[pacouy@unive-tours.fr](mailto:pacouy@unive-tours.fr)

[jf.pacouy@gmail.com](mailto:jf.pacouy@gmail.com)

# Séance 1

Projet : Master.Mission

|  |
| --- |
| Executer: windows +r |
| VS:   * F5 = executer * +Ctrl+B : générer   Clique droite dossier: Propriete   * Application: Startup object=> Master…. * Build: cocherXML…… |
| Creer nouvelle solution:   * clique droite “Solution Master.Mission”=> add=> new project => application Window : « Master.Mission.Win » * clique droite “Solution Master.Mission”=> add=> new project => bibliotheque : « Donnees » * clique droite “Solution Master.Mission”=> add=> new project => bibliotheque : « Metier » * clique droite “Solution Master.Mission”=> add=> new project => bibliotheque : « Outils » |
| Boite outils = Toolbox   * Sur Form.cs: choisir les bouton ; menuStrip |
| Créer new folder solution   * clique droite “Solution Master.Mission”=> add=> new solutionFolder |
| Ajouter un bibliotheque externe  Exemple : clique droite metier => add=> reference=>project => Donnees   * IHM reference Outils et Metier * Metier reference Outils et Données * Donnees reference Outils * Outils reference rien |
| Clique droite « Master.Mission » =>add=>class=> class : « Identification », changer public class ; ajuter des boucles, des variables, des methodes…. |
| Snippets : créer un constructeur => taper ct et 2 fois de tap |
| C# : langage 100% objet |

|  |  |
| --- | --- |
| Projet de test | New project => Test Unit  Pour tester : choisir Test => run all test |
| Dans Metier , | Creer un fichier Modele avec la classe Salarie |
| Diagramme | Clique droite sur une classe => choisir class Diagramme : prop + TabTab  Entrer les données dans Properties de diagramme |
| Sealed(string) | On ne peut pas créer un |
| internal | Classe ne sont pas accessible à partir une autre BDD |
| Analyze | * Calcul codes metric for solution |

# Séance 2:

Choisir tous les icons, taper dans click : GestionCentralisee

DRY : ne pas se repeter

KEAP le plus simple

YAGNI en ai-je besoin ?

# Séance 3:

|  |  |
| --- | --- |
| Extension   * Indent guides * Ghost doc * Power tools * Vs commands * Database * Log4net | Outils => Extension et Update |
| Ajouter paquet **entity Framework** dans Données | Choisir Données =>Gerer les packages nuget => Entity Framework |
| * Form 1 => Form Accueil * ShortcutKeys * Changer dans les proppriété, * Créer nouveau Win Form * Action de Salarie/Creér(Dans forme Accueil) : les instancier ForlSalaire, Show() | * Choisir les butons correspondes * Dans Click = « GestionCentralisee » * Add=> win form=> changer Text : gestion des salarie * Ajouter le code dans file formAccueil.cs  |  | | --- | | if (sender is ToolStripMenuItem)  {  var menu = ((ToolStripMenuItem)sender).Name;  switch(menu)  {  case "quitterCtrlQToolStripMenuItem": Close(); break;  case "créerCtrlNToolStripMenuItem":  new FormeSalarie().Show();  break;  default: break;  }  } | |

|  |  |
| --- | --- |
| Ajouter une source de donnée dans Win Form | Project => Add new data Source => Objet=> Métier=> Salarie |
| Affichage et faire glisser | Afficher=>autre fenetre=>Data Source=> Faire glisser dans Gestion des salarie |
| Créer un nouveau dossier (Service) => créer un nouvelle classe ( Service.Mission) | Ecrire le code dans ServiceMission   |  | | --- | | using Metier.Modele;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Metier.Service  {  public class ServiceMission  {  private List<Salarie> listSalarie;  public Salarie GetSalarieByID(int id)  {  foreach (var item in listSalarie)  {  if (item.Identifiant == id)  {  return item;  }  }  return null;  }  public IList<Salarie> GetAllSalarie()  {  return this.listSalarie;  }  public ServiceMission()  {  //ajouter 3 salaries à la listes  var salarie1 = new Salarie(){  Identifiant = 1  ,Nom = "Jean"  ,Prenom = "Titi"  };  var salarie2 = new Salarie(){  Identifiant = 2  ,Nom = "Pierre"  ,Prenom = "Toto"  };  var salarie3 = new Salarie(){  Identifiant = 3  ,Nom = "Marie"  ,Prenom = "Tata"  };  this.listSalarie = new List<Salarie> { salarie1, salarie2, salarie3 };  // ou add  //this.listSalarie.Add(salarie1);  //this.listSalarie.Add(salarie2);  //this.listSalarie.Add(salarie3);  }  }  } | |
| Loading les info dans le tableau de la classe : FormSalarie.cs Designe | Clique double sur FormSalarie.cs(Design) et modifier le code |

Diff Merge : pour synchro de code source

# Séance 4:

Merise : JMerise, Dia, Viro

\*\*\*Procédure 4.1. Le composant de connexion.

Société INFRAGISTICS

|  |  |
| --- | --- |
| Créer un contrôle utilisateur dans Master.Mission.Win | Add => User Control |
| Ajouter les codes | public string Identifiant {  get{  return this.textBoxIdentifiant.Text;  }  }  public string MotDePasse{  get{  return this.textBoxMotDePasse.Text;  }  } |
| Ajouter un onglet | * Cliquer à droite du côté ToolBox,=> new tab( onglet => Nommé Master CCI * Générer solution * Puis cliquer à droite => choisir les items * Parcourir pour chercher Master.mission.Win => bin=> debug =>….exe=> choisir User… |
| Créer nouvelle formulaire | Add=> Windows Form=> Nommer « FormConnexion » => Glisser dans outils, le pointer de Master 2 CCI -> FormConnexion |
| Créer nouveau Form Window => « Form Option » => changer dans programme.cs : Application.Run(new FormOption()); | APPARENCE : TextBox=>Propriete :  [AutoCompleteCustomSource] => Reset=> tapper les chaines  Clique droite [AutoCompleteMode] => SuggesAppend -> pour proposer les suggestions  Clique droite [AutoCompleteSource] => Custom source  LANGUE : ComboBox=>Propriete :  [Items]  [DataSource]=> nom de variable langue  [ValueMember]=> pays de langue  [DisplayMember]=> Langue  Changer le code de FormOptions   |  | | --- | | private void FormOption\_Load(object sender, EventArgs e)  {  var langues = new List<Langage>()  {  new Langage(){Pays="FR",Langue="Français"},  new Langage(){Pays="AN",Langue="Anglais"},  new Langage(){Pays="ES",Langue="Espagne"},  new Langage(){Pays="IT",Langue="Italie"},  new Langage(){Pays="VI",Langue="Vietnam"},  new Langage(){Pays="DE",Langue="Allemand"},  };  //.....alimenter la listes evec FR/MA/ES/VI....  comboBoxLangue.DataSource = langues;  comboBoxLangue.DisplayMember = "Langue";  comboBoxLangue.ValueMember = "Pays";  }  internal class Langage  {  public string Pays { get; set; }  public string Langue { get; set; }  } | |
|  |  |
|  | : ComboBox=>Propriete => Events :  [SelectedValueChanged] => ajouter code   |  | | --- | | private void comboBoxLangue\_SelectedValueChanged (object sender, EventArgs e)  {  MessageBox.Show("Choix" + ((Langage)comboBoxLangue.SelectedItem).Pays);  } |   LIST VIEW : name =listViewLangue  Ajouter le code  Hh :mm :ss  Label name= labelHeure  TIMER  [name] : timerHeure  …True  …true  …1000  **EVENT**: tapper=> timerHeure\_tick  Ajouter code   |  | | --- | | private void timerHeure\_tick(object sender, EventArgs e)  {  labelHeure.Text = DateTime.Now.ToString("hh:mm:ss");  } |   BACKGROUNDWORKER :  [DoWork]  [ProgressChanged]  [RunWorkerCompleted]: fin de travail |

# SÉANCE 5

Console.WriteLine() : cw+tabtab

|  |  |  |
| --- | --- | --- |
|  | Ajouter reference service | |
| New projet-> application win forme “WinComposants” |  | |
| Outils-> ajouter classe ->”programme.cs” | Transformer class en classe main en ajoutant  static void Main()  {  Console.WriteLine("Simplification c#");  Console.ReadLine();  } | |
| METHODE DE DEMARRAGE=>  clique droite de Outils -> Propriété |  | |
| Ajouter classe Vehicule dans Outils -> Programme.cs  Propriete est public  Attribut est privé | //test proprietes avec initialiseurs  var nguyencar = new Vehicule(){Kilometrage = 30.89, Plaque =" A06", Puissance = 4};  Console.WriteLine(nguyencar.Kilometrage);  Console.WriteLine(nguyencar.Plaque);  Console.WriteLine(nguyencar.Puissance); | |
| Objet anonyme | Nouvelle collection | |
| expression lambda  Le lambda-calcul est un formalisme de représentation des fonctions. | Relier les variables. Par exemple x=>f(x)  Exemple | |
| Les extensions | Permet d'étendre un type sans recompiler le type  public static class Extensions  {  public static string Gauche(this string s, int i)  {  Return “ ”;  }  }  Console.WriteLine("MasterCCI".Gauche(1)); | public static class Extensions  {  public static string Gauche(this string s, int i)  {  if (i <= s.Length)  return s.Substring(0, i);  else return s;  }  }  Console.WriteLine("MasterCCI".Gauche(1)); |
| Paramètre nommé | Dans classe vehicule  public string CarteGrise (string plaque, int puissance = 5, double Km = 0 )  {  return "";  }  nguyencar.CarteGrise("test");  nguyencar.CarteGrise("test",4);  nguyencar.CarteGrise(Km:75000, plaque:"test", puissance:15); | |

**Generatedata. Com => tester**

|  |  |
| --- | --- |
| LINQ :  Ecriture est inverse, select est à la fin | //LINQ /Collection  var ti = new int[] { 1, 2, 3, 4, 5, 6 };  //recherche des entiers paires  var respair = from vi in ti  where vi % 2 == 0  orderby vi ascending  select vi;  // respair en méthode d'extension  var respairme = ti.Where(t => t % 2 == 0)  .OrderByDescending(t => t);  foreach( var item in respair )  Console.WriteLine(item);  foreach (var item in respair.ToList())  Console.Write(item);  Console.WriteLine("Sum= "+ respair.Sum());  Console.WriteLine("Max= " + respair.Max());  //chiffres pairs, utilise expression lamda <=4  var respair2 = respair  .OrderBy(r=>r)  .TakeWhile(r => r <= 4);  Console.WriteLine(respair2.Sum()); |

|  |  |
| --- | --- |
| Outils -> new element -> recherche conf -> File configuratn application | On peut vérifier dans Outils -> bin -> debug |
| Outils -> ckique droite -> propertis-> | Console.WriteLine(new Settings().Version);  Console.WriteLine(ConfigurationManager.AppSettings["Auteur"]);  Manipulation |

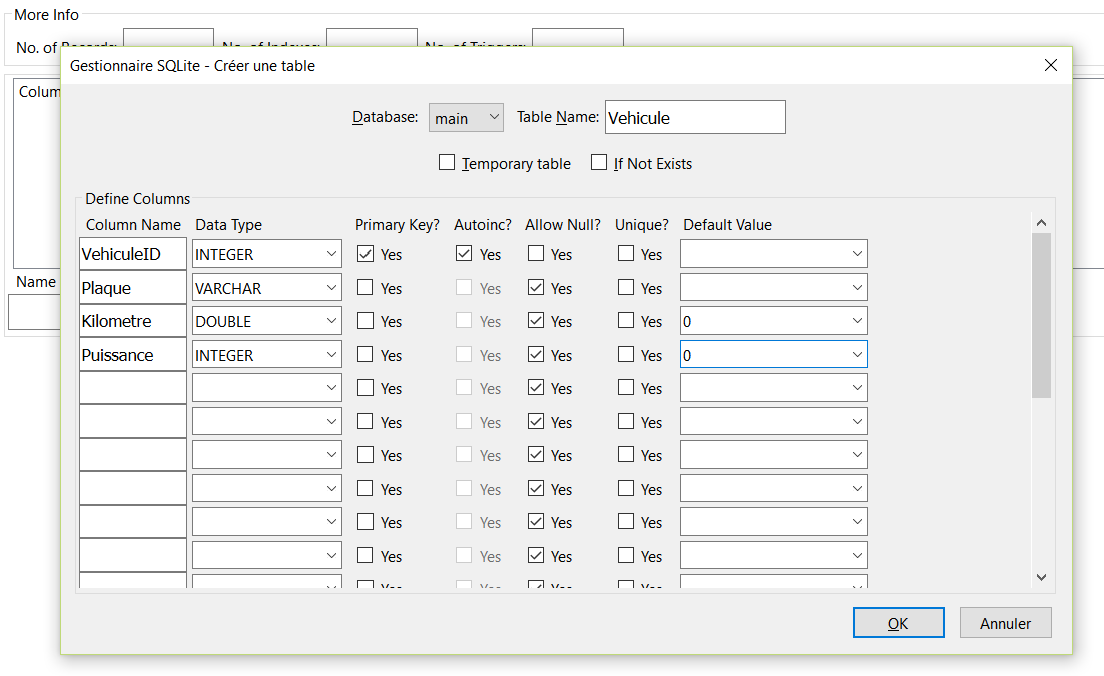
# SEANCE 6

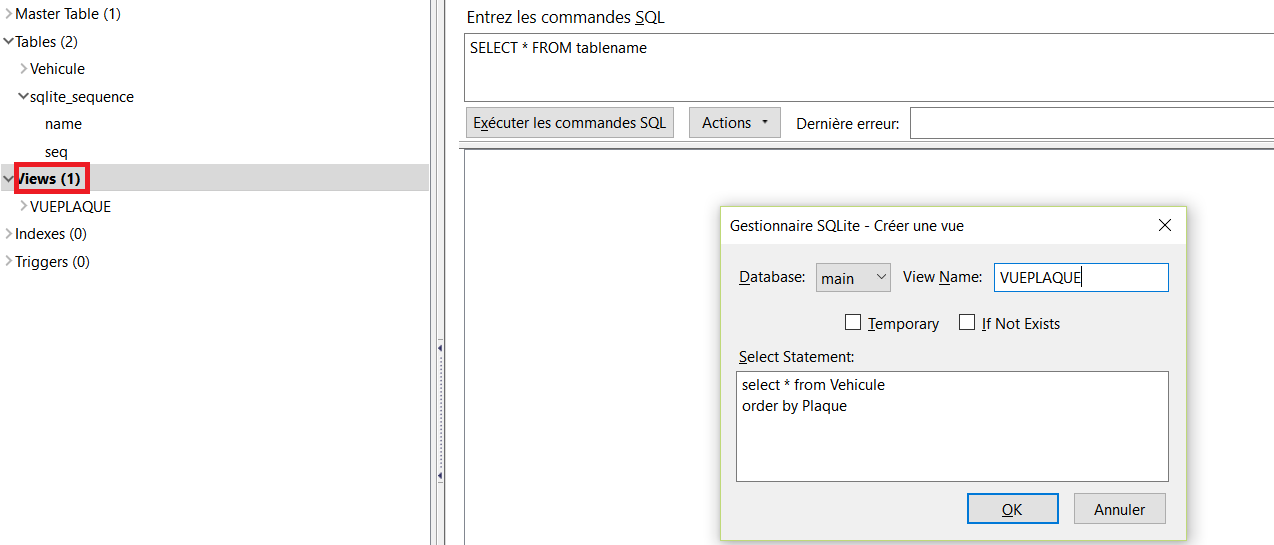
Mysql provider ado.net

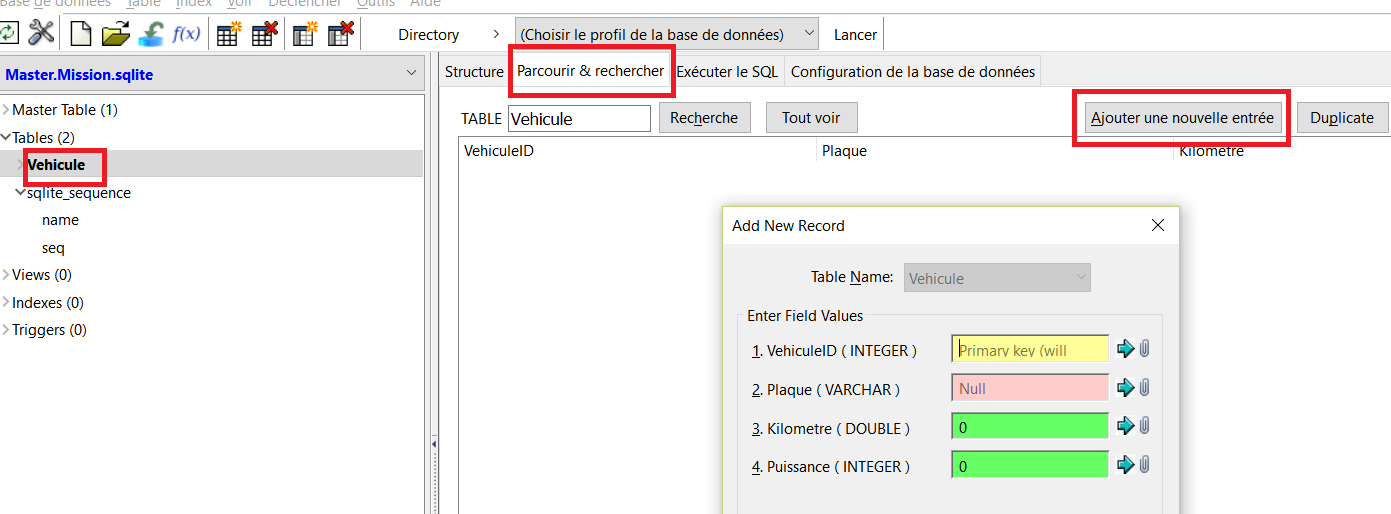
Connectionstrings.com

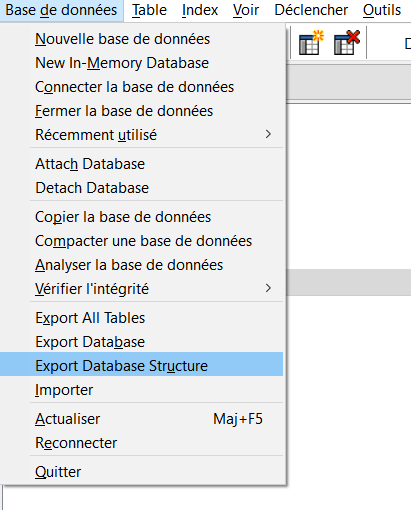
Manipuler avec SQLite :

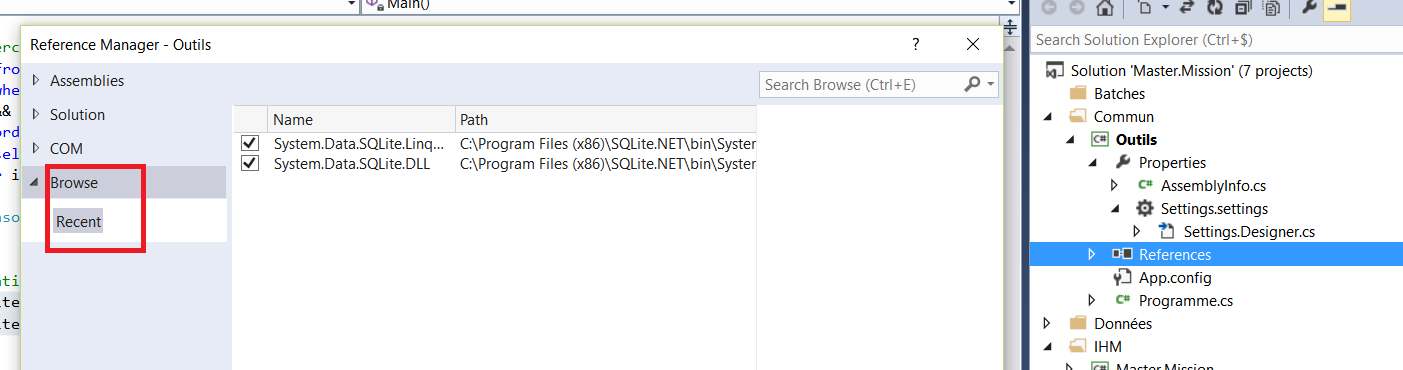
* créer une base SQLite
* FireFox -> Module -> Extension -> SQLite manager -> Installer
* Outils -> SQLite Manager
* Créer nouvelle BDD -> « Master.Mission » et enregistre dans le bureau
* Table -> Créer une table

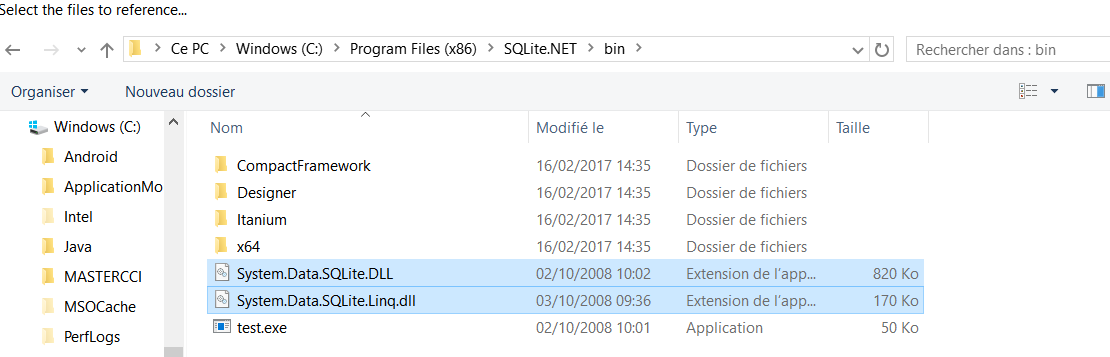








Enregistrer dans la page



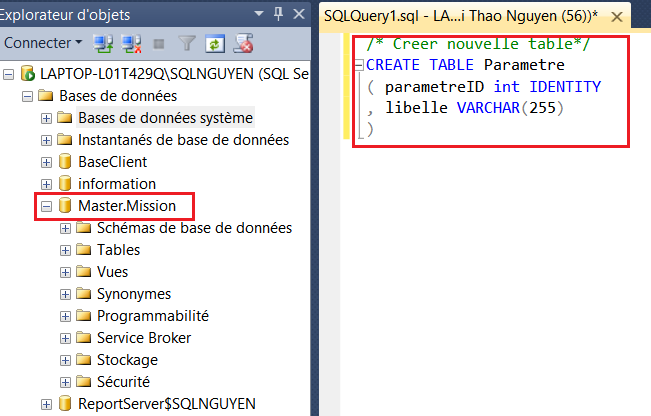
|  |  |
| --- | --- |
| Connexion    Faire glisser la base donnee dans Outils  Autre methode  !!!!!Choisir ajouter comme Lien | // 1.Connexion  SQLiteConnection connection = new SQLiteConnection("Data Source=Master.Mission.Sqlite; Version=3;");  connection.Open();    // 2. Commande Select  SQLiteCommand cmd = new SQLiteCommand("Select \* from vehicule", connection);  // 3. Lecture  SQLiteDataReader dataReader = cmd.ExecuteReader();  dataReader.Close();  connection.Close();  **Resoudre le probleme**    Puis ajouter dans App.config  <startup useLegacyV2RuntimeActivationPolicy="true">  <supportedRuntime version="v4.0"/>  </startup>  **Clique droite de fichier Master.Mission .Sqlite** |
| Classe Fabrique Vehicule   * List<Vehicule>lv * Construire nouvelle Liste * Ajouter(String I, String p, String Personne) * List<Vehicules() | **New class FabriqueVehicule.cs**  **Dans programme.cs** |

# SÉANCE 7

Ado.net -> data table -> data rows and data columns

LinQ to SQL Server: universel => que pour Microsoft

O/RM : object relationnal mapping

BDD : Master.Mission

**Visual Basic:**

1. Classe
2. Context (ED Modele)
3. Chaine de connexion
4. Nuget/Console
   1. Saisir le projet Donnees
   2. enable-migrations
   3. add- migration -> Name :initial
   4. update-database -verbose -> SQL Server

## Classe

## Context (ED Modele)

### 1. Creation

Console.WriteLine("Creation");

var p1 = new Parametre()

{

parametreID = 1, libelle = "Auteur: TRAN"

};

var p2 = new Parametre()

{

parametreID = 2, libelle = "Version: 1.0.0"

};

### 2. Instancier le context

Console.WriteLine("Instancier le context");

Master\_MissionDataContext context= new Master\_MissionDataContext();

context.Parametre.InsertOnSubmit(p1);

context.Parametre.InsertOnSubmit(p2);

context.SubmitChanges();

### 3. Select

Console.WriteLine("Selection");

foreach (var item in context.Parametre)

{

Console.WriteLine(item.libelle);

}

Console.WriteLine("============");

var req = from p in context.Parametre

where p.libelle.Contains("Auteur")

select p;

foreach (var item in context.Parametre.Where(p=>p.libelle.Contains("Auteur")))

{

Console.WriteLine(item.libelle);

}

Console.WriteLine("============");

### 4.Update

Console.WriteLine("Update");

foreach (var item in context.Parametre)

item.libelle = item.libelle.ToUpper();

context.SubmitChanges();

Console.WriteLine("============");

### 5.Delete

Console.WriteLine("Delete");

foreach (var item in context.Parametre)

{

context.Parametre.DeleteOnSubmit(item);

context.SubmitChanges();

}

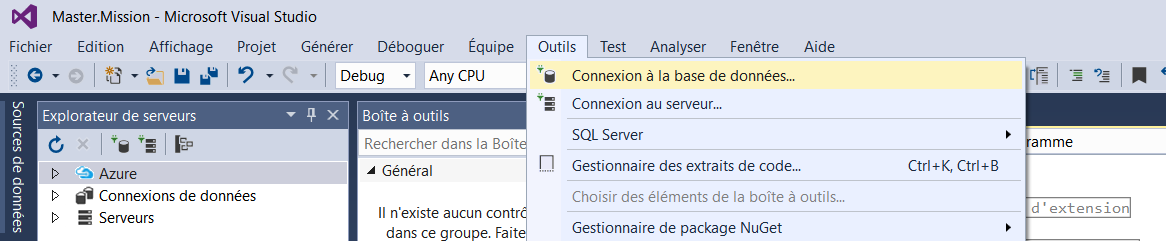
Console.WriteLine("============");

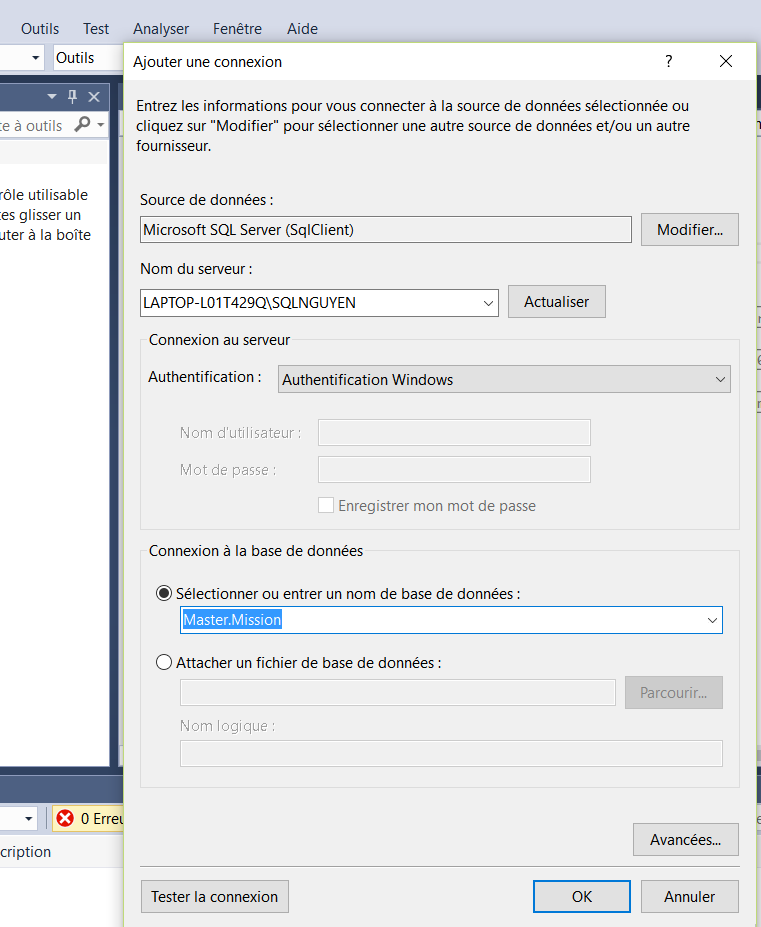
## Chaine de connexion

### Etape 1

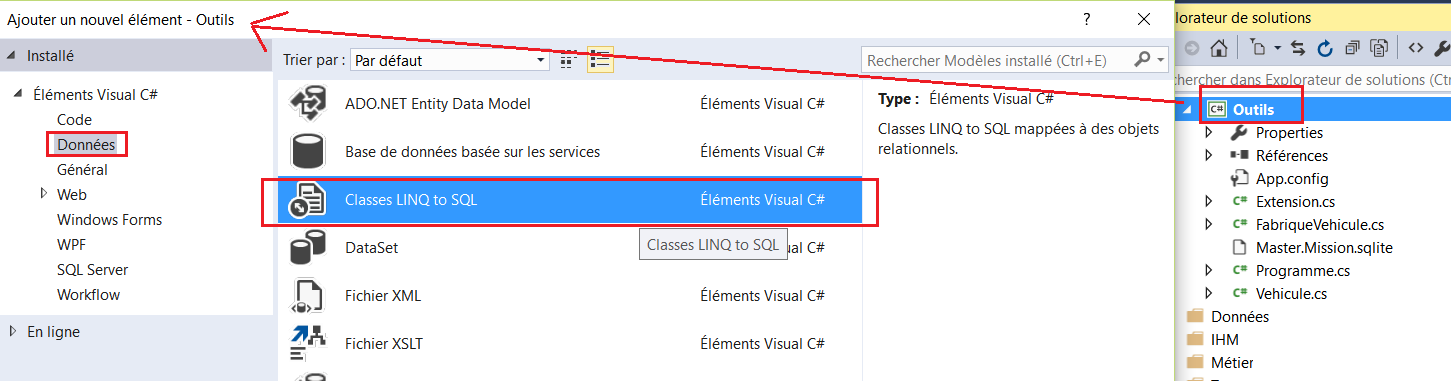
Outils -> Connecter à la BDD

Si ça marche pas, on va au Service Windows. Pour verifier si Server SQL marche ou pas.



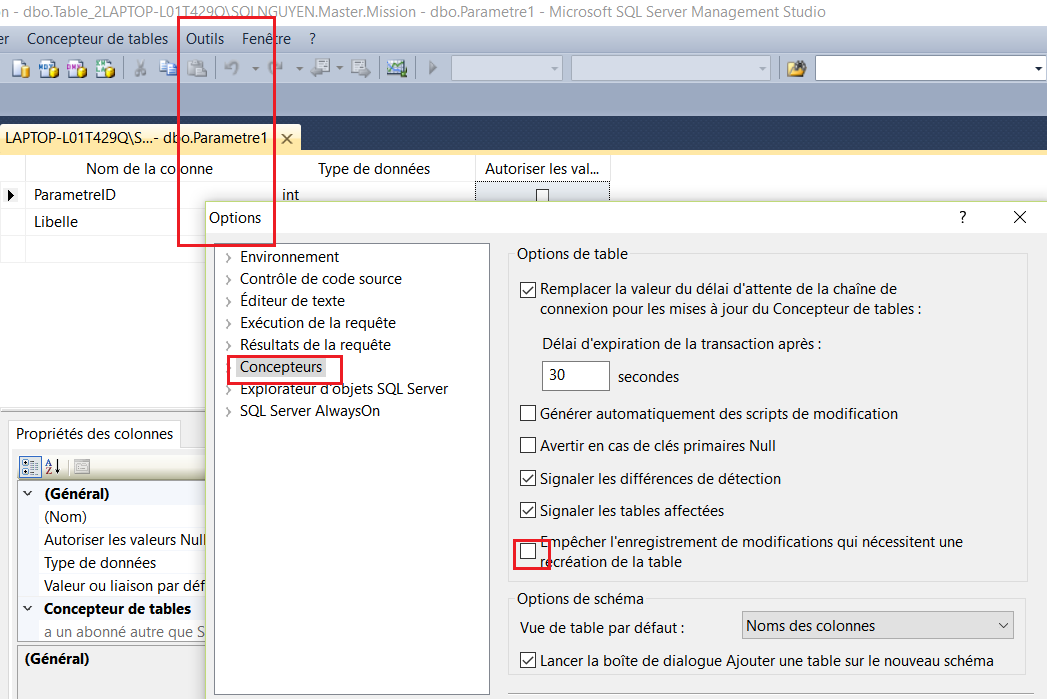


### Etape 2

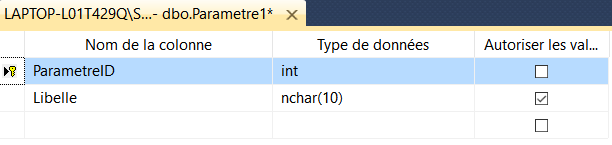
****

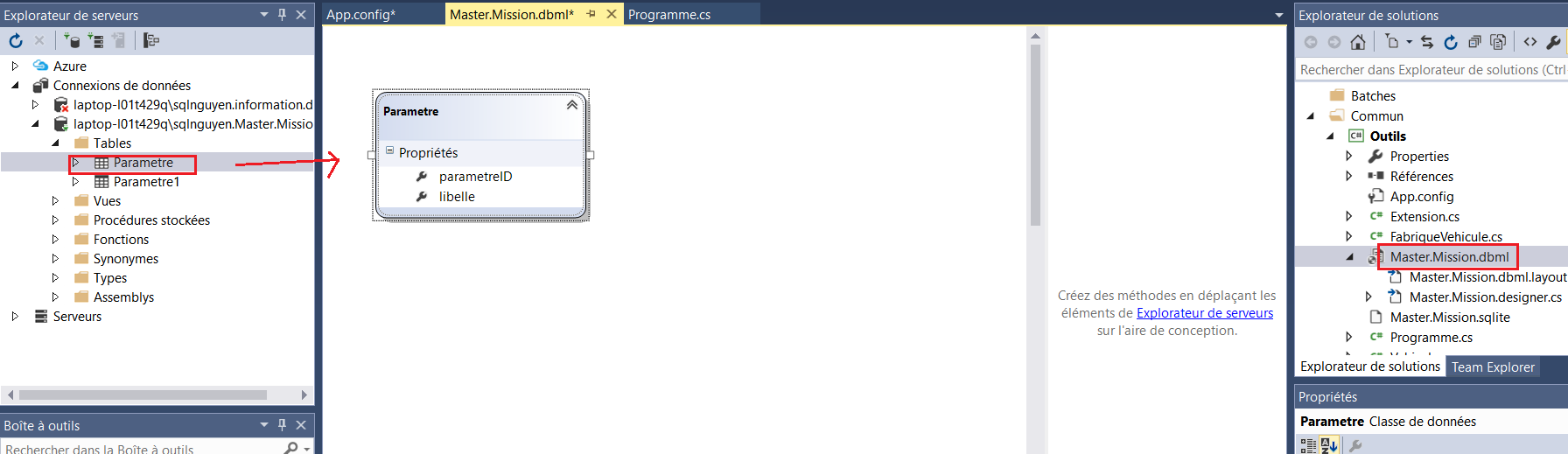
**SQL Server:**

Outils -> Options



**Choisir ParametreID comme clé primaire:**



Dans VISUAL BASIC  


* Dans app.Config

Auteur: mon nom

Version: 1.0.0

Entity Framework

Database first: SQL -> C#

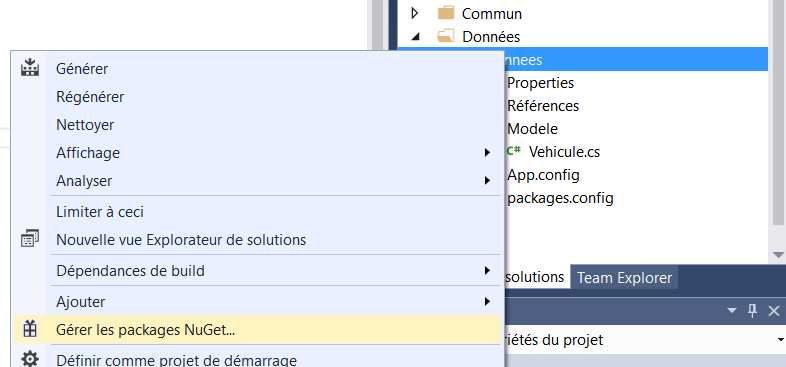
Modele First: Designe -> C# + SQL (Mérise)

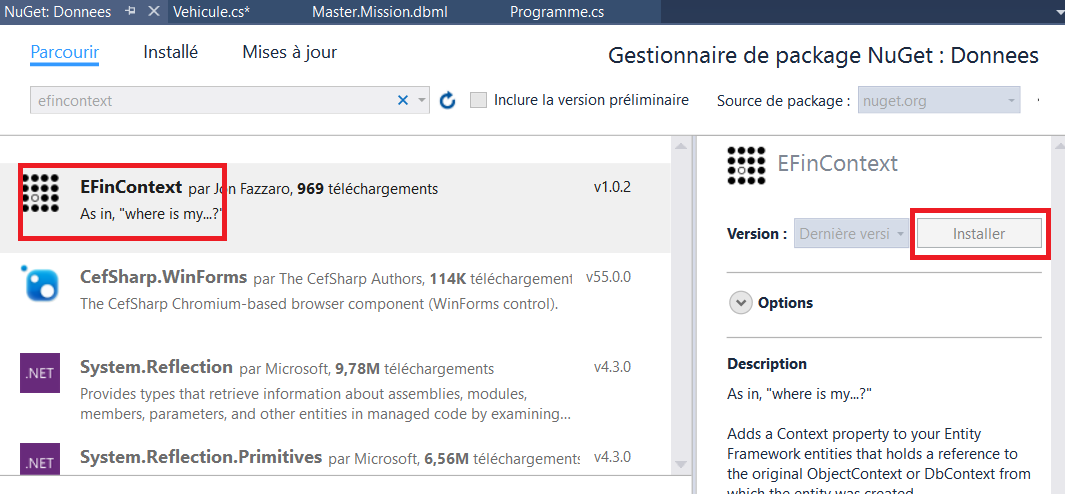
Code First: C# -> SQL

Context: code commande

**PRATIQUE**

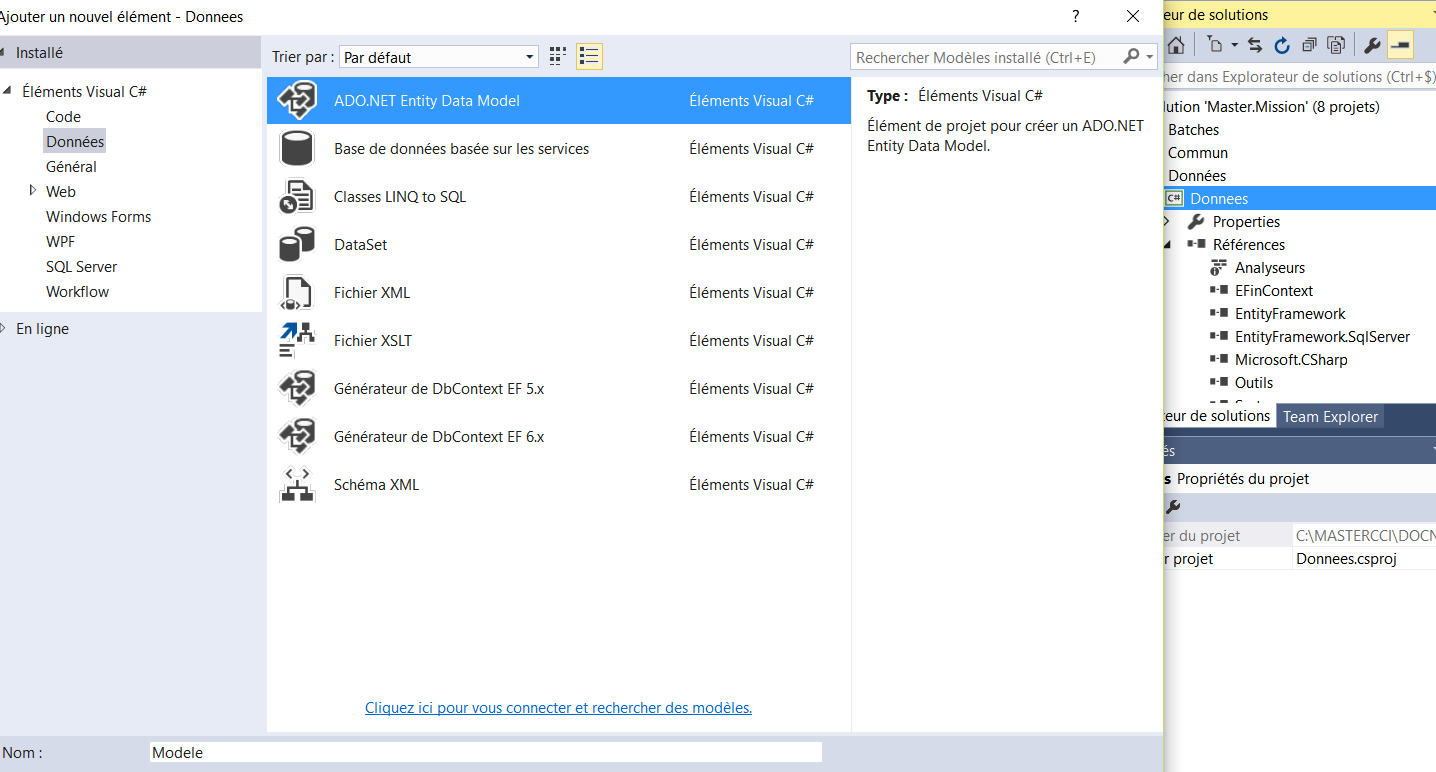
Données -> Modèle->new classe “Vehicule”

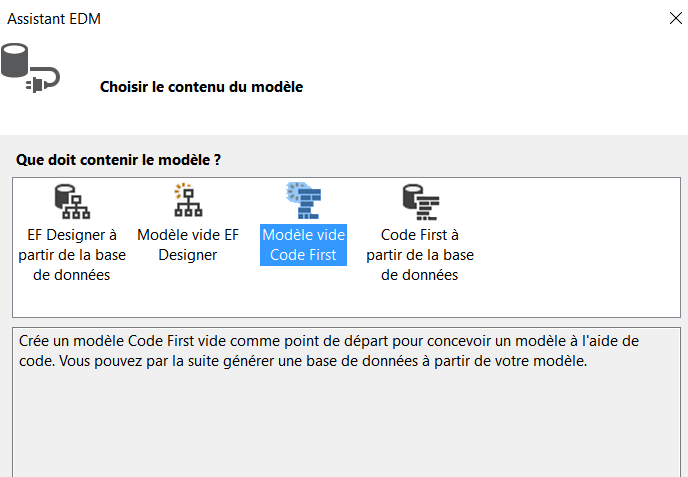


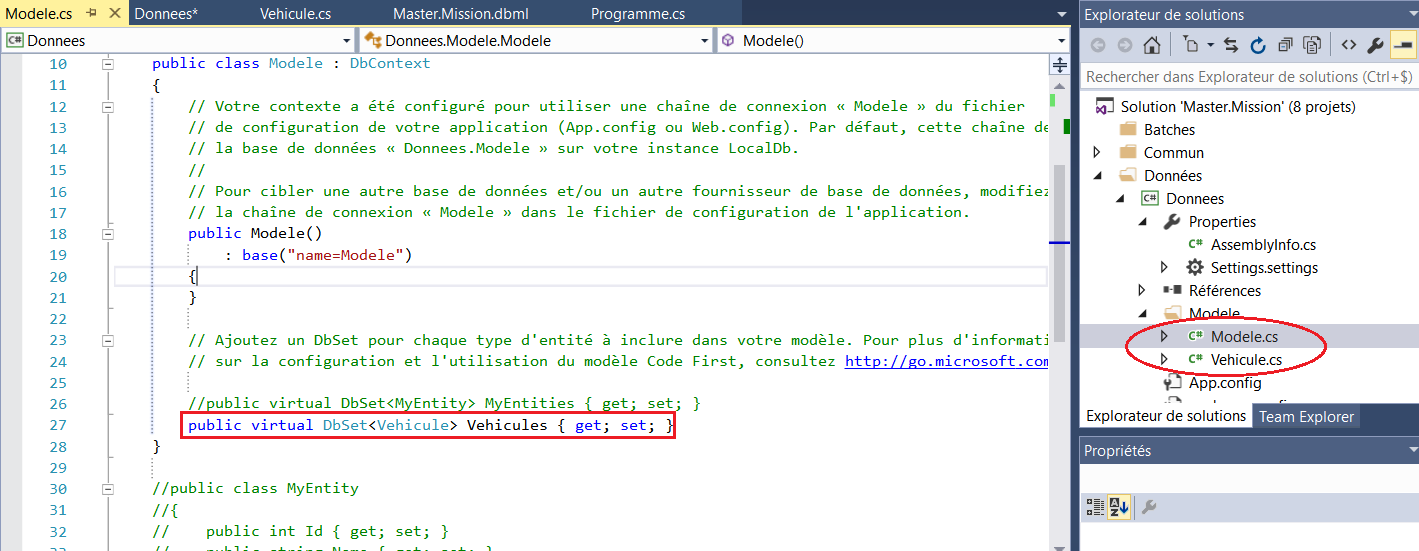


!!!Ajouter Modele dans Modele

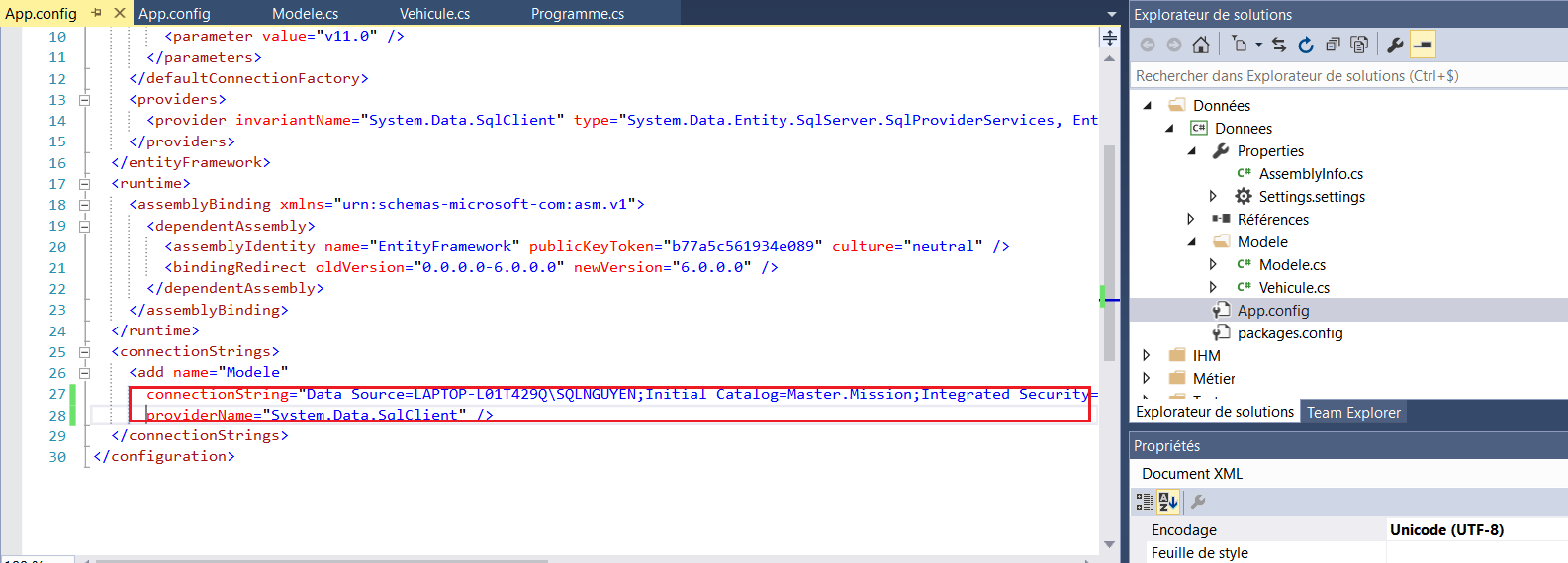
Virtual => abstract

Class Vehicule est public



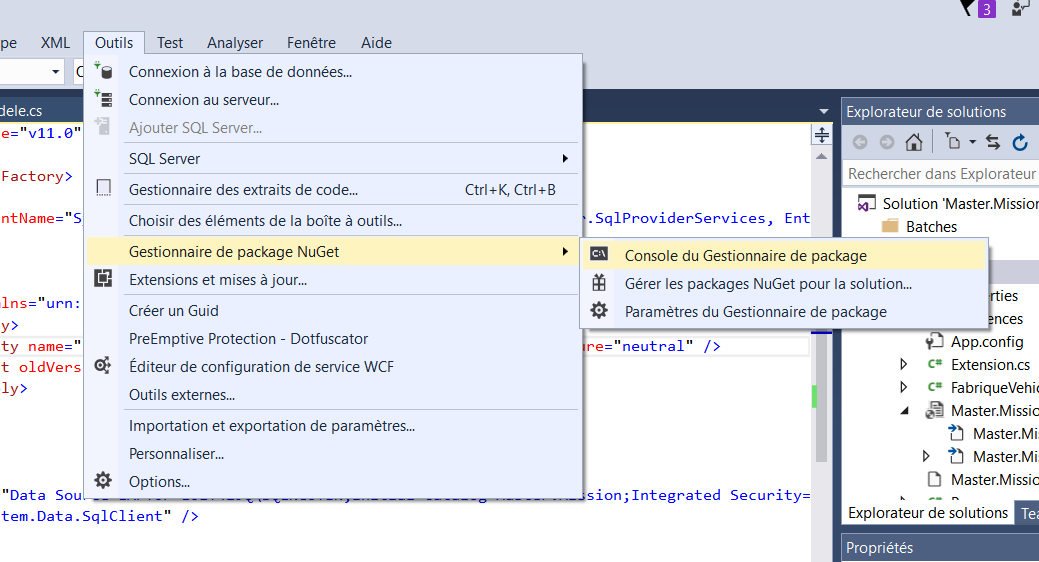


Copier « ConnectionString … »  dans app.conf(Outils) et puis coller dans App conf (Données)

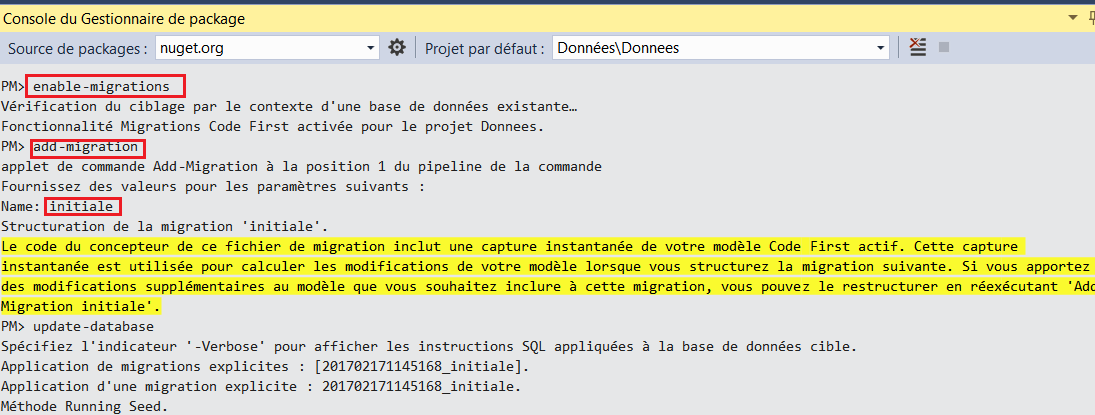
Final dans app.conf de Données

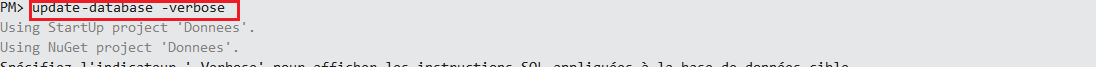
Code First Migration C# -> SQL

## Nuget/Console







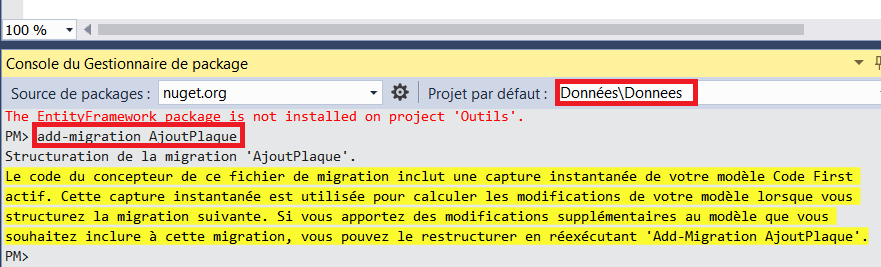


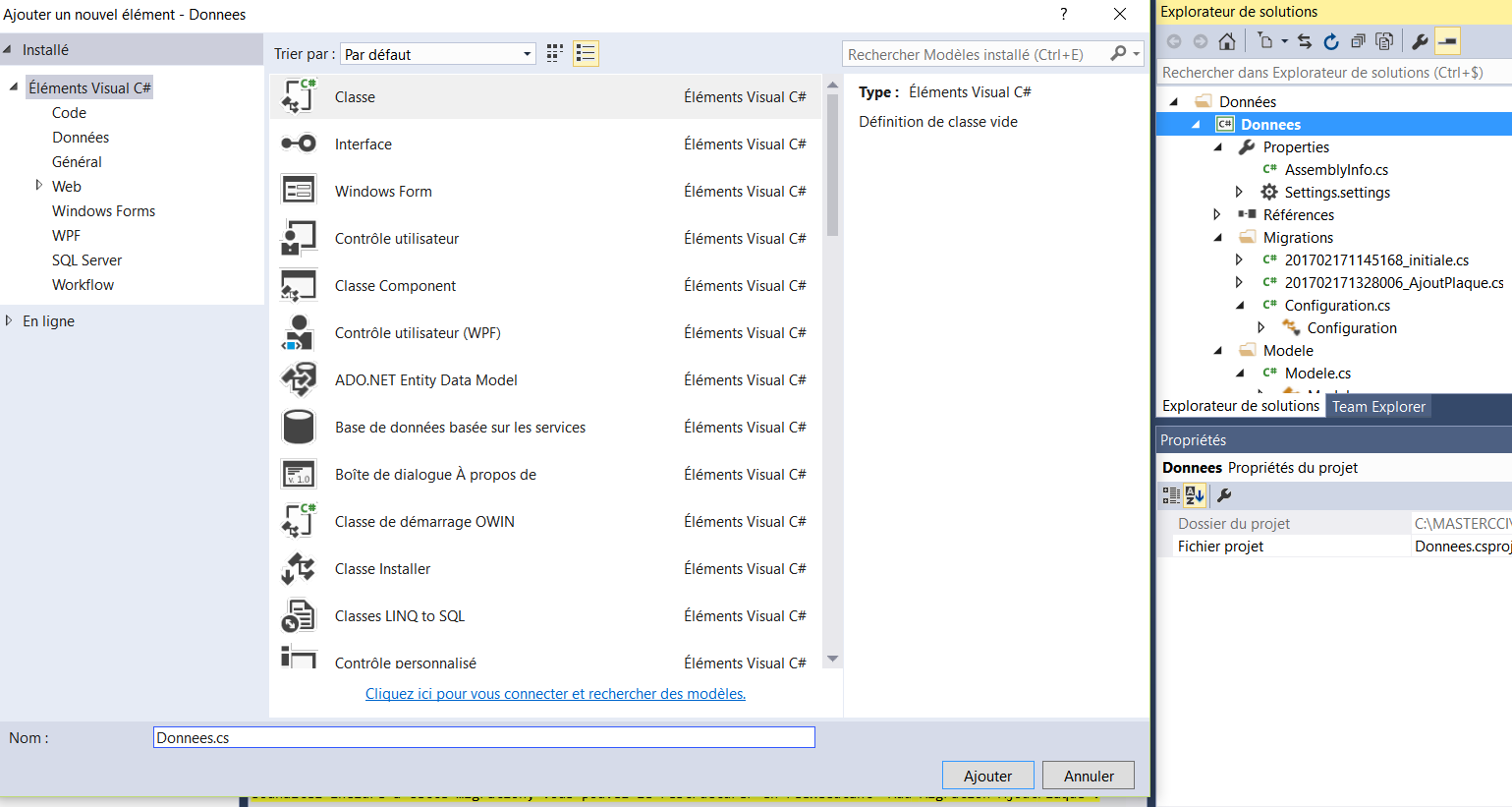
Si il ya des probleme => Cliquedroite le projet -> Ouvrir le dossier dans l’explotateur de fichier -> supprimer package

**SÉANCE 8**

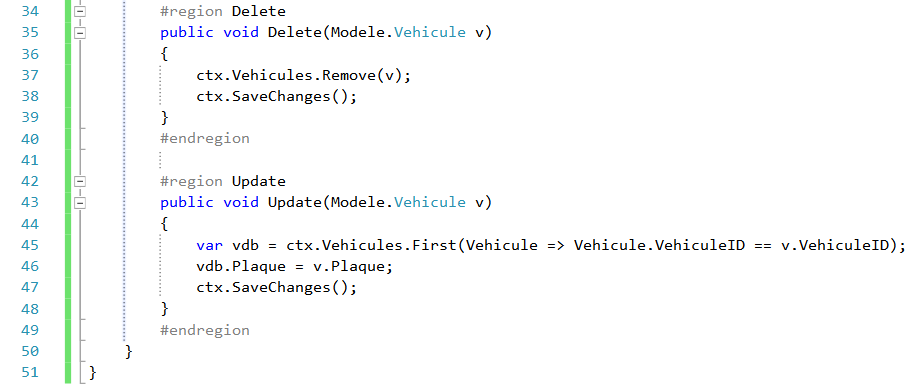
**Ajouter Plaque dans vehicule avec la meme méthode**











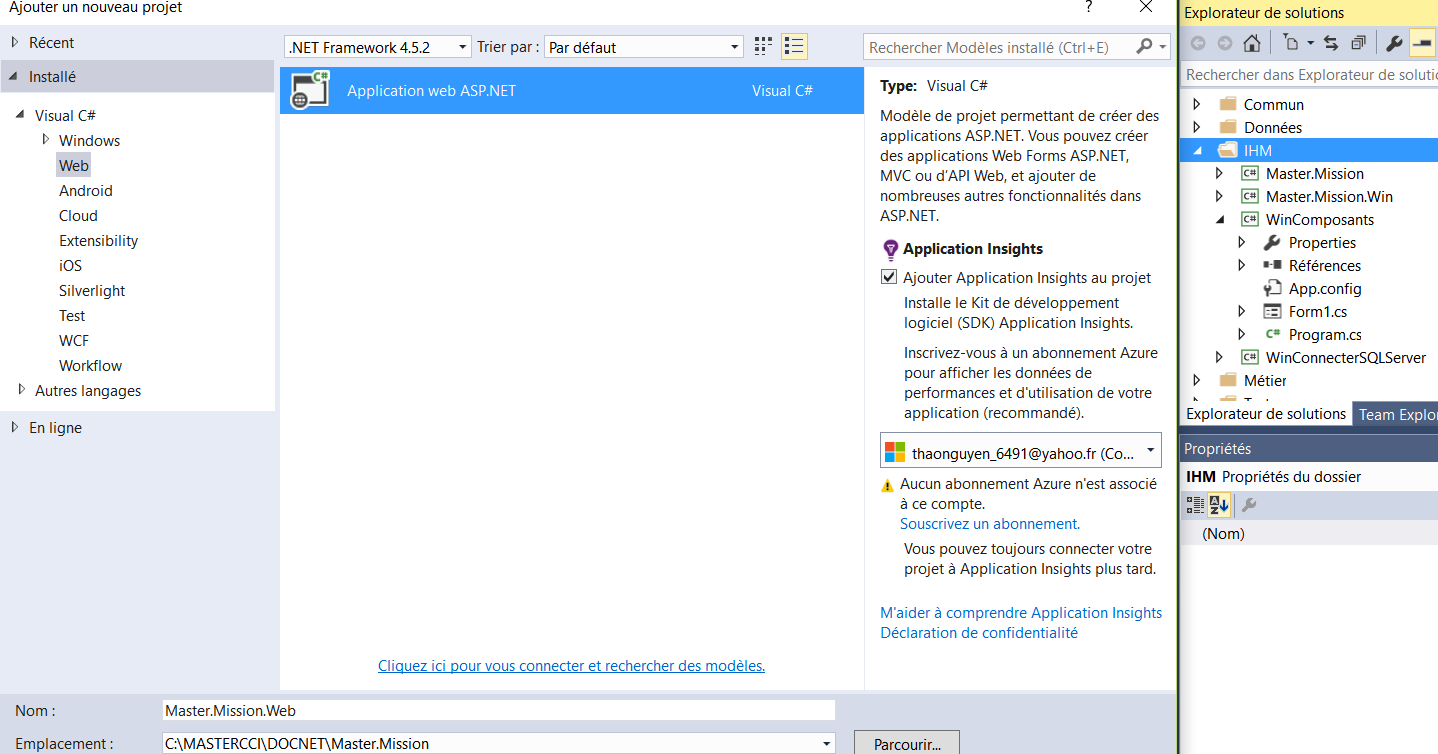
**Pour définir valeur NULL => Nullable ou int, ?**

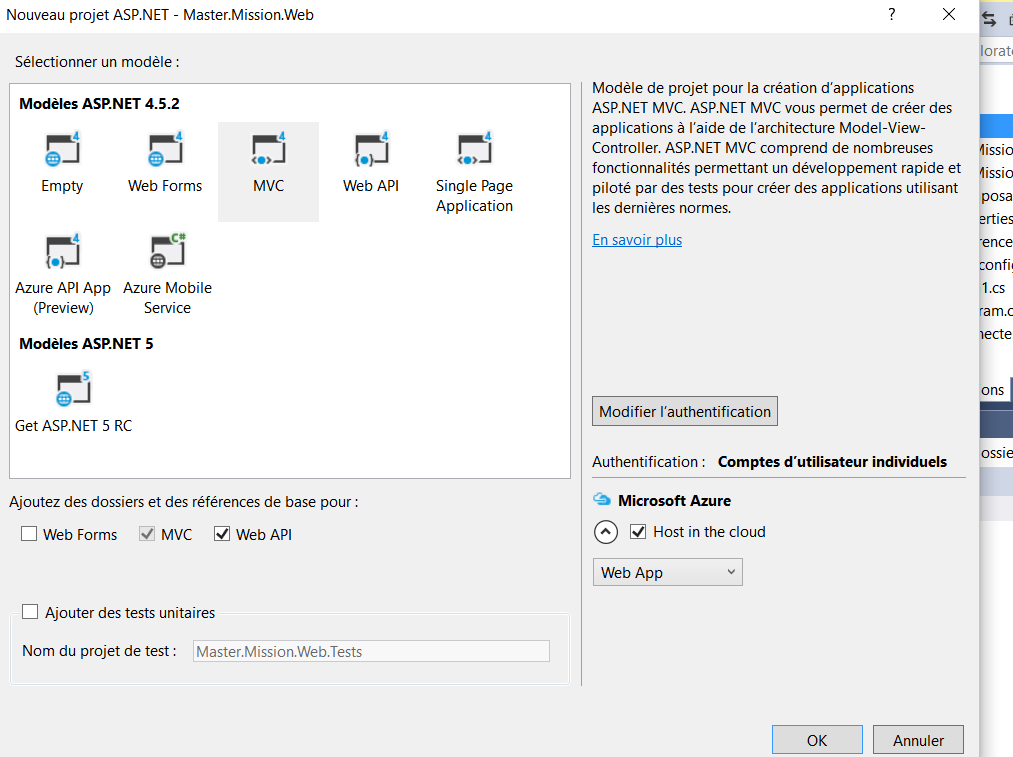
public Nullable<int> VehiculeID{ get ; set ;}

public virtual Vehicule Vehicule {get ;set;}

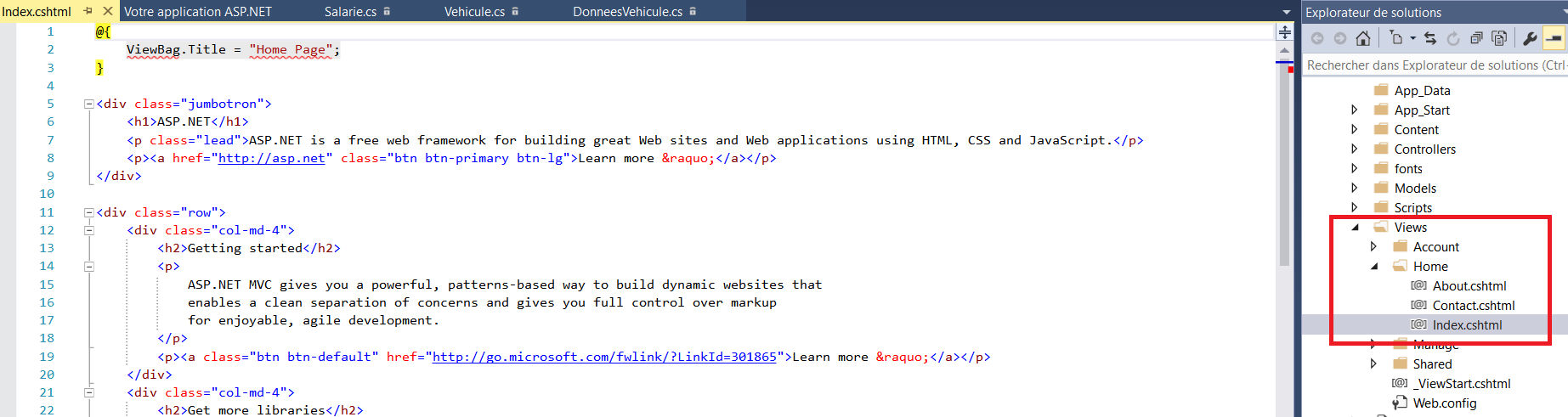
public List<Ligne> Lignes{ get ; se t ;}

**Créer WEB :**

****

****

* Choisir Definir comme projet de demarrage => demarer
* !!!Garder en mode fonctionner



=> Changer comme on veut

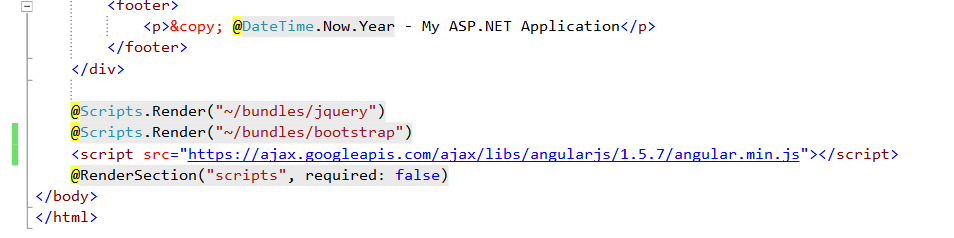


**Angular JS**

<https://developers.google.com/speed/libraries/>

Copier le code et coller dans layout





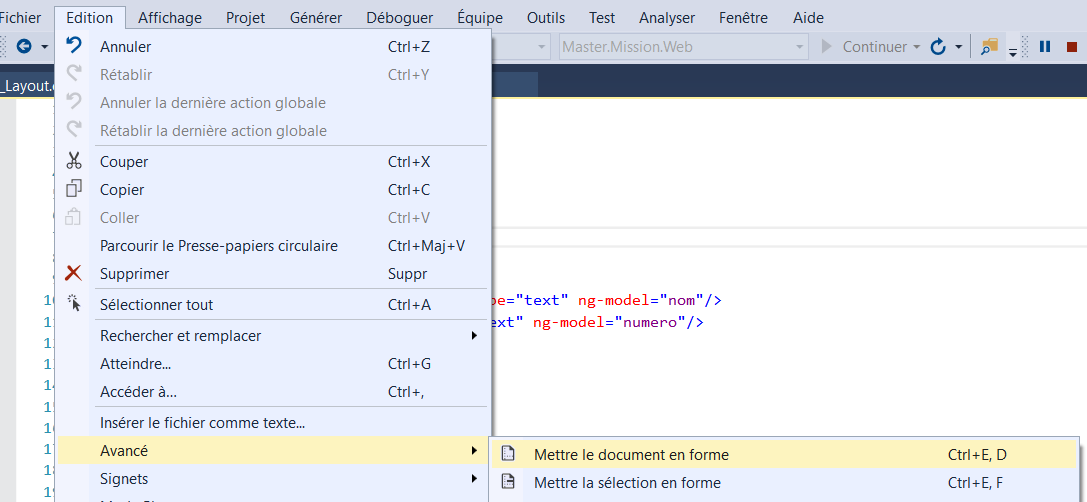
Les scripts dans index:



Les scripts dans Layout :



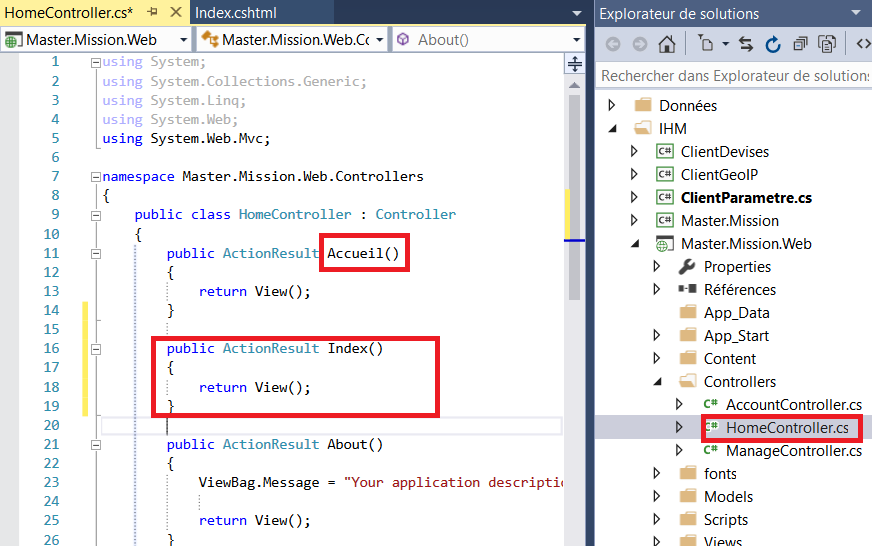
Pour mettre en forme



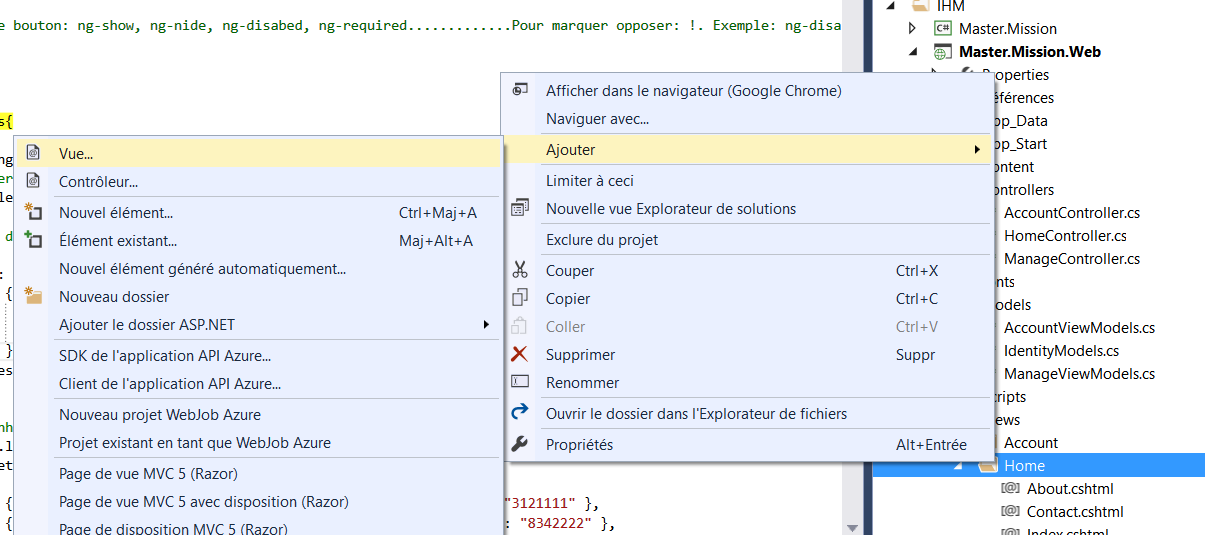


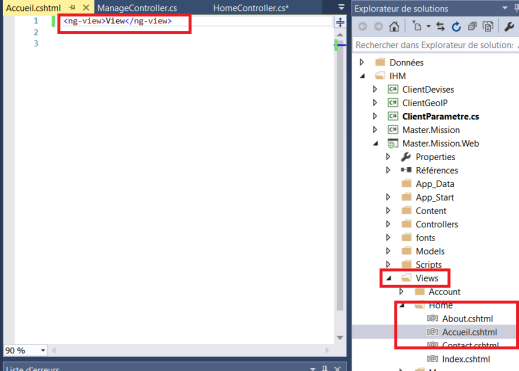


**La continuation**

****

Créer une vue et nommé Accueil

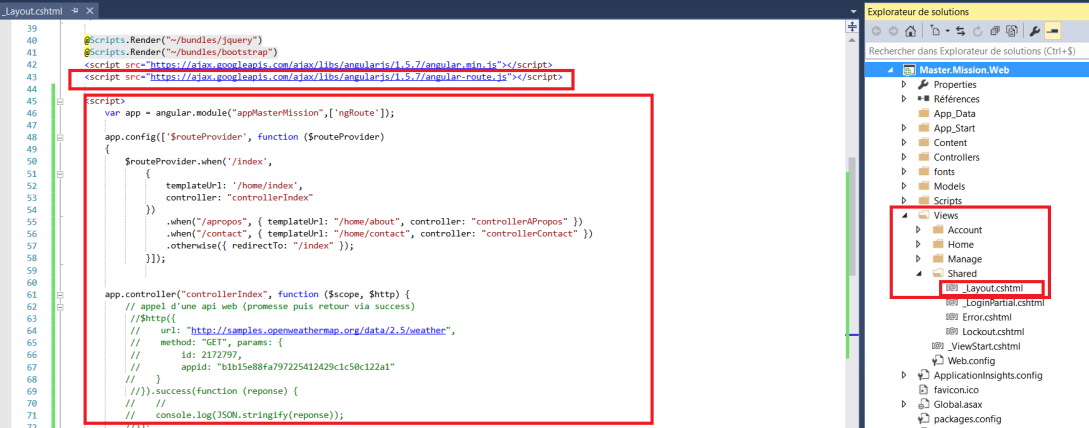
****

****

Tous les parties de app et app.config sont chargées dans layout standard grace au scripts



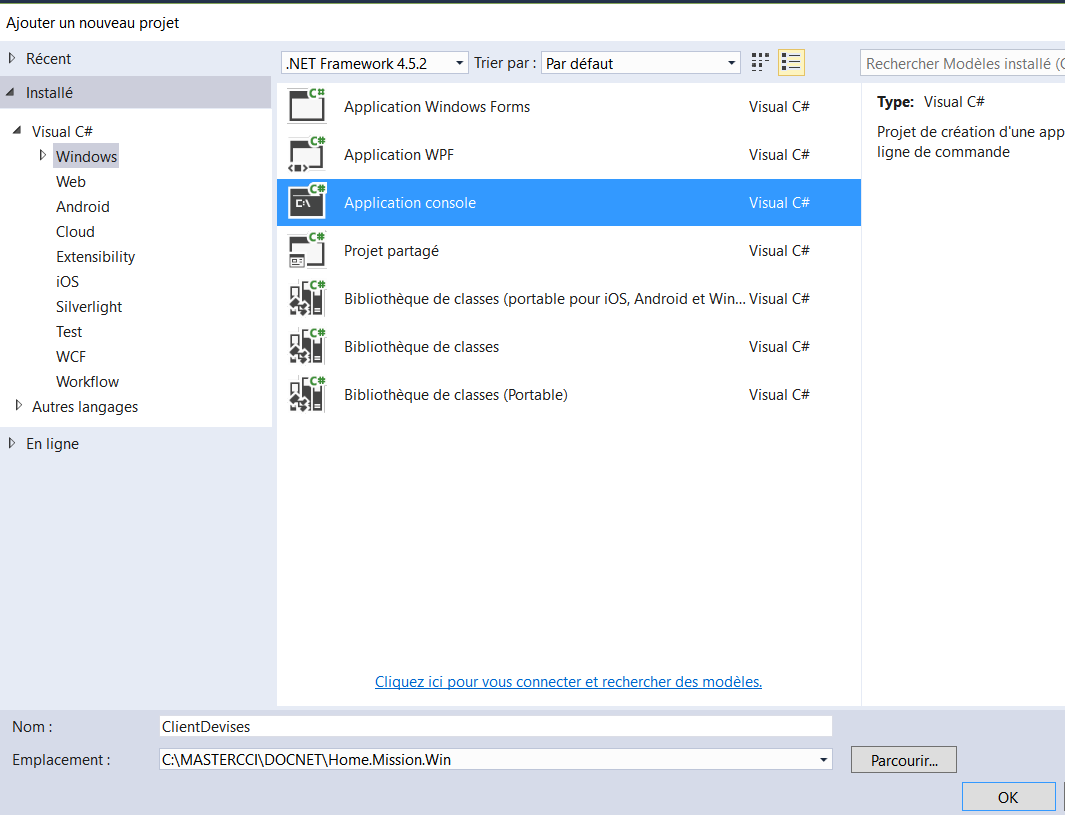
Remplace tous dan index vers layout . Le lancement peut etre plus lente mais





**Séance 9**

WebService : XMethods => webServiceX.net



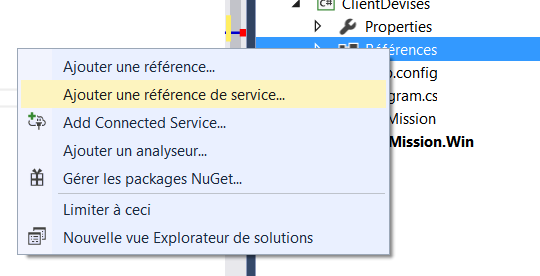
## Client devise

**Webservicex.net => Bussiness et commerce => Currency Convertor**

Copier le link : **http://www.webservicex.net/CurrencyConvertor.asmx?WSDL**

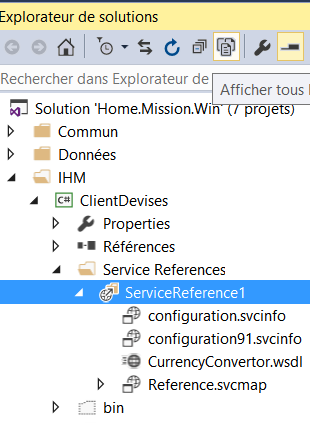
Ajouter dans le main : Console.ReadLine() ;

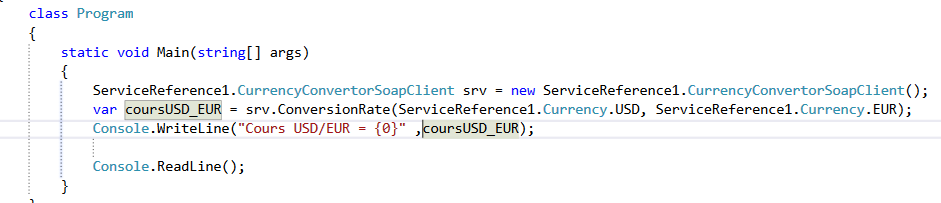
Ajouter une référence de service



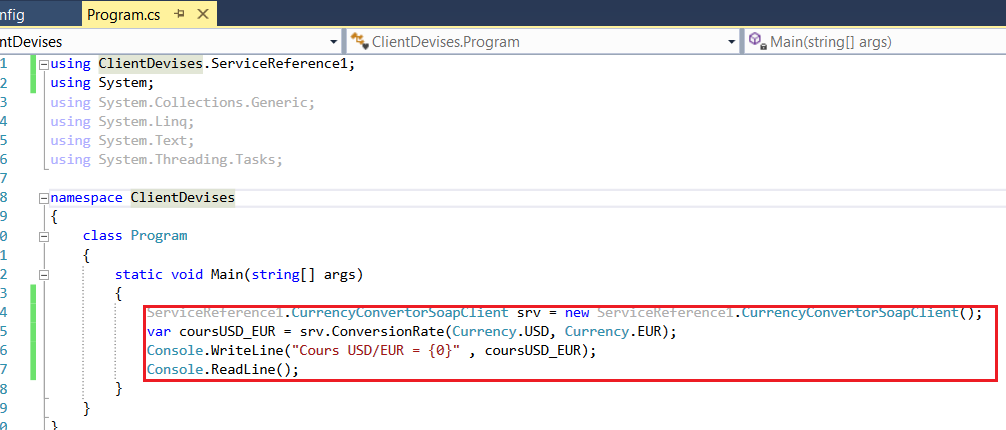
Coller le lien dans Adress => aller à => choisir ….httpGet …=> OK

Pour afficher tous





Si ça marrche pas , changer dans app.config



Essayer avec une autre commande :

var coursEUR\_MAD= srv.ConversionRate(Currency.EUR, Currency.MAD);

Console.WriteLine("Cours EUR/MAD = {0}", coursUSD\_EUR);

Console.ReadLine();

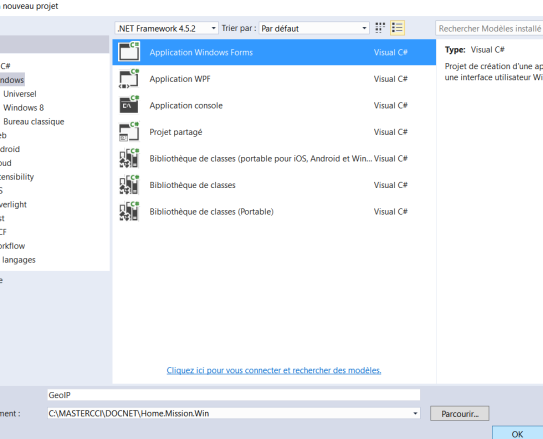
Pour essayer avec une autre types :

**Webservicex.net => Utilities => GlobalService**

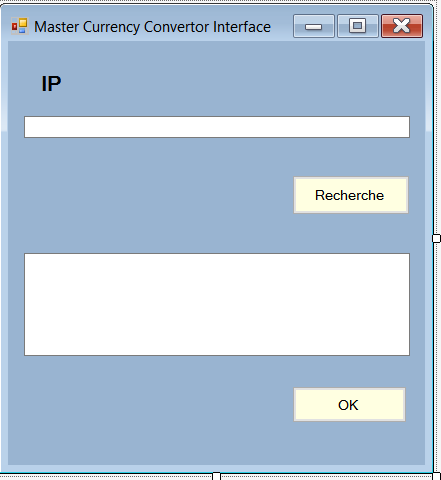
# Webservicex.net => Utilities => GeoIPService

## Application pour geolocaliser adresse IP de client

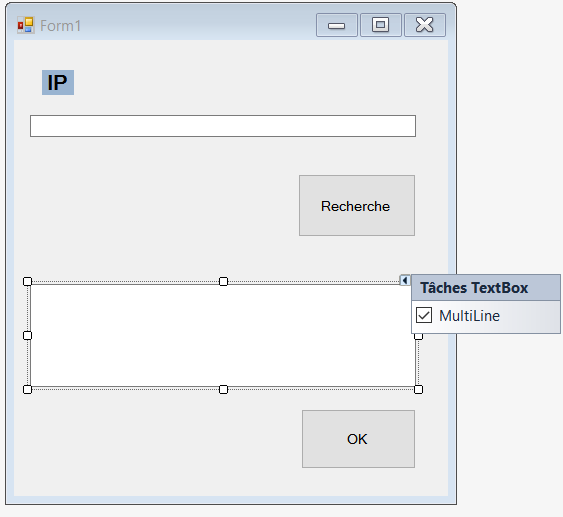
http://www.webservicex.net/geoipservice.asmx?WSDL



Créer un Winform

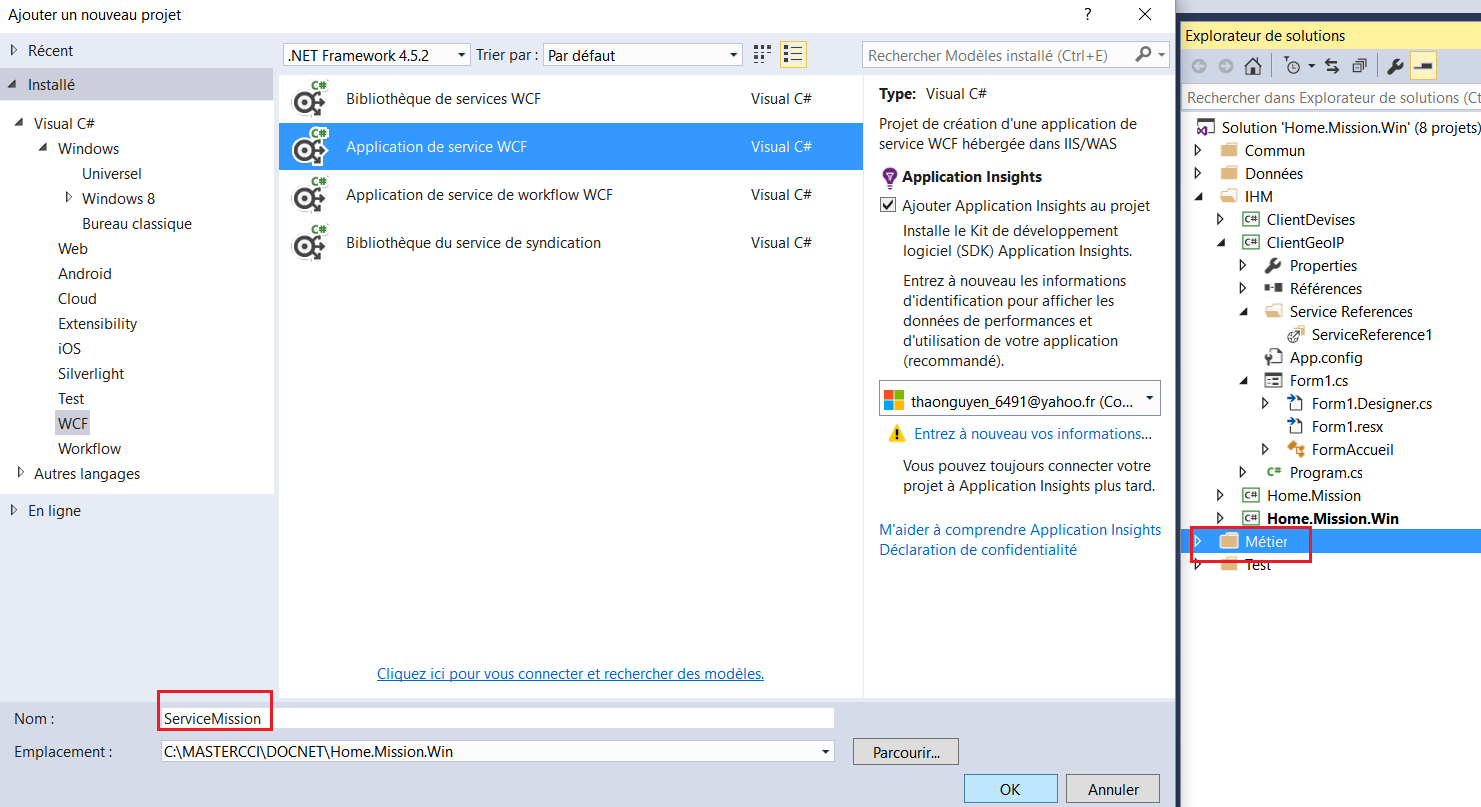


Pour textbox multiligne



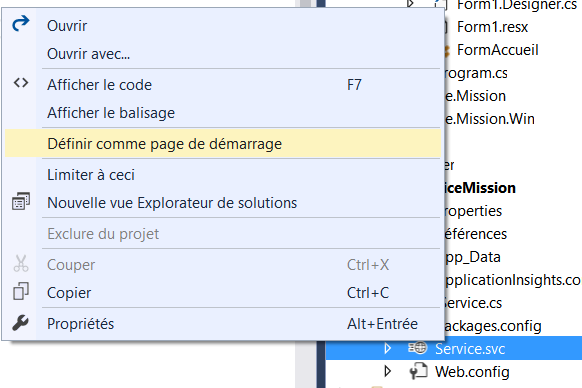


## SERVICE Application web

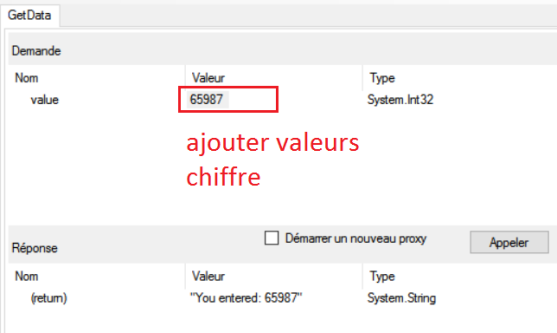


Puis, changer le nom , supprimer le 1

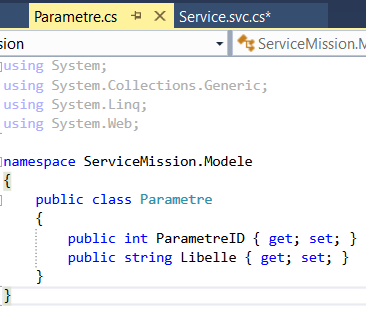
Ensuite activer Service.svc



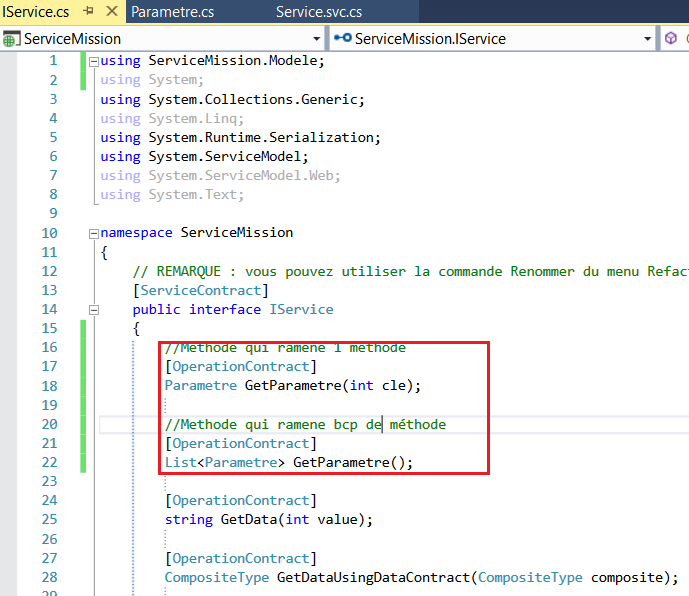
Pour tester



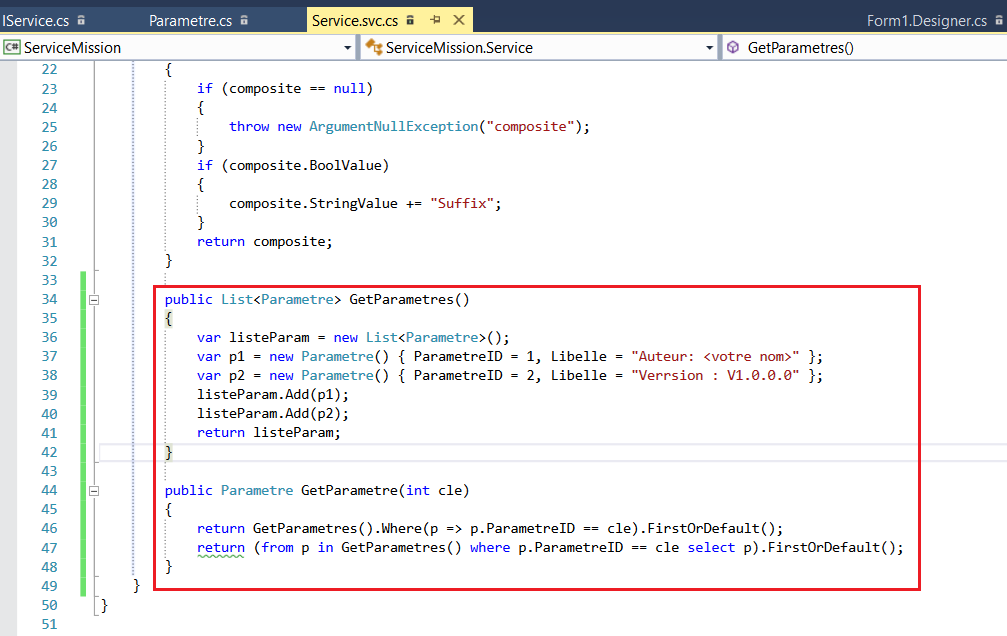
Créer un dossier Modele dans ServiceMission => Créer une classe « Parametre.cs »



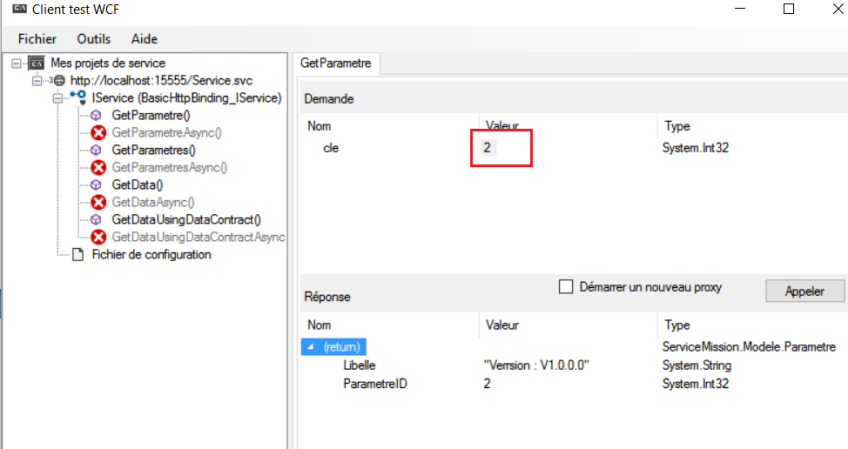
**Dans IService.cs :**

****

Implementer dans service.svc.cs => ampoule jauen = > automatique



Démarrer service.svc :



Créer un client : Client Parametres

* Consomme ServiceMission
* Affiche tous les parametres

Ajouter nouvelle console : « ClientParametre »

Clique droite => ajouter nouvelle reference web => decouvert= > choisir IService =>ok

Dans Main : ajouter



Projet de test : Representationnal State Transfer

REST API

Protocole http

URL -> /Vehicule

Méthodes -> get Post delete put

Get/Vehicule -> tous les Véhicules

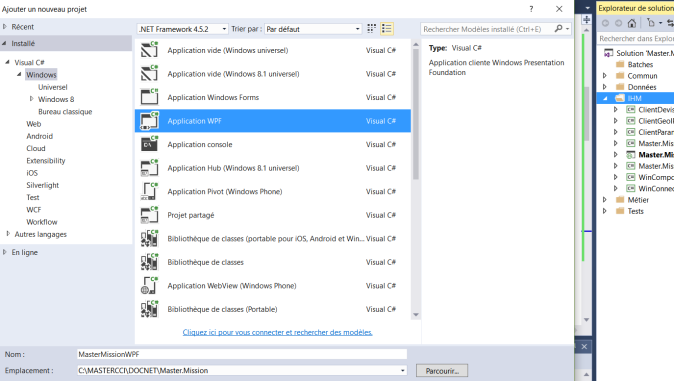
Get/Vehicule/1 -> 1 Véhicule

**OpenWeatherMap.org**

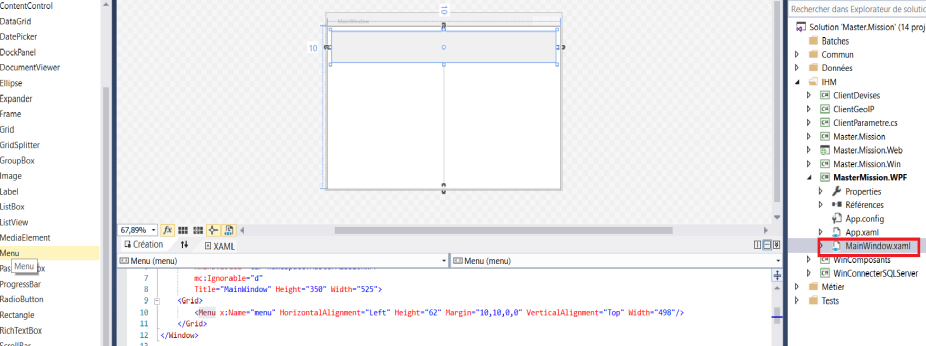
# SEANCE 10 : Application WPF, WPF/XBAP, déploiement click once, Application internet riche (RIA), Silverlight et au delà.

Stack Panel : pour les listes

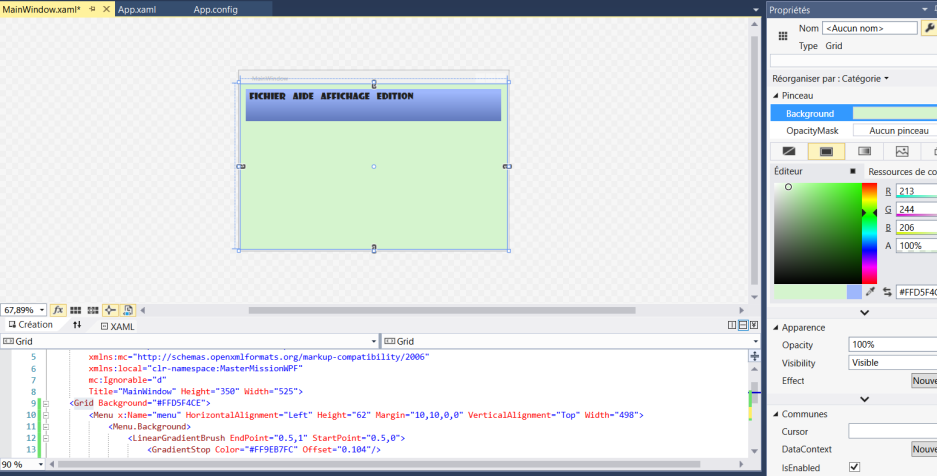
Créer un nouveau projet WPF => Exécuter



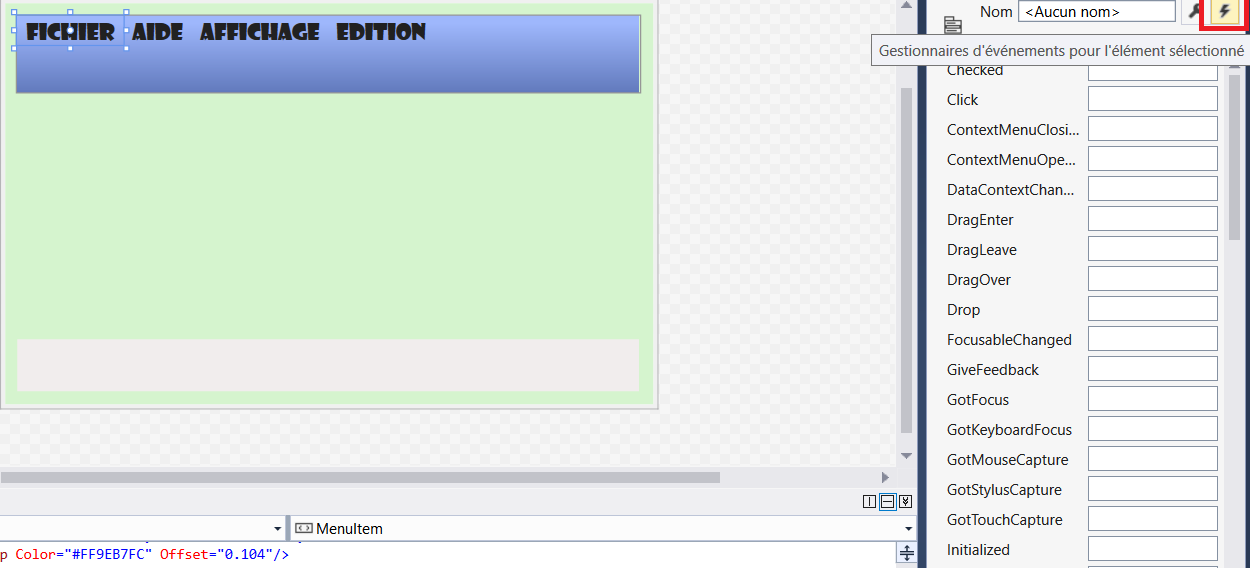
Dans MainWindow.xaml :



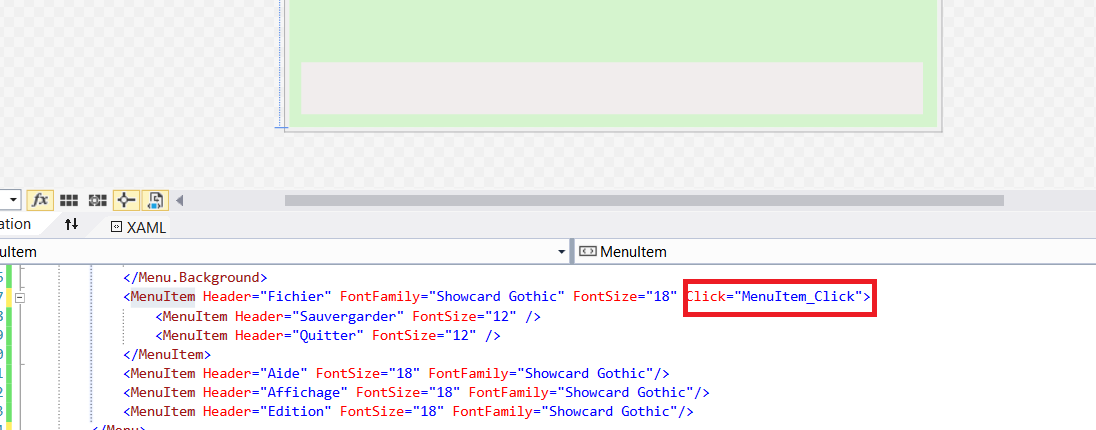


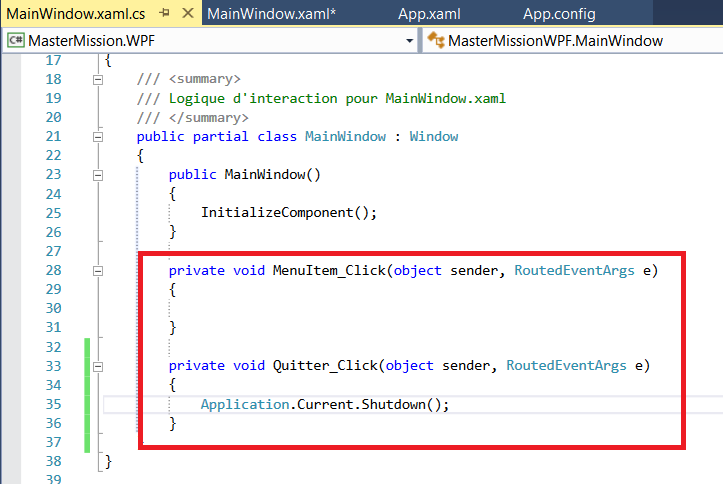


* Ajouter des sous items => faire comme les items

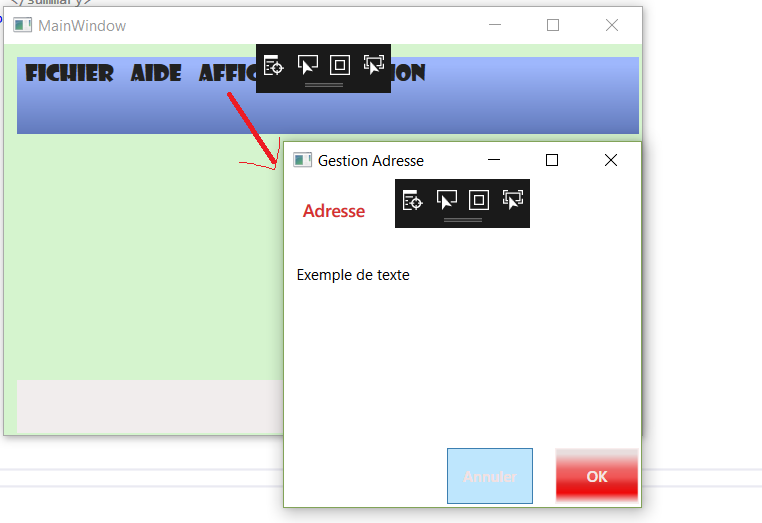


* Ajouter « MenuItem\_Click » dans Click => apparait comme ci-dessus

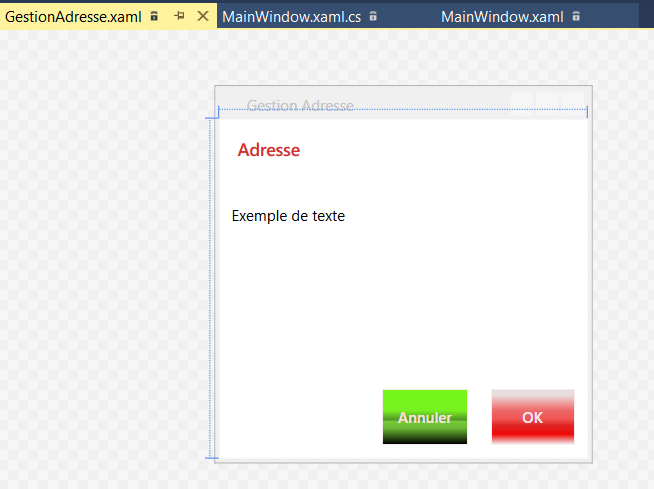




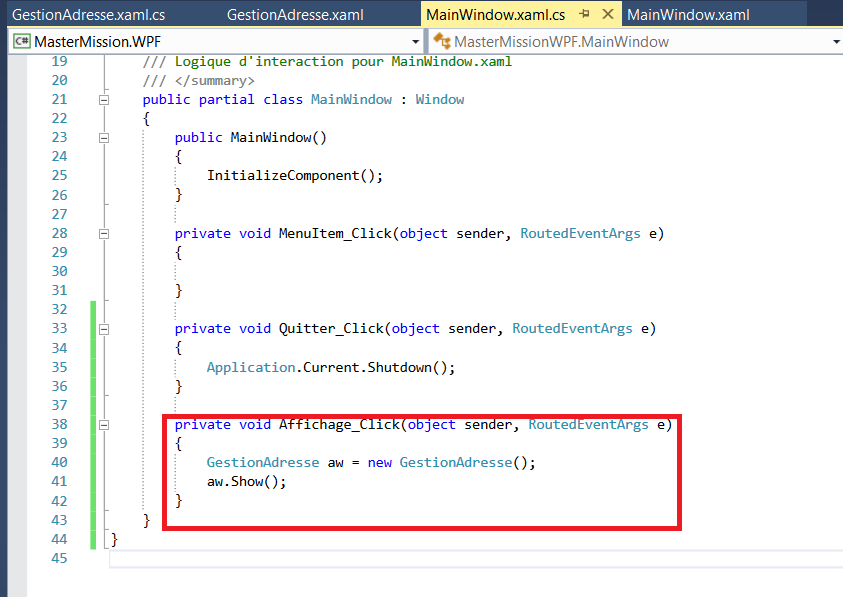
Autre exemple : Pour afficher deux tableaux consécutifs



Pour commencer

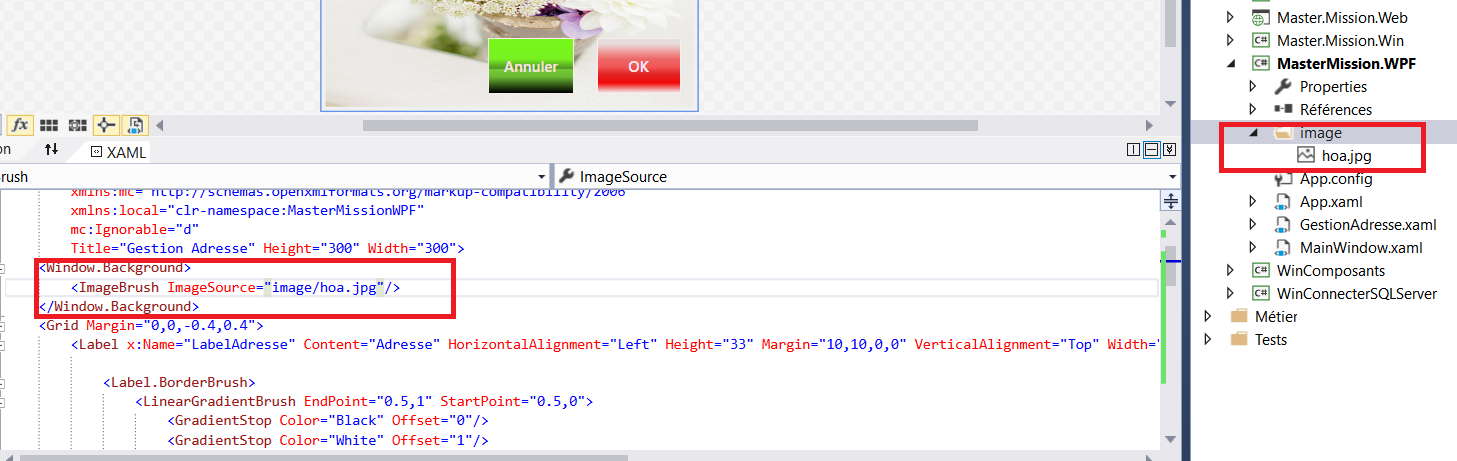




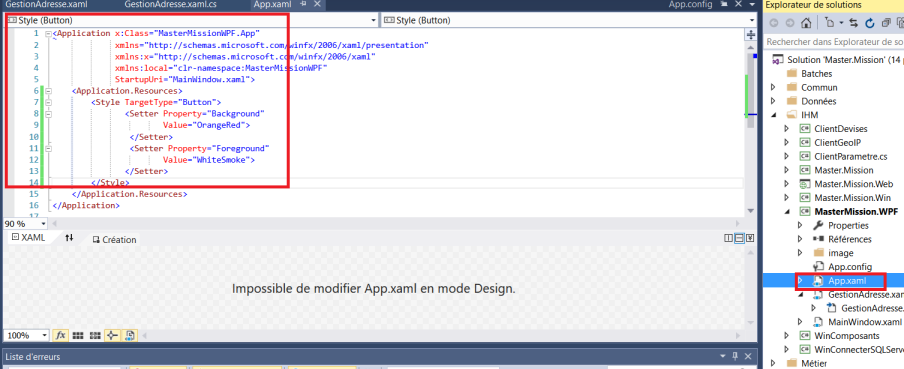


Pour ajouter les photos : créer un folder image => ajouter

Copier les commandes dans fichier .cs



Fichier de modifier les style



## Contrôle utilisateur WPF

## Application de navigateur wpf

Créer siteWeb :

visual studio.com