MySQL Connexion C#

Jonathan Dale

11.01.2022



# Introduction

Le but de ce projet est de créer une application en C# qui permet de se connecter à une base de données MySQL et visualiser les différentes bases de données, tables et entrées de la table.

Le temps à disposition initial est 5 après-midi répandus sur 5 semaines.

# Table des matières

Table des matières

[1 Introduction 2](#_Toc98244503)

[2 Table des matières 2](#_Toc98244504)

[3 Analyse de fonctionnalités demandées 3](#_Toc98244505)

[3.1 Connexion avec son SGBD 3](#_Toc98244506)

[4 Maquette du projet 4](#_Toc98244507)

[5 Test effectués 4](#_Toc98244508)

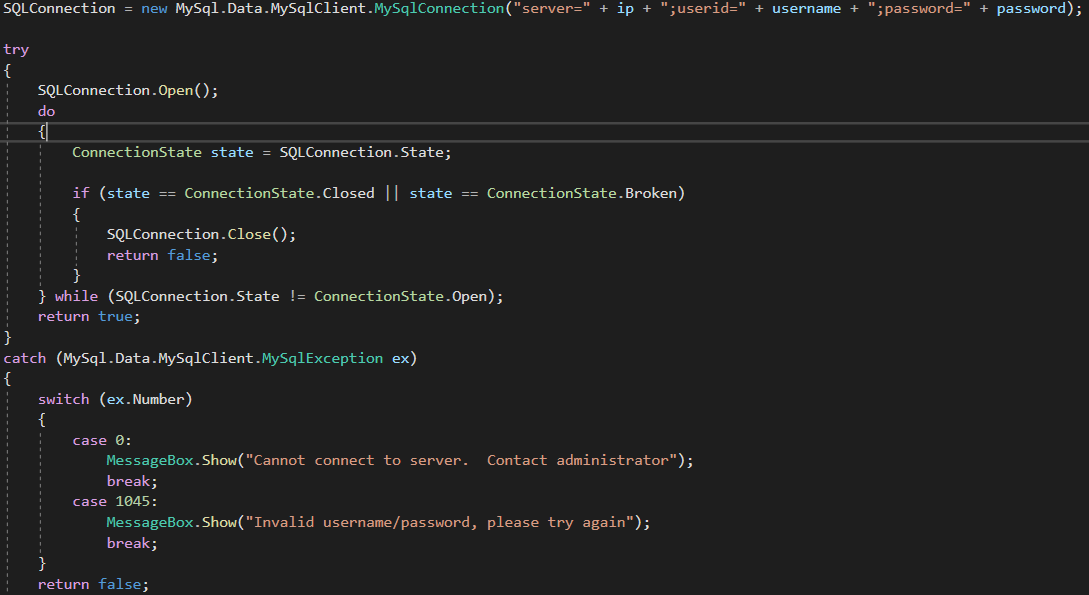
[6 Conclusion 4](#_Toc98244509)

[7 Informations pour la modification (login, IP, mdp, etc.) 5](#_Toc98244510)

# Analyse de fonctionnalités demandées

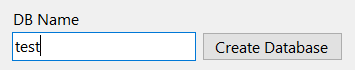
## Connexion avec son SGBD

Pour me connecter à la base de données MySQL j’ai dû instancier un objet MySql.Data.MySqlClient.MySqlConnection en utilisant une chaine de caractère pour la connexion comme paramètre du constructeur de l’objet et utiliser une boucle do while avec un try catch pour vérifier l’état de la connexion et agir en fonction du résultat de l’essai de connexion tout en évitant un crash de l’application entière.

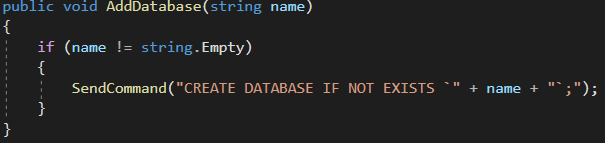


## Création/suppression d’une base de données

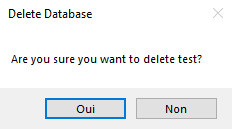
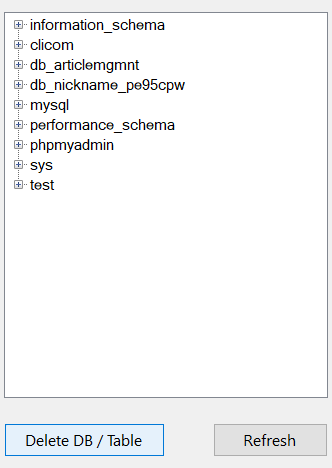
Pour créer et supprimer une base de données, j’ai créé des fonctions dans le fichier Network.cs. Pour la création il suffit de remplir une boite de texte et appuyer sur le bouton créer.



Ceci appelle la méthode AddDatabase(string name) et crée une base de données



Pour la suppression, il suffit de cliquer dans le TreeView sur la table ou la database et appuyer sur le bouton Delete DB/Table

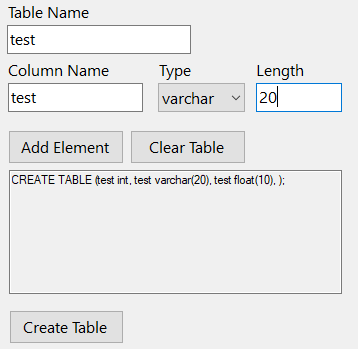


Ensuite il suffit de confirmer en appuyant sur Oui

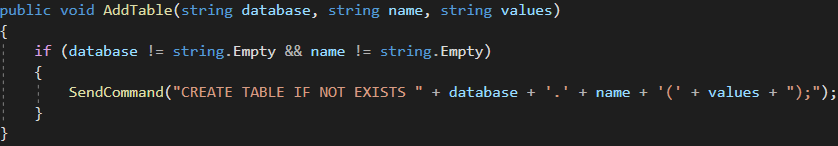
## Création de tables

### Insertion des données

Pour insérer les données, j’ai fait un système assez simple qui consiste en l’utilisation de boutons et d’une liste déroulante pour créer une commande qui va servir à créer une table avec ses attributs.



Ceci crée un nouvel objet TableCommand qui va utiliser des informations les informations de l’utilisateur et créer une commande qui va être envoyée dans Network.cs dans la fonction AddTable



### Suppression des données

J’ai réussi à faire la suppression des tables mais pas celle des données dans la table

### Mise à jour des données

Même histoire mais j’ai seulement commencé à créer le Form d’affichage des données.

## Moteur de recherche

Non implémenté.

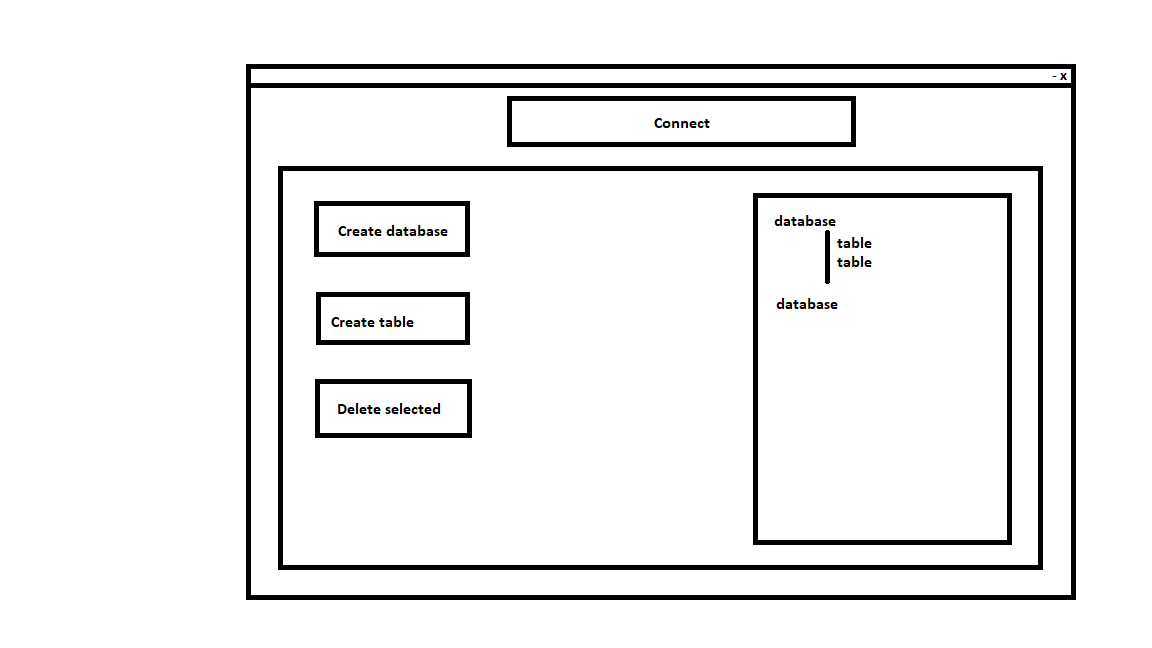
## Création d’un utilisateur

Non implémenté.

## Chargement des données via CSV

Non implémenté.

# Maquette du projet



# Test effectués

Pour les tests effectués, je dirais que celui qui m’a été le plus important c’est le test de connexion à la base de données. J’ai découvert qu’il y a une énumération de statuts de connexion qui s’appelle ConnectionState et que l’objet SQLConnection contient un membre Status qui indique le statut de la base de données.

Après ça il y a eu les tests pour les requêtes et pour voir si elles avaient marché. Pour faire cela j’ai dû mettre en place l’élément TreeView et le mettre à jour après chaque fin de requête.

# Conclusion

Pour conclure tout cela, je pense qu’il y avait des éléments assez durs à mettre en place et que je n’ai pas terminé comme le fait de pouvoir voir et ajouter des entrées au tables en instanciant un nouvel objet Form. Sinon le projet a été grandement simplifié par l’utlisation du package C# MySQL.

# Informations pour la modification (login, IP, mdp, etc.)

Afin de modifier les valeurs du projet, il y a quelques petites choses à savoir.

Première, pour la connexion à la base de données, il y a des constantes après la déclaration de classe frmMain dans le code du fichier MySQLApp.cs

Dans le cadre de ce projet il n’y avait aucun besoin d’utiliser un nouveau compte donc le compte root est utilisé par défaut.

Sinon à propos du code il y a une structure :

Le point d’entrée est situé dans Program.cs instancie ensuite un nouvel objet frmMain qui gère l’aspect visuel du projet et exploite les autres fichiers. Le fichier le plus exploité et Network.cs qui manie les requêtes SQL. Par exemple, pour stocker les databases et tables dans le programme, un nouvel objet Database et créé à partir d’une réponse du serveur et objets Table sont créés dans l’objet Database.