



Travaux Pratiques (Noté)

Un situation courante dans l'industrie

Intervenant : Nicolas DAGNAS

Durée du TP : 10h



Objectif du TP

- Ce TP a pour objectif la création d'une application de gestion d'un catalogue de fournitures de bureau par l'intermédiaire d'une base de données de type SQLite à l'aide de Microsoft Visual Studio .
- Il va vous être demandé trois fonctionnalités :
 - Intégrer un fichier « csv » pour alimenter la base de données.
 - Permettre la gestion de la base de données via des écrans clairs et explicites.
 - Permettre l'export de toutes les données dans un fichier « csv » au même format que le fichier « csv » fournis pour la première demande.
- Une série de contraintes seront explicitées au fur et à mesure de l'avancé de votre projet. Ces contraintes sont là pour vous donner un aperçu de ce que l'on trouve dans le monde professionnel. Le respect de ses contraintes représenteront une part importante de votre note finale, suivez-les rigoureusement.

TP – Les règles de codage

1. Votre code devra être commenté et documenté (Intellisense).
2. Les noms des classes, structures, attributs, méthodes et variables devront suivre le même schéma, **première lettre de chaque mot en majuscule** comme présenté ci-après (Ex: OnLoad, ObjetLeft, etc...). Les variables trop courtes de type « i » ou « Id » sont proscrites.
3. Espacez vos lignes de codes (ni trop, ni trop peu) pour l'éclaircir.

```
/// <summary>
/// Déclenche l'événement Load.
/// </summary>
/// <param name="Args"><b>EventArgs</b> qui contient les données de l'événement.</param>
protected override void OnLoad ( EventArgs Args )
{
    base.OnLoad ( Args );

    // Récupère la valeur de la propriété d'application 'Left'

    object SettingLeft = Bacchus.Properties.Settings.Default["Left"];

    // Si on a une valeur, on l'applique

    if ( SettingLeft != null && SettingLeft is int )
        this.Left = (int)SettingLeft;

    //...
}
```

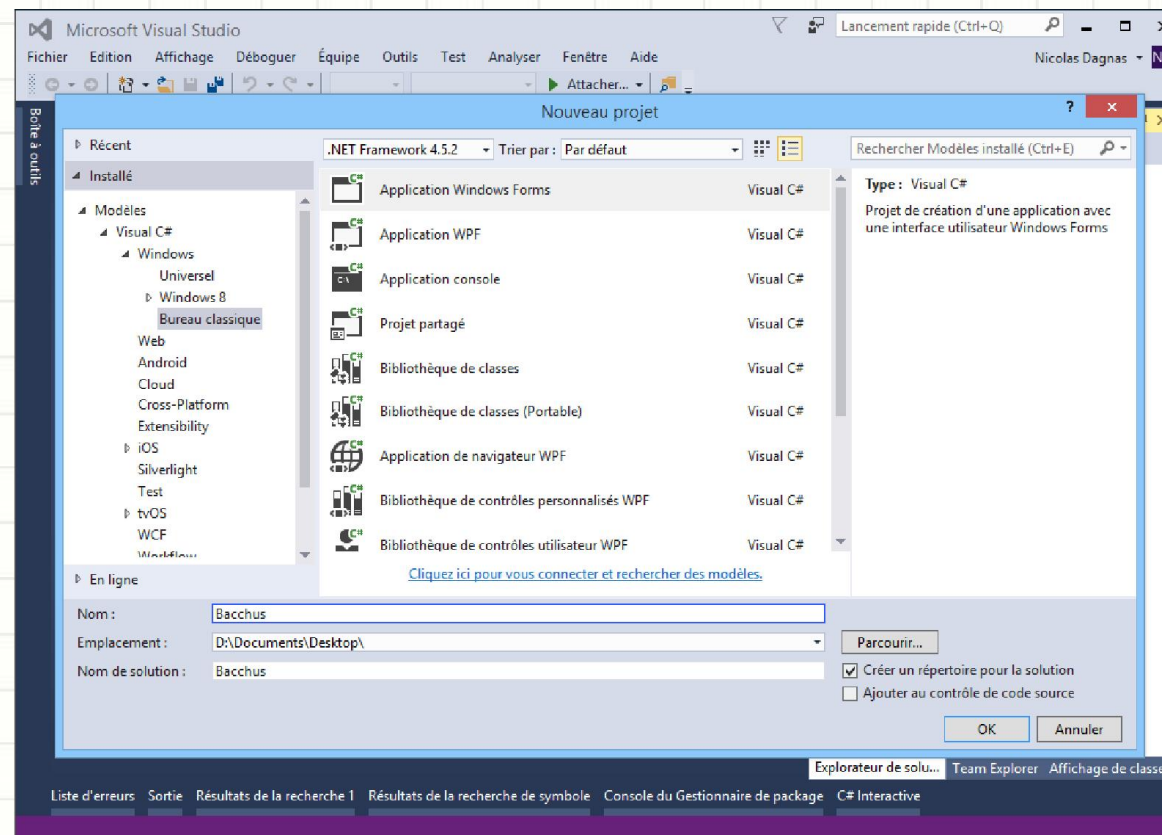



Le rendu du TP

- Le TP sera à rendre au plus tard le lundi 11 juin 2020.
- Le rendu devra se faire à l'aide d'une archive contenant votre projet sans les dossiers « bin », « obj » et « packages » (avec ces dossiers, l'archive serait trop grosse).
- En cas de TP en binôme (voir trinôme si besoin), un seul rendu suffira.
- N'oubliez pas de préciser les noms et prénoms de chaque intervenant dans le nom de l'archive.
- La propreté du code et la simplicité d'utilisation de vos fenêtres représentent la majeure partie de votre note. Donc inutile d'aller trop vite, soignez votre travail car vous pouvez avoir une meilleure note avec un travail incomplet mais propre qu'avec un travail bâclé même si fonctionnel. Inspirez-vous de vos outils habituels.
- Je testerais votre projet avec les fichiers « Csv » et « SQLite » originaux, il est donc de ce fait fortement déconseillé de modifier le fichier SQLite par exemple sous peine de perdre des points pour dysfonctionnement de certaines fonctions.

TP – Nouveau projet

- Créez maintenant un nouveau projet de type « Application Windows Form (.Net Framework) » que vous nommerez « Bacchus ».
- Renommez le nom de la fenêtre principale « Form1 » en « FormMain » via un clic droit sur le nom de l'objet dans l'explorateur de solution.



TP – Fenêtre principale

- Ajoutez un menu « Fichier » à votre fenêtre principale.
 - Ajoutez à votre menu les sous-menus : « Actualiser », « Importer » et « Exporter ».
- Ajoutez un objet « StatusStrip » qui se placera en bas de votre fenêtre principale.
- Ajoutez ensuite un objet « SplitContainer » qui devrait remplir le reste de votre fenêtre principale, si besoin configurez-le pour. Configurez aussi son comportement pour que la partie de gauche ne se réduise pas en dessous de 200 pixels, testez son comportement en exécution.
- Ajoutez un objet « TreeView » dans la partie gauche de votre objet « SplitContainer » qu'il devra remplir.
- Ajouter enfin un objet « ListView » dans la partie droite de votre objet « SplitContainer », même chose pour le remplissage. L'objet devra être en mode de vue « détails ».

TP – Sous-menu « Importer »

- Pour exploiter le fichier « Bacchus.SQLite » fournis, vous commencerez par récupérer le package « System.Data.SQLite » sur Nuget directement via Visual Studio.
 - **Attention : puisque je testerais avec le fichier « SQLite » original, vous ne pouvez pas modifier la structure de la base de données.**
- Pour visualiser votre base de données, utilisez 'SQLite Browser' disponible ici <https://sqlitebrowser.org/dl/> aussi bien pour Pc que pour Mac.
- Ajoutez le fichier « Bacchus.SQLite » fourni à votre projet en le configurant pour qu'il soit copié dans le répertoire de sortie quand il est modifié (Rappelez-vous du fichier txt dans le TD).
- Maintenant que c'est fait, codez votre sous-menu « Importer » pour qu'il ouvre une fenêtre modale (centrée par rapport à la fenêtre principale) contenant :
 - Un bouton permettant de sélectionner un fichier « csv » (le fichier csv fournis ne contient aucune « quantité », cela est volontaire).
 - Un champ contenant le nom du fichier sélectionné.
 - Un bouton permettant de lancer l'intégration en mode écrasement (la Bdd devra donc être vidée avant import).
 - Un bouton permettant de lancer l'intégration en mode ajout (L'import devra mettre à jour les éléments divergents et ajouter les nouveaux).
 - Une barre de progression représentant l'intégration des données.
- Je répète : N'hésitez pas à vous inspirer d'interfaces existantes pour vos fenêtres.
- **Attention, lisez la suite avant de commencer le codage de l'intégration.**



TP – Sous-menu « Importer »



- Le résultat de l'intégration, nombre d'articles ajoutés, anomalies, devra être affiché via un MessageBox.
- Comme on dit toujours, qui peut le plus, peut le moins. Donc pour l'utilisation des données, vous utiliserez exclusivement les objets « SQLiteConnection », « SQLiteCommand » et « SQLiteDataReader », « SQLiteParameter » est aussi autorisé. **Vous n'utiliserez aucun schéma simplifiant l'utilisation des données, en gros vous ferez vos propres requêtes à la main.**
- La base de données vous étant fournis, à vous de l'analyser pour voir comment intégrer les données. Les champs IDs sont en auto-incrément, pensez à correctement les gérer.
- Une astuce : ne faites pas une intégration « à l'arrache », soyez plus malin ;) Par exemple, la création et l'utilisation de classes ou de structures reflétant les tables de la base de données pourrait être une bonne piste.
- NB : les indexes des tables « Familles », « SousFamilles » et « Marques » sont en auto-incréments, il vous faudra les gérer correctement.



TP – Sous-menu « Exporter »

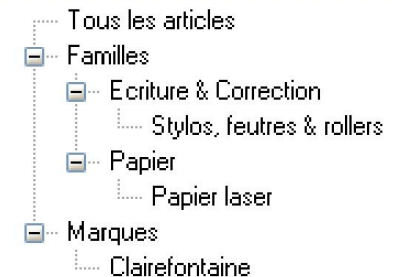


- Vous allez maintenant coder votre sous-menu « Exporter ». C'est bien entendu la même chose que l'intégration mais à l'envers, enfin presque puisque l'on n'a pas besoin d'options car cette fonction exportera toutes nos données.
- Cela devra se faire tout naturellement via une fenêtre modale (toujours centrée par rapport à la fenêtre principale).
- Attention, le format des données du fichier « csv » généré devra être rigoureusement le même que le format du fichier fournis, c'est-à-dire qu'on doit être capable d'intégrer ce fichier via le sous-menu « Intégrer » sans anomalie.
- Vous pouvez ajouter le champ « Quantité » en fin de ligne si vous le souhaitez, mais il ne sera pas géré dans l'import.

TP – Fenêtre principale (suite)

- Dans votre TreeView (Partie gauche) vous ajouterez plusieurs éléments qui permettront de charger votre ListView (Partie droite) selon certains critères.
- Nœuds racines (Nœuds de premier niveau) :
 - Nœud « Tous les articles » : Chargera la liste complète des articles dans le ListView.
 - Nœud « Familles » : Chargera la liste des familles dans le ListView.
 - Nœud « Marques » : Chargera la liste des marques dans le ListView.
- Le nœud « Familles » devra disposer de la liste des complète familles » en tant que nœuds enfants, et chaque nœud enfant « famille » devra disposer des « sous-familles ».
 - Un clic sur un nœud « famille » chargera la liste des « sous-familles » dans le ListView.
 - Un clic sur un nœud « sous-famille » chargera la liste des « articles » appartenant à cette sous famille dans le ListView.
- Le nœud « Marques » devra avoir en nœuds enfants la liste complète des marques.
 - Un clic sur un nœud « marque » chargera la liste des « articles » appartenant à cette marque dans le ListView.
- Dans chacun des cas, vous devrez gérer le trie des données à l'écran (exclusivement dans l'objet « ListView », le but n'est pas de refaire une requête) via un clic sur les entêtes de colonne. Cette option devra permettre de trier de manière ascendante et descendante.

EXEMPLE





TP – Les listes

- Les entêtes de colonnes de l'objet « ListView » devront être les suivantes en fonction de ce qui est chargé :
 - Si on a chargé une liste « d'Articles » :
 - « Description »
 - « Familles »
 - « Sous-familles »
 - « Marques »
 - « Quantité »
 - Si on a chargé une liste de « Familles », de « Sous-familles » ou de « Marques » :
 - « Description »

TP – Les groupes

- En fonction de ce qui sera chargé dans l'objet « ListView », vous devrez gérer les groupes d'éléments :
- <https://docs.microsoft.com/fr-fr/dotnet/framework/winforms/controls/how-to-group-items-in-a-windows-forms-listview-control>
- Quand les articles sont chargés, vous devrez grouper les articles selon le tri en cours qui correspond au tri visible sur les entêtes de colonnes (Visualisez cela comme un explorateur de fichier).
 - Si le tri est sur la colonne « description », groupez par première lettre.
 - Si le tri est sur la colonne « familles », groupez par famille.
 - Si le tri est sur la colonne « sous-familles », groupez par sous-famille.
 - Si le tri est sur la colonne « marque », groupez par marques.
- Quand se sont des « familles », « sous-familles » ou des « marques » qui sont chargés, ne groupez que par première lettre puisque l'on ne doit avoir que des descriptions de chargés dans le ListView (Slide précédent).



TP – Interactions

- Les interactions suivantes avec l'objet « ListView » sont à coder :
 - La touche « Entrée » ou un double clic sur un élément ouvrira la fenêtre de modification de l'élément associé.
 - La touche F5 rechargera la liste des éléments tout comme le sous-menu « Actualiser ».
 - La touche « Supp » demandera la suppression de l'élément sélectionné.
 - Le clic droit devra ouvrir un menu contextuel permettant d'ajouter un élément, de modifier ou de supprimer l'élément sélectionné.

TP – Fenêtres Ajout/Modif.

- Les fenêtres d'ajout et de modification d'un élément devront toutes être modales et centrées par rapport à la fenêtre principale.
- Dans le cas de la fenêtre d'ajout/modification d'un article et uniquement dans celle-là, vous utiliserez des objets « ComboBox » pour sélectionner les « marques », « familles » et/ou « sous-famille » en mode « DropDownList ».
- Vous contrôlerez aussi l'existence, le format, ou la taille de ce qui est sélectionné/saisie pour éviter toute erreur (on ne peut valider la fenêtre si un « ComboBox » est vide par exemple ou si une référence est trop grande ou une quantité invalide).
- Toutes vos fenêtres devront respecter les règles basiques du design. Les champs doivent être alignés et la tabulation doit suivre un chemin logique, leurs apparences respectives doivent être confortables à l'œil et simples à utiliser.
- En gros, vous devez assister l'utilisateur final pour qu'il n'ait pas à trop réfléchir lors de l'utilisation de vos fenêtres, envisagez l'utilisateur comme quelqu'un n'y connaissant strictement rien en informatique et découvrant l'outil.
- L'ajout, la modification ou la suppression d'une « famille », d'une « sous-famille » ou d'une « marque » devra être répercutée dans l'objet « TreeView ».



TP – Compléments

- Assurez-vous de n'avoir aucune anomalie ni avertissement à la compilation de votre projet.
- Enregistrez la position de votre fenêtre principale quand vous la fermez. Pour cela, utilisez « Settings » comme vous pouvez le voir dans le *slide 3*, le statut « Maximisé » ne doit pas être oublié.
- Puisque maintenant nous avons cette information, replacez votre fenêtre principale lors de son lancement à son dernier emplacement.
- Nous l'avons oublié, mais nous avons un objet « StatusStrip » sur notre fenêtre principale, et bien placez-y le nombre d'articles, de familles, de sous-familles et de marques actuellement dans notre base de données et mettez-le à jour quand cela vous semble logique de le faire ;)