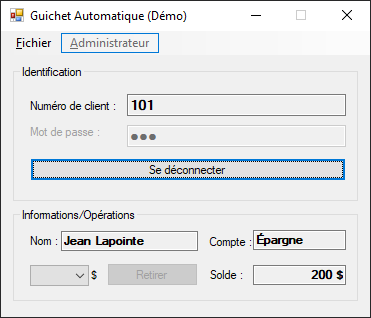
simulation guichet automatique

OBJECTIFs

* Créer et utiliser des classes et des sous-formulaires.
* Redéfinir la méthode *ToString* d’un objet.
* Utiliser des types par énumération.
* Lancer des Exceptions
* Valider des données et utiliser le contrôle *ErrorProvider*
* Afficher des objets dans une *ListView*.
* Manipuler des dates de type *DateTime.*

Description DE l’APPLICATION

Simuler l’utilisation d’un guichet automatique. Un client peut se connecter avec son numéro et son mot de passe. Il peut retirer de l’argent si son solde le permet. L’administrateur du guichet peut visualiser la liste des clients ou la liste des transactions effectuées sur le guichet.



INTERFACE

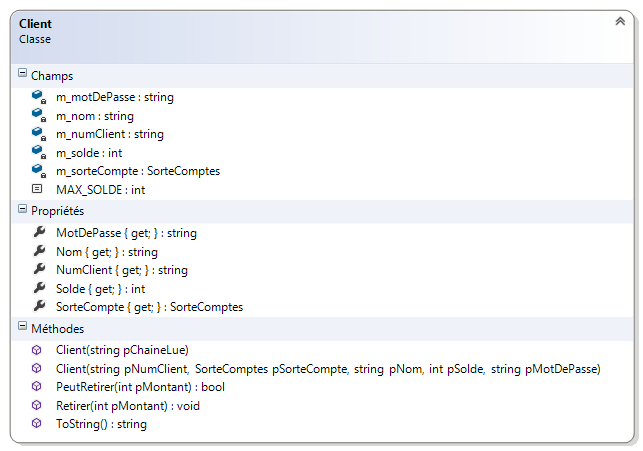
Voici un exemple où le numéro de client était valide et donc a pu se connecter. Voir la démo pour le cas où le numéro du client et/ou le mot de passe ne sont pas valides.

Étapes de réalisations

***Noter****: Vous ne devez pas vous préoccuper du mécanisme de connexion avant d’avoir préalablement réalisé les étapes 1 à 5*

Étape #1 (15%): définir et tester la classe **Client**

* Utiliser le diagramme ci-dessous pour définir la classe Client



Consignes pour Client :

Le MAX\_SOLDE est de 1 million $.

Dans le 1er constructeur, le paramètre **pChaineLue** contient les informations de chacun des champs séparés par une virgule.

Voici un exemple : *1RL,2,Robert Laporte,5000,abc*

No du client, Sorte de compte, Nom, Solde, Mot de passe

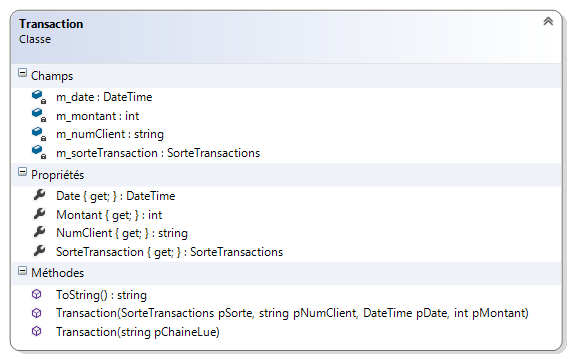
La méthode **ToString()** permet de former une chaine de caractères contenant les informations de chacune des variables membres séparés par une virgule.

Les exceptions possibles sont ArgumentOutOfRangeException

**Tests pour l’étape #1 : A01 à A05**

Étape #2 (15%) : définir et tester la classe **Transaction**

* Utiliser le diagramme ci-dessous pour définir la classe Transaction



Consignes pour Transaction :

La méthode **ToString()** permet de former une chaine de caractères contenant les informations de chacune des variables membres séparés par une virgule. Voici un exemple :

*2,101,2009-01-23 15:07:25,100*

Sorte de transaction, No du client, Date et heure, Montant

Le format pour la date est le suivant :

"yyyy-MM-dd HH:mm:ss"

**Tests pour l’étape #2 : B01 à B05**

Étape #3 (20%) : lecture et écriture dans les fichiers texte des 2 collections

* Dans le formulaire principal, compléter les méthodes suivantes. Utiliser les constantes.
  + private void ChargerCollectionClients()
  + private void ChargerCollectionTransactions()
  + public void EnregistrerCollectionClients()
* Compléter le constructeur du formulaire principal afin de lire les 2 fichiers et stocker les données dans les 2 collections.

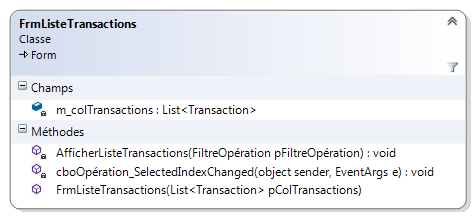
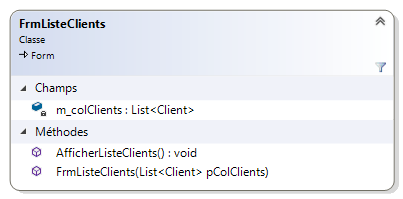
**Tests pour l’étape #3 : F01 à F03**

NOTE IMPORTANTE :

À partir de ce stade, vous pouvez commencer à faire fonctionner vos formulaires.

Utilisez les tests uniquement lorsque vous aurez terminé vos formulaires.

Étape #4 : compléter les 2 sous-formulaires et le menu Administrateur



* Compléter le code des sous-formulaires tel que décrit dans les diagrammes de classe ci-dessus :
* Le menu **Administrateur/Liste de clients** est disponible pour l’administrateur seulement. On affiche la collection des clients dans le sous-formulaire **FrmListeClients.** Voir la démo.
* Le menu **Administrateur/Liste des transactions** est disponible pour l’administrateur seulement. On affiche la collection filtrée des transactions en ordre décroissant de date dans le sous-formulaire **FrmListeTransactions.**

**Tests pour l’étape #4 : F04 à F06**

Étape #5 : le bouton Connexion/Déconnexion dans le formulaire principal

* Compléter le code du gestionnaire btnConnexion\_Click(…)
* Compléter les méthodes suivantes

public void EnregistrerLaTransaction(SorteTransactions pSorteTransaction,

int pMontantRetrait)

public Client RechercherClient(string pNumClient)

**Algorithme de validation du numéro de client et de son mot de passe**

**Tests pour l’étape #5 : F07 à F11**

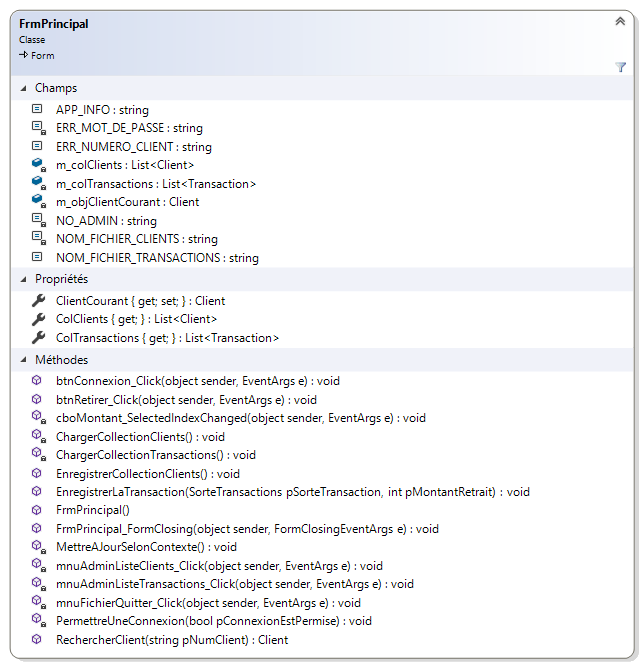
Étape #6 : Le bouton Retirer de l’argent

Ce bouton est disponible si le client peut retirer de l’argent dans son compte. L’administrateur ne peut jamais retirer de l’argent. Lorsqu’il y a un retrait, on enregistre le **retrait** dans le fichier des transactions : sorte de transaction, date et heure actuelle, numéro du client et montant du retrait.Voici un exemple ***2,102,2010-03-09 14:08:51,25***

**Tests pour l’étape #6 : F12**

ÉVALUATION

* Lab1 : étapes 1 et 2, rendre une version NomPrenomTP3Lab1.zip via moodle (pénalité de 25%)
* Lab2 : étapes 3 et 4, rendre une version NomPrenomTP3Lab2.zip via moodle (pénalité de 25%)
* Vous devez rendre avant la date d’échéance la version complète du TP3 nommé : NomPrenomTP3Complet.zip via moodle.
* La qualité de code peut diminuer la note du programme de correction jusqu'à **15%**.
  + Soigner les commentaires pour les classes créées🡪 jusqu’au **10%** pourrait être enlevé.



*Remarque* : Même si ce travail émule le fonctionnement d’un guichet automatique, plusieurs éléments ne sont pas réalistes. Par exemple si une panne de courant survenait les soldes des clients seraient à valider, par contre on pourrait utiliser le fichier des transactions pour rétablir la situation. Normalement les mots de passe ne sont pas lisible par l’administrateur, ils sont habituellent encrytés.