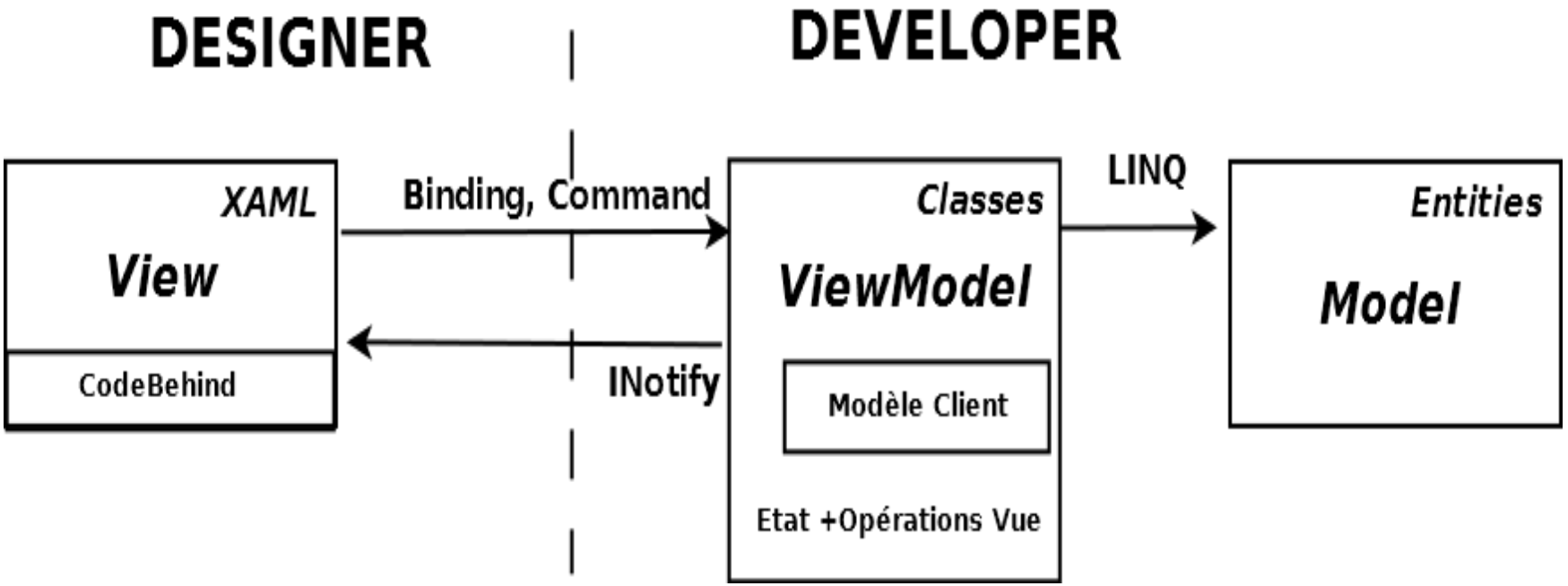
HAUTE ÉCOLE LÉONARD DE VINCI

WPF – MVVM - LINQ

- Notifications
- Command
- ObservableCollection
- Démo

Séparation en couches



HAUTE ÉCOLE LÉONARD DE VINCI

Notifications

- Interface INotifyPropertyChanged
- Event PropertyChangedEventHandler
- Création d'une méthode générique OnPropertyChanged

HAUTE ÉCOLE LÉONARD DE VINCI

Notifications

```
class LegumeModel : INotifyPropertyChanged {
   // Property changed standard handling
        public event PropertyChangedEventHandler PropertyChanged;
        // La view s'enregistera automatiquement sur cet event
        protected virtual void OnPropertyChanged(string propertyName)
        {
            if (PropertyChanged != null)
            {
                PropertyChanged(this, new PropertyChangedEventArgs(propertyName));
            }
        }
…
set { _legume.name = value; OnPropertyChanged("Name"); }
```

HAUTE ÉCOLE LÉONARD DE VINCI

WPF – MVVM - LINQ

- Gestion des événements
  - EventHandler
  - `<Button Click="Button_Click"`

Inconvénients EventHandler

- Liaison forte entre le code qui utilise(s'abonne) à un événement et le code qui propose l'événement
- Le code de traitement est dans le fichier XAML.cs

- … D'où idée de l'utilisation des **Command**

HAUTE ÉCOLE LÉONARD DE VINCI

ICommand

- `<Button Command="...."`
- Interface ICommand
  - Execute(Object)
  - CanExecute (Object)
  - Event handler CanExecuteChanged

HAUTE ÉCOLE LÉONARD DE VINCI

ICommand

- Interface ICommand
  - Ok mais alors pour chaque command (bouton), je dois créer une classe implémentant ICommand ?
  - En théorie oui, seulement c'est ingérable en pratique
  - En pratique, nous créons une seule classe implémentant ICommand à laquelle nous passerons la méthode à exécuter.

HAUTE ÉCOLE LÉONARD DE VINCI

DelegateCommand

```
public class DelegateCommand : ICommand
    {
        private Action _executeMethod;
        public DelegateCommand(Action executeMethod)
        {
            _executeMethod = executeMethod;
        }
        public bool CanExecute(object parameter)
        {
            return true;
        }
        public event EventHandler CanExecuteChanged;
        public void Execute(object parameter)
        {
            _executeMethod.Invoke();
        }
    }
```

HAUTE ÉCOLE LÉONARD DE VINCI

Command

- Ok mais comment lier tout ceci ?
  - `<Button Command="…"`
  - `Binding → DelegateCommand → Trt`

HAUTE ÉCOLE LÉONARD DE VINCI

Command

- Exemple
  - XAML : <Button Command="{Binding delCommand} »
  - ViewModel : propriété delCommand qui instancie une delegateCommand à laquelle on donne la méthode à exécuter (del)
  - ViewModel : définition d'une méthode del effectuant le traitement

HAUTE ÉCOLE LÉONARD DE VINCI

ObservableCollection

- Problème : maj dans le ViewModel d'une liste utilisée dans la vue → notifier la vue
  - Solution 1 : Notifications (InotifyPropertyChanged et/ou InotifyCollectionChanged)
  - Solution 2 : ObservableCollection (Implémente déjà InotifyCollectionChanged)