Nicolas MAS

24 Novembre 2023

Contents

1 Contextualisation :

Nous avons conçu une application de gestion de clients et d'adresses pour une entreprise.
Malgré des efforts pour assurer la qualité, un dysfonctionnement est apparu. Un utilisateur a signalé que lors de la création d'une fiche client, les adresses enregistrées ne correspondaient pas au client.
Après enquête, il s'avère que les adresses appartiennent à un autre client créé simultanément par un collègue. La résolution de ce problème est maintenant nécessaire.

2 Problématique :

Comment résoudre le problème signalé par le client ?

3 Contraintes :

3.1 Code initial désorganisé

3.2 Recréer le code de l'inteface

3.3 SQL Server Mangement Studio à prendre en main

4 Plan d'action :

4.1 Analyse des ressources à disposition

4.2 Suivre le super cours de monsieur Planas

4.3 Réaliser la base de donnée

4.4 Réaliser l'interface

4.5 Récupération de la base de donnée par l'interface

5 Solution :

Nous avons décider de diviser notre programme en 3 parties, cela évite d'avoir à recréer des scripts pour connecter l'interface à la base de donnée à chaque changement d'interface.

5.1 SQL

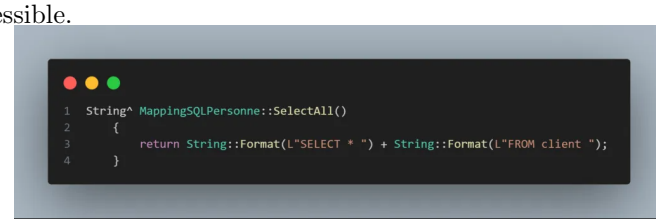
La partie SQL est chargé de mettre en place la connexion entre la base de donnée.

```
DataSet^ SQLAccess::GetRows(String^ sql, String^ dataTableName)
    {
        DataSet^ result = gcnew DataSet();
        SqlConnection^ sqlConnection = gcnew SqlConnection(_connectionString);
        SqlCommand^ sqlCommand = gcnew SqlCommand(sql, sqlConnection);
        SqlDataAdapter^ sqlDataAdapter = gcnew SqlDataAdapter(sqlCommand);
        sqlDataAdapter->Fill(result, dataTableName);
        delete sqlConnection;
        return result;
    }
```

```
void SQLAccess::ExecuteNonQuery(String^ sql)
    {
        SqlConnection^ sqlConnection = gcnew SqlConnection(_connectionString);
        SqlCommand^ sqlCommand = gcnew SqlCommand(sql, sqlConnection);
        sqlConnection->Open();
        sqlCommand->ExecuteNonQuery();
        delete sqlConnection;
    }
```

5.2 Mapping SQL

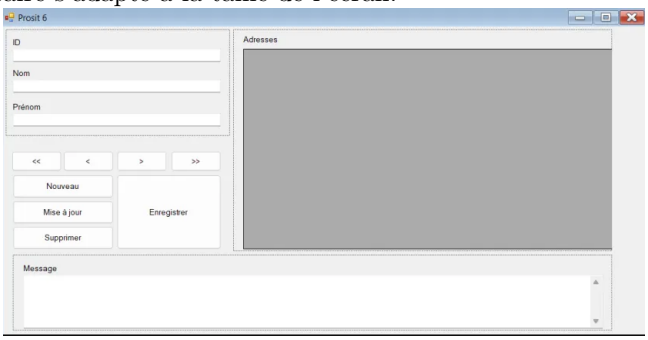
Le mapping SQL permet de créer et gérer des requêtes SQL et les rendre facilements accessible.



```
String^ MappingSQLPersonne::SelectAll()
    {
        return String::Format(L"SELECT * ") + String::Format(L"FROM client ");
    }
```

5.3 FrmMain

Le FrmMain est la partie correspondante à l'interface graphique, nous avons également regroupé certains éléments entre eux dans des panels pour que la taille nécessaire s'adapte à la taille de l'écran.



6 Conclusion :

Ce problème a mis en lumière le besoin de fragmenter le code pour une programmation durable.