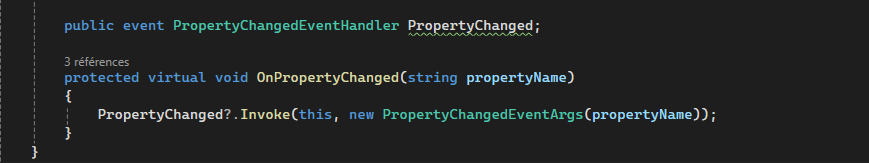
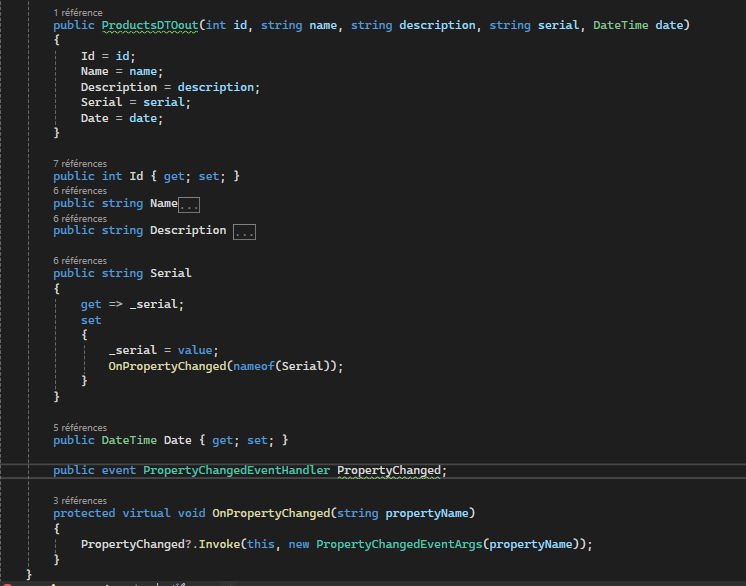
**1 – Implémentation de INotifyPropertyChanged au sein de la classe :**

Cette interface force l’implémentation de PropertyChangedEventHandler PropertyChanged, qui permet d’indiquer au client que l’une des propriétés de la classe a changé.



Dans le DTO, on établit la liste des propriétés et le constructeur. Puis on modifie le setter des propriétés en se servant de la méthode OnPropertyChanged.

Dans le MainWindow.xaml.cs :

* Dans le constructeur, en plus d’initialiser un composant on instancie un controller et on récupére la liste des DTO products, sous forme de d’une ObservableCollection de ProductDTOout qu’on met dans la propriété DataContext.
* public ObservableCollection<ProductsDTOout> DataContext { get; private set; }
* public MainWindow()
* {
* InitializeComponent();
* \_controller = new ProductsController();
* DataContext = new ObservableCollection<ProductsDTOout>(getListeProducts());
* }

Précisons que les listes fonctionnent également à la place des ObservableCollection.

private List<ProductsDTOout> getListeProducts()

{

List<ProductsDTOout> listeproducts = \_controller.GetAllProducts();

return listeproducts;

}

Ensuite Quand on clique sur le bouton update, alors on arrive dans la méthode qui va caster le selectedItem du DataGRiD en tant que DTOOut, et si la convesion réussi alors on retrouve une instance de l’objet créé, avec toutes ses propriétés.

private void btn\_update\_Click(object sender, RoutedEventArgs e)

{

ProductsDTOout p = dtg\_products.SelectedItem as ProductsDTOout;

// attribution des inputs ) l'objet

p.Name = txt\_Name.Text;

p.Description = txt\_Description.Text;

p.Serial = txt\_Serial.Text;

p.Date = Convert.ToDateTime(txt\_Date.Text);

Console.Write(p.Id);

\_controller.UpdateProduct(p.Id, p);

// récupération de la liste des produits

//getListeProducts();

// chargement de la nouvelle datagrid

//LoadDatas();

}

Puis on arrive dans le controller, qui va faire appel au service puis au contexte afin de mettre à jour la base de donnée ( ou le fichier json)

L’avantage de cette technique est que l’objet est mis à jour sans repasser par ke couche base de données : L’objet est bien modifié en bdd, mais la mise à jour se fait via la classe métier.

Pour formater correctement le fichier Data.json : Shift + Alt + f