Implantation IESN

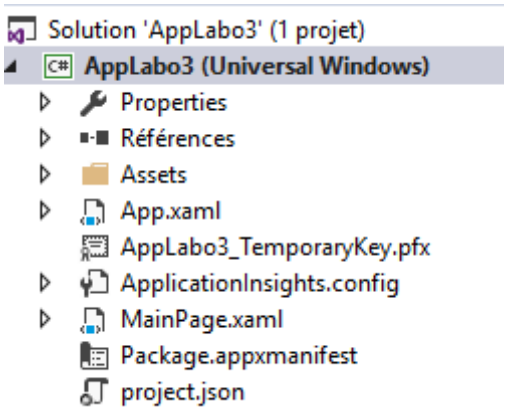
Environnement de développement de logiciels

IG3 — Labo 3 — 2016-2017

Objectifs

- Première approche du XAML
- Première approche du Data Binding
- Styles/Dictionary

Créez une nouvelle application de type UWP :



Le fichier MainPage.xaml correspond à :

```
<Page
    x:Class="AppLabo3.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:AppLabo3"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d">

    <Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    </Grid>
</Page>
```

Le fichier MainWindow.xaml.cs (code-behind) contient :

```
namespace AppLabo3
{
    /// <summary>
    /// Une page vide peut être utilisée seule ou constituer une page de destination au sein d'un frame.
    /// </summary>
    public sealed partial class MainPage : Page
    {
        public MainPage()
        {
            this.InitializeComponent();
        }
    }
}
```

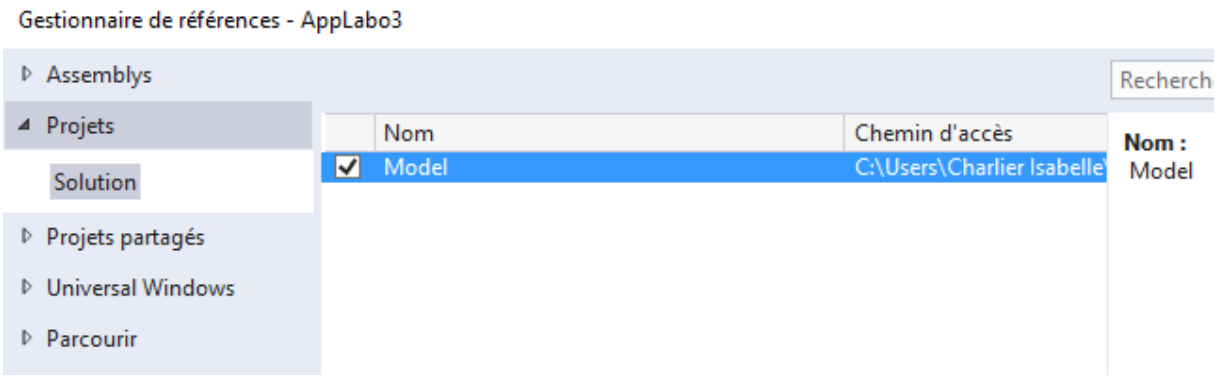
Nous modifierons par la suite pour que la grille comporte un titre, deux colonnes dont la première reprendra la liste des contacts et la seconde un contact sélectionné[1].

Ajoutez un projet à votre solution qui correspondra au modèle de données (couche Modèle ou Model).

Pour ce faire, dans cette application, vous ajoutez (click droit en étant sur la solution) un projet de type Class Library; ce projet portera le nom de Model. Vous avez ainsi deux projets dans la solution.

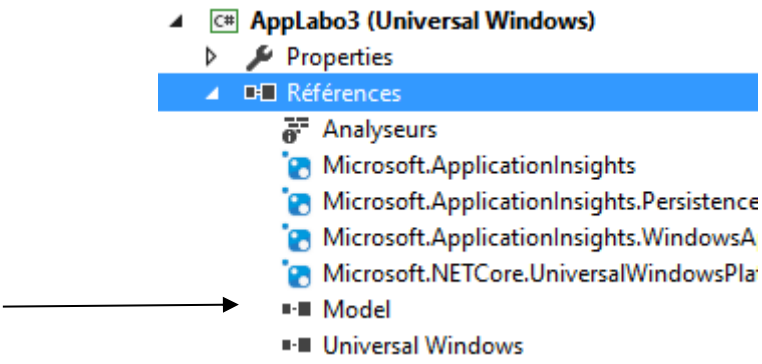
Dans le projet Model , créez une classe Contact.cs avec les propriétés FirstName, LastName et Email ainsi qu'un constructeur général.

Après avoir sélectionné References dans AppLabo3, click droit, Add Reference, Projects. Sélectionnez le projet Model.



On obtient :

[1] Cf. module controls



Ainsi, vous pouvez accéder aux classes de Model dans le projet AppLabo3.

Dans le code-behind associé à MainPage, c'est-à-dire dans MainPage.xaml.cs, ajoutez :

```
using Model;
```

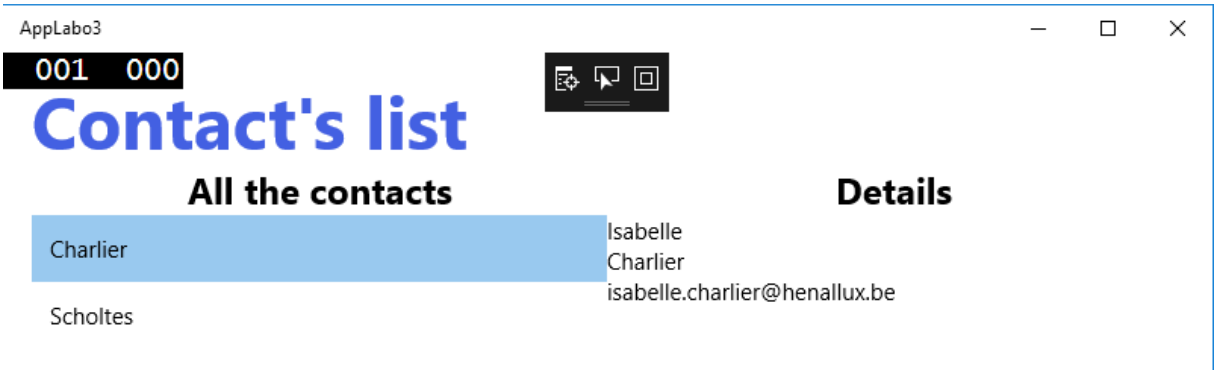
Ensuite, créez temporairement une liste de contacts à votre choix (voir ci-dessous).

```
public sealed partial class MainPage : Page
{
    private List<Contact> lstContacts = new List<Contact>()
    {
        new Contact("Charlier","Isabelle","isabelle.charlier@henallux.be"),
        new Contact("Scholtes","Samuel","samuel.scholtes@henallux.be")
    };

    public MainPage()
    {
        this.InitializeComponent();
        DataContext = lstContacts;
    }
}
```

La ligne DataContext servira à faire le lien entre la liste lstContacts ici créée et la ListView qui sera dans l'interface (dans le XAML).

Créez l'interface utilisateur via le code XAML de MainPage pour obtenir [2]:



[2] Interface non soignée : le but est d'apprendre quelques contrôles et le binding

Attirons votre attention sur quelques lignes xaml :

```
<ListView Grid.Row="1" ItemsSource="{Binding}" Name="Contacts">
    <ListView.ItemTemplate>
        <DataTemplate>
            <TextBlock Text="{Binding LastName}"/>
        </DataTemplate>
    </ListView.ItemTemplate>
</ListView>
```

Le fait d'écrire {Binding} lie cette ListView aux données inscrites dans le DataContext précédent.

```
<StackPanel Orientation="Vertical" Grid.Column="1" Grid.Row="1" DataContext="{Binding ElementName=Contacts,Path=SelectedItem }">
    <TextBlock Text="{Binding FirstName}"/>
    <TextBlock Text="{Binding LastName}"/>
    <TextBlock Text="{Binding Email}"/>
</StackPanel>
```

Ici, on lie à la listView nommée Contacts (voir propriété ElementName du Binding, qui correspond à la propriété Name donnée à la ListView) et on considère l'élément sélectionné. Le DataContext du StackPanel sera l'élément sélectionné dans la liste Contacts. L'élément en question sera de type Contact, puisque la liste affiche des éléments de type Contact.

Styles

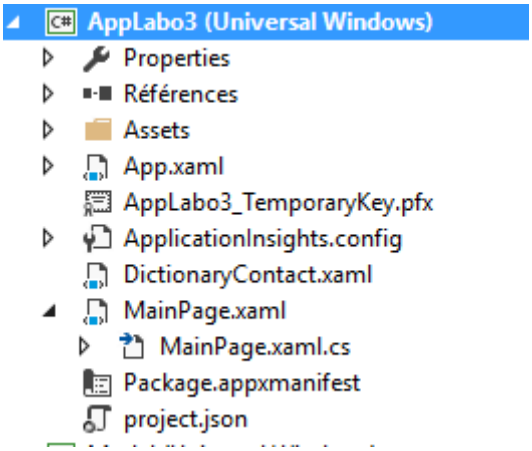
On souhaite que tous les textes qui sont de type TextBlock soient centrés.
Plusieurs solutions existent : la première étant de garnir la propriété TextAlignement avec le centrage et ce, à chaque champ de type TextBlock. Très lourd!

Autres solutions préconisées : via les styles.

Soit définir dans le même fichier XAML une ressource qui sera automatiquement appliquée.

```
<Page.Resources>
    <Style TargetType="TextBlock">
        <Setter Property="TextAlignment" Value="Center"/>
    </Style>
</Page.Resources>
```

Soit définir ce style dans un nouveau fichier de type ResourceDictionary , ici DictionaryContact à ajouter dans le projet :



Ensuite, modifiez le code xaml de la fenêtre :

```
<Page.Resources>
    <ResourceDictionary Source="DictionaryContact.xaml"/>
</Page.Resources>
```

Pour terminer, compléter le fichier DictionaryContact.xaml :

```
<ResourceDictionary
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:AppLabo3">
    <Style TargetType="TextBlock">
        <Setter Property="TextAlignment" Value="Center"/>
    </Style>
</ResourceDictionary>
```

Soit dans le fichier App.xaml où se trouvent tous les éléments qui concernent le xaml de tout le projet.