TP .Net Maui n° 2: Application à notes multiples

- AppShell.xaml :

```
<?xml version="1.0" encoding="UTF-8" ?>
<Shell
    x:Class="TPMaui2.AppShell"
    xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

    xmlns:view="clr-namespace:TPMaui2.Views"
    Shell.FlyoutBehavior="Disabled">

    <!--<ShellContent
        Title="Home"
        ContentTemplate="{DataTemplate view:Note}"
        Route="MainPage" />-->
    <TabBar>
        <ShellContent Title="Liste Notes"
            ContentTemplate="{DataTemplate view:ListeNotes}" Icon="icon_notes.png" />
        <ShellContent Title="Notes"
            ContentTemplate="{DataTemplate view:Note}" Icon="icon_notes.png" />
        <ShellContent Title="A Propos"
            ContentTemplate="{DataTemplate view:APropos}" Icon="icon_about.png" />
    </TabBar>
</Shell>
```

- Note.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:models="clr-namespace:TPMaui2.Models"
             x:Class="TPMaui2.Views.Note"
             Title="Note">
    <VerticalStackLayout Spacing="10" Margin="5">
        <Editor x:Name="TextEditor"  Placeholder="Taper votre note" HeightRequest="100"
Text="{Binding Text}"/>
        <Grid ColumnDefinitions="*,*" ColumnSpacing="4">
            <Button Text="Enregistrer" Clicked="ButtonEnregistrer_Clicked" />
            <Button Grid.Column="1"  Text="Supprimer"  Clicked="ButtonSupprimer_Clicked" />
        </Grid>
    </VerticalStackLayout>
</ContentPage>
```

- Note.xaml.cs

```
namespace TPMaui2.Views;
[QueryProperty(nameof(ItemId), nameof(ItemId))] // Précise que la page peut recevoir en paramètre
un itemId
public partial class Note : ContentPage
{
    string _fileName = Path.Combine(FileSystem.AppDataDirectory, "notes.txt");
    public string ItemId  // Précise une variable ItemId à la classe de la page Note
    {
```

```
        set { ChargerNote(value); }
    }

    public Note()
       {
            InitializeComponent();
        //if (File.Exists(_fileName))
        //    TextEditor.Text = File.ReadAllText(_fileName);

        Routing.RegisterRoute($"{nameof(ListeNotes)}", typeof(ListeNotes)); // Spécifie la route
vers ListeNotes
        string appDataPath = FileSystem.AppDataDirectory;
        string sFileName = $"{Path.GetRandomFileName()}.notes.txt";  // Nom de fichier aléatoire
xxxxx.notes.txt
        ChargerNote(Path.Combine(appDataPath, sFileName));
    }

    private void ChargerNote(string fileName)
    {
        Models.CNote noteModel = new Models.CNote();
        noteModel.Filename = fileName;
        if (File.Exists(fileName))
        {
            noteModel.Date = File.GetCreationTime(fileName);
            noteModel.Text = File.ReadAllText(fileName);
        }
        BindingContext = noteModel;
    }
    private async void ButtonEnregistrer_Clicked(object sender, EventArgs e)
    {

        if (BindingContext is Models.CNote note)
            File.WriteAllText(note.Filename, TextEditor.Text);
        await Shell.Current.GoToAsync("..");

        //File.WriteAllText(_fileName, TextEditor.Text); // Partie 1
    }
    private async void ButtonSupprimer_Clicked(object sender, EventArgs e)
    {
        if (BindingContext is Models.CNote note)
        {
            if (File.Exists(note.Filename))
            {
                File.Delete(note.Filename);
            }
        }
        await Shell.Current.GoToAsync("..");

        // Partie 1
        //if (File.Exists(_fileName))
        //                File.Delete(_fileName);
        //TextEditor.Text = string.Empty;

    }
}
```

- APropos.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:models="clr-namespace:TPMaui2.Models"
             x:Class="TPMaui2.Views.APropos"
             Title="A Propos">
    <ContentPage.BindingContext>
        <models:CAPropos />
    </ContentPage.BindingContext>
```

```
    <VerticalStackLayout Spacing="10" Margin="10">
        <Image Source="logo_groupe3il.jpg"
        SemanticProperties.Description="Cette application est une production 3iL"
        HeightRequest="64" HorizontalOptions="CenterAndExpand"/>
        <!--<Label FontSize="22" FontAttributes="Bold" Text="Notes 3iL" VerticalOptions="End"
        TextColor="{StaticResource Secondary}"/>
        <Label FontSize="22" Text="v1.0" VerticalOptions="End" TextColor="{StaticResource
        Secondary}"/>
        <Label Text="Application XAML/C# écrite en .NET MAUI."
        TextColor="{StaticResource Primary}"/>-->
        <Label FontSize="22" FontAttributes="Bold" Text="{Binding Title}" VerticalOptions="End"
            TextColor="{StaticResource Secondary}"/>
        <Label FontSize="22" Text="{Binding Version}" VerticalOptions="End"
            TextColor="{StaticResource Secondary}"/>
        <Label Text="{Binding Message}" TextColor="{StaticResource Primary}"/>
        <Button Text="En savoir plus sur 3iL" Clicked="APropos_Clicked" />
    </VerticalStackLayout>
</ContentPage>
```

- APropos.xaml.cs

```
namespace TPMaui2.Views;

public partial class APropos : ContentPage
{
       public APropos()
       {
             InitializeComponent();
       }

    private async void APropos_Clicked(object sender, EventArgs e)
    {
        //await Launcher.Default.OpenAsync("https://www.3il-ingenieurs.fr/");
        if (BindingContext is Models.CAPropos apropos)
        {
            await Launcher.Default.OpenAsync(apropos.MoreInfoUrl);
        }

    }
}
```

- CPropos.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TPMaui2.Models
{
    internal class CAPropos
    {
        public string Title => "Notes 3iL - " + AppInfo.Name;
        public string Version => AppInfo.VersionString;
        public string MoreInfoUrl => "https://www.3il-ingenieurs.fr/";
        public string Message => "Application XAML/C# écrite en .NET MAUI.";
    }
}
```

- CNote.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;

namespace TPMaui2.Models
{
    internal class CNote
    {
        public string Filename { get; set; }
        public string Text { get; set; }
        public DateTime Date { get; set; }


    }
}
```

- ListeNotes.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="TPMaui2.Views.ListeNotes"
             Title="Liste des Notes">
    <VerticalStackLayout>
        <CollectionView x:Name="CollectionDeNotes"
SelectionChanged="CollectionDeNotes_SelectionChanged"
                ItemsSource="{Binding CollNotes}" Margin="20" SelectionMode="Single">
                <CollectionView.ItemsLayout>
                    <LinearItemsLayout Orientation="Vertical" ItemSpacing="10" />
                </CollectionView.ItemsLayout>
            <CollectionView.ItemTemplate>
                <DataTemplate>
                    <StackLayout>
                        <Label Text="{Binding Text}" FontSize="22"/>
                        <Label Text="{Binding Date}" FontSize="14" TextColor="Silver"/>
                    </StackLayout>
                </DataTemplate>
            </CollectionView.ItemTemplate>
        </CollectionView>
        <Label
            Text="Welcome to .NET MAUI!"
            VerticalOptions="Center"
            HorizontalOptions="Center" />
    </VerticalStackLayout>
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="Ajouter" Clicked="Ajouter_Clicked" IconImageSource="{FontImage
                Glyph='+', Color=White, Size=22}" />
    </ContentPage.ToolbarItems>
</ContentPage>
```

- ListeNotes.xaml.cs

```
//using static Android.Provider.ContactsContract.CommonDataKinds;


namespace TPMaui2.Views;

public partial class ListeNotes : ContentPage
{
       public ListeNotes()
       {
             InitializeComponent();
        Routing.RegisterRoute($"{nameof(Note)}", typeof(Note)); // Spécifie la route vers la page
Note
        BindingContext = new Models.CListeNotes();
```

```
    }

    private async void Ajouter_Clicked(object sender, EventArgs e)
    {
        await Shell.Current.GoToAsync(nameof(Note));
    }

    private async void CollectionDeNotes_SelectionChanged(object sender,
SelectionChangedEventArgs e)
    {
        if (e.CurrentSelection.Count > 0) // Il existe au moins une note
        {
            var sNote = (Models.CNote)(e.CurrentSelection[0]);
            await
Shell.Current.GoToAsync($"{nameof(Note)}?{nameof(Note.ItemId)}={sNote.Filename}");
            CollectionDeNotes.SelectedItem = null;
        }
    }

    // avec await Shell.Current.GoToAsync(".."); dans le bouton Enregistrer de la page Note
    protected override async void OnAppearing() // Est appelé chaque fois que la page apparaît
    {
        base.OnAppearing();
        BindingContext = new Models.CListeNotes();
    }
}
```

- CListeNotes.cs

```
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
//using static Xamarin.Google.Crypto.Tink.Mac.AesCmacParameters;

namespace TPMaui2.Models
{


    internal class CListeNotes
    {
        public ObservableCollection<CNote> CollNotes { get; set; } = new
ObservableCollection<CNote>();
        // collNotes est une collection accessible d'objets de classe CNote
        public CListeNotes() => ChargerNotes();  // précise que l'appel au constructeur de fait
qu'appeler ChargerNotes


        public void ChargerNotes()
        {
            string appDataPath = FileSystem.AppDataDirectory; // Répertoire de l'application

            IEnumerable<CNote> listenotes = Directory.EnumerateFiles(appDataPath, "*.notes.txt")
                                                     .Select(filename => new CNote()
                                                     {
                                                         Filename = filename,
                                                         Text =
File.ReadAllText(filename),
                                                         Date =
File.GetCreationTime(filename)
                                                     })
                                                     .OrderBy(note => note.Date);
            // Directory.EnumerateFiles(appDataPath, "*.notes.txt") renvoie tous les ficjier du
répertoire appDataPath et
            // de type *****.notes.txt
```

```
            // .Select(filename => new CNote() indique que les fichier vont être renvoyés sous
forme d'objet CNote
            // en respectant les préconisations entre les {} Filename reçoit le nom du fichier,
Text le contenu et
            // Date la date de création

            CollNotes.Clear();
            foreach (var listenote in listenotes)
            {
                CollNotes.Add(listenote);
            }

        }
    }
}
```