

# Linq To Entities / MVVM : Exercice

## Objectifs

1. Développer une petite application professionnelle intégrant les notions LINQ To Entities, WPF, MVVM vues

## Aperçu de l'application à développer

MainWindow

Employees

First Name	Birth Date
Andrew Fuller	19/02/1952
Janet Leverling	30/08/1963
Margaret Peacock	19/09/1937
Steven Buchanan	4/03/1955
Michael Suyama	2/07/1963
Rob King	29/05/1960
Laura Callahan	9/01/1958
Anne Dodsworth	27/01/1966

MVVM

Last Name : King

First Name : Rob

Title : Mr.

Birth Date : 29/05/1960

Hire Date : 2/01/1994

OrderID	OrderDate	OrderTotal
11074	6/05/1998	17,4500
11066	1/05/1998	40,6500
11055	28/04/1998	91,0000

Add Remove

Save Cancel

## Considération sur l'architecture

Pour réaliser cette application, nous allons bien entendu utiliser le pattern MVVM. Nous aurons donc dans la solution un répertoire « Views », « Models » et « ViewModels ». Le modèle sera généré via Linq to Entities et placé dans le répertoire « Models ». La vue a déjà été réalisée par un designer qui a également déjà défini les noms pour le binding (jetez un œil au fichier xaml).

Votre travail consistera donc surtout à créer le ViewModel. Pour ce faire, il faudra créer un modèle client « *EmployeeModel* » en utilisant la technique du wrapping. Ce modèle définira les propriétés d'un employé accessible par la vue (voir Binding de la vue).

Il vous faudra également une classe *EmployeeVM* qui contiendra la liste des *EmployeeModel* et s'occupera de peupler cette liste et de la rendre accessible à la vue.

**En résumé l'idée est donc la suivante :**

Le datagrid contient une liste d'employés qu'il ira rechercher dans le ViewModel (*EmployeeVM*). Cette liste contiendra des *EmployeeModel*.

→ `ItemsSource="{Binding EmployeesList}"`

Pour afficher un élément de cette liste c'est-à-dire un employé(*EmployeeModel*), la vue ira rechercher une propriété dans le modèle client (*EmployeeModel*)

→ `<DataGridTextColumn Binding="{Binding FullName}"`

## Exercice

---

1. Vérifiez que la base de données «Northwind» est bien installée sur votre machine.
  1. Cfr. Cours sur Linq to Entities
2. Récupérez sur Claco le projet de base (WpfApplicationEmployeeWithoutCommand)
3. Première étape → lien avec la base de données
  1. Générez le modèle à partir de la base de données Northwind
4. Deuxième étape → Remarquez que le binding dans la vue a déjà été fait par le designer
  1. Vous devrez respecter ceci
5. Troisième étape → affichage des « Employees » dans le datagrid
  1. Pensez à respecter MVVM
  2. Créez une méthode LoadEmployees()
  3. Pensez à créer une classe EmployeeModel qui encapsulera un Employee(généré par LINQ to Entities)
  4. Votre liste d' « Employee » contiendra des « EmployeeModel »
  5. Le datagrid affiche le fullName (concaténation du nom et prénom) et le birthdate
    1. Que faire pour que cela s'affiche ? Où placer les propriétés de binding ? Que contient votre « EmployeeList » ?
  6. Pensez également au DataContext
    1. Comment la vue est-elle liée au viewModel ?
6. Quatrième étape → affichage dans le formulaire lors d'une sélection dans le datagrid
  1. ne vous préoccupez pas trop de la combo-box pour l'instant
7. Cinquième étape → affichage des différents titres de courtoisie dans la combo-box
  1. Rappelez-vous que le datacontext est hérité → la combo-box a donc un datacontext par défaut lié au grid « form ». Celui-ci contient un « employeeModel ». Ceci ne nous arrange pas car nous voulons tous les titres de courtoisies possibles et non pas celui d'un employé en particulier. C'est pourquoi le designer a précisé dans le XAML d'utiliser le datacontext de la fenêtre et nous utiliserons une propriété ListTitle dans ce datacontext.

2. Vous en savez maintenant normalement assez → faites en sorte d'afficher les différents titres de courtoisie possibles dans la combo-box.
3. Inspirez-vous également de LoadEmployees()