Le développement cross plateforme avec Xamarin

Module 6 – Développement de contrôleurs Xamarin.Forms

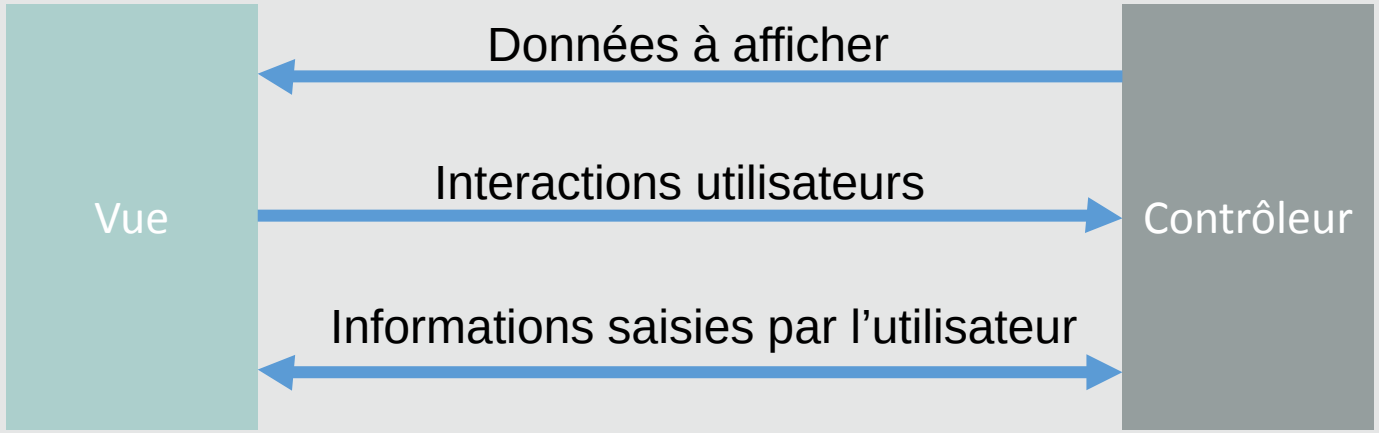
Objectifs

- Savoir déclencher une action au clic sur un bouton
- Savoir récupérer les valeurs saisies par l'utilisateur dans les champs de saisie
- Savoir alimenter les composants de la vue avec des données

Rôle du contrôleur dans une application Xamarin.Forms

Un contrôleur Xamarin.Forms a pour rôle d'alimenter la vue en données, de récupérer les informations saisies par l'utilisateur, de les vérifier et enfin de réagir aux interactions

Rôle du contrôleur dans une application Xamarin.Forms



Rôle du contrôleur dans une application Xamarin.Forms

Le contrôleur n'a pas vocation à réaliser des calculs complexes, à effectuer des choix métier, à interroger une base de données ou une API

Une fois les données récupérées, vérifiées et transformées, le contrôleur s'appuie sur la couche de services pour réaliser toutes les opérations métier

Réagir au clic sur un bouton

Xamarin.Forms, comme les technologies natives qu'il surcharge, est basé sur un système d'évènements.

Les composants interactifs proposent de s'abonner à des évènements (clic, saisie, touché, glissement, etc.) et d'appeler une fonction dans le contrôleur au moment où l'évènement est déclenché.

Réagir au clic sur un bouton

Dans le cas d'un bouton, l'attribut **Clicked**, associé à une méthode **public** dans le contrôleur présentant une signature définie, permet de réagir à un clic de l'utilisateur

```
<Button Text="Clic Me!" Clicked="ClicMe_Clicked"/>

public void ClicMe_Clicked(object sender, EventArgs e)
{
}
```

Réagir au clic sur un bouton

La méthode qui est appelée au clic sur l'évènement expose deux arguments :

- un de type **object** (**sender**)
- et l'autre de type **EventArgs** (**e**)

Le premier est une référence vers le composant qui a émis l'évènement. Il doit être casté pour être utilisé.

```
private void ClicMe_Clicked(object sender, EventArgs e)
{
    (sender as Button).Text = "I was just clicked!";
}
```

Réagir au clic sur un bouton

Le deuxième argument est un paramètre facultatif qui peut servir à transmettre un identifiant, un mode ou toute autre information utile au traitement de l'évènement

Réagir au clic sur un bouton

- D'autres évènements que **Clicked** peuvent être positionnés sur un bouton :
  - **Pressed** : l'utilisateur appuie sur le bouton
  - **Released** : l'utilisateur relâche le bouton
  - **Clicked** : l'utilisateur a effectué un cycle pression/relâchement sur le bouton

Lire les valeurs d'un composant visuel

Tout composant se voit attribuer différentes valeurs à son initialisation, à travers la vue XAML.

Ces valeurs peuvent être amenées à changer au cours de la vie de la page, par une saisie utilisateur par exemple.

Le contrôleur doit pouvoir alors récupérer ces valeurs afin de les valider, les transformer et éventuellement les transmettre à la couche service.

Lire les valeurs d'un composant visuel

Pour pouvoir interroger un composant depuis le contrôleur, il suffit de le nommer depuis la vue XAML.

Le fichier XAML étant par la suite transformé en code C#, le compilateur se charge de créer un attribut C# pour chaque composant nommé dans le XAML.

```
<Label x:Name="titre"></Label>
```

Lire les valeurs d'un composant visuel

Une fois la solution regénérée, un attribut correspondant au nom donné au composant est disponible.

Cette référence peut être utilisée pour récupérer ou positionner toutes les valeurs qui peuvent être positionnées dans le XAML.

L'intégralité de l'API du composant est à ce moment accessible, contrairement au XAML qui ne donne accès qu'à une partie des propriétés d'un composant.

Lire les valeurs d'un composant visuel

```
private void ClicMe_Clicked(object sender, EventArgs e)
{
        Console.WriteLine(this.titre.Text.ToString());
        Console.WriteLine(this.titre.FontSize);
        Console.WriteLine(this.titre.TextColor);
}
```

Mettre à jour les valeurs d'un composant visuel

Pour mettre à jour les valeurs d'un composant visuel, la même stratégie est appliquée que pour la récupération :

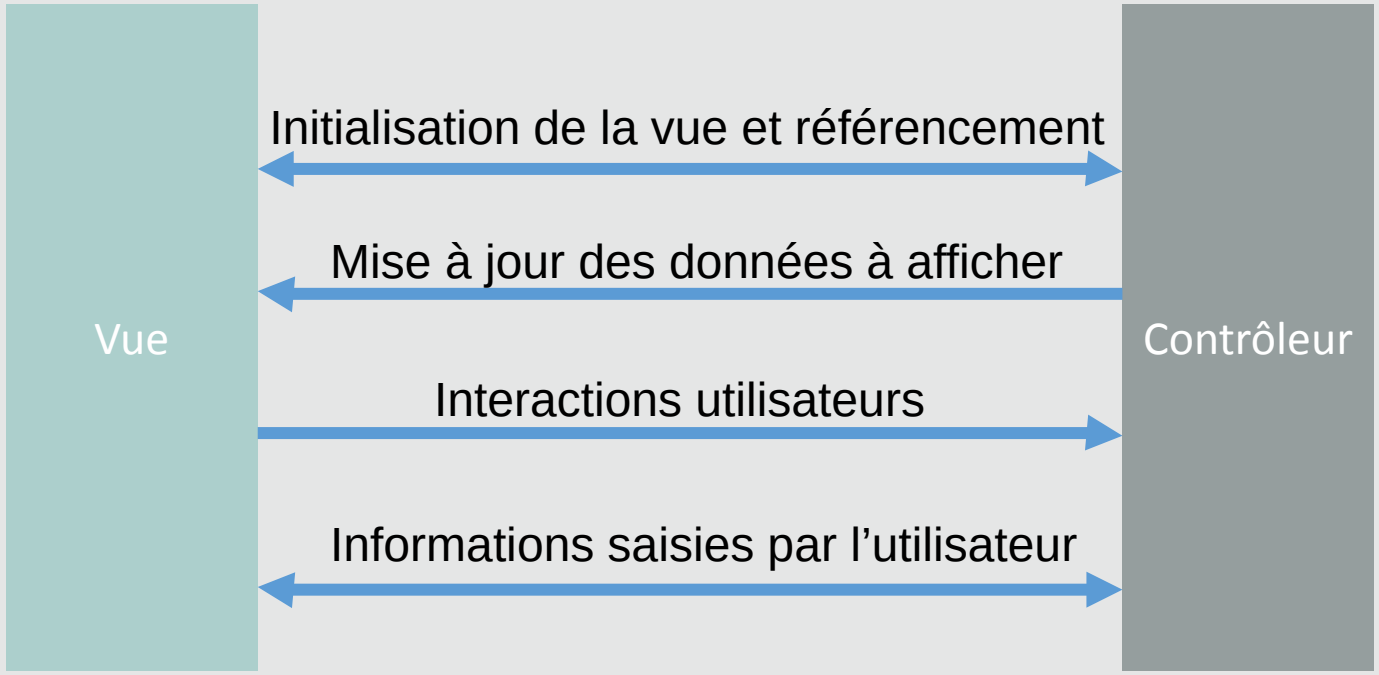
- positionner un nom grâce au paramètre **x:Name** dans le XAML
- récupérer la référence du composant grâce à l'attribut nouvellement créé dans le code behind
- positionner la valeur dans la propriété voulue

Mettre à jour les valeurs d'un composant visuel

```
private void ClicMe_Clicked(object sender, EventArgs e)
{
        this.titre.Text = " Button Clicked !";
}
```

Mettre à jour les valeurs d'un composant visuel

Le cycle standard d'interaction entre la vue et le contrôleur devient donc le suivant :



Les images

La conception de vues en XAML présente certaines limites : toutes les propriétés des composants ne sont pas accessibles et si une donnée demande un pré-traitement ou une transformation avant d'être affichée, alors il est impossible de le faire directement dans la vue.

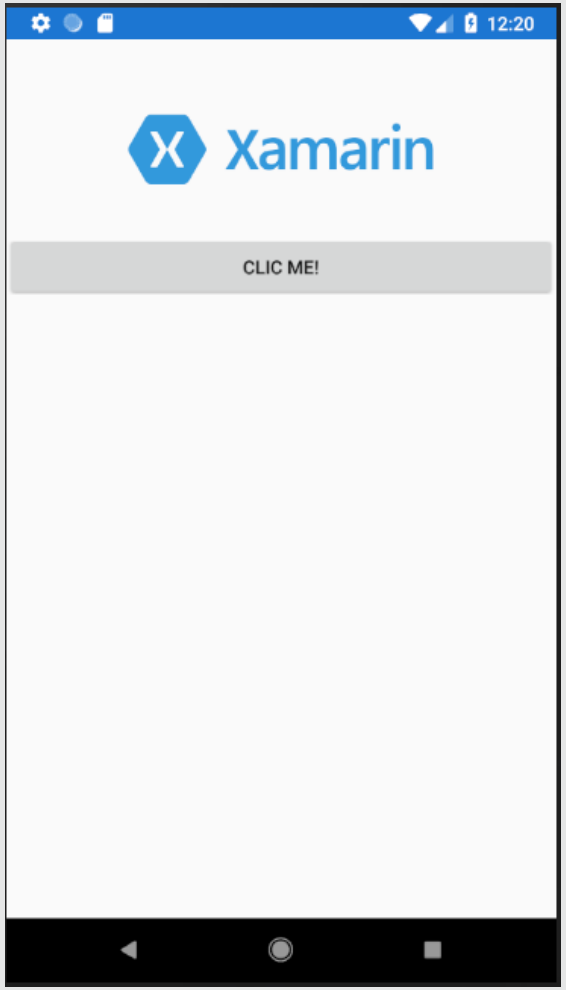
Par exemple, des images en provenance d'Internet ou du projet .NET Standard exigent une préparation avant d'être affichées. Cette opération se déroule dans le contrôleur.

Les images

Pour les images en provenance d'Internet, il suffit de déclarer la source de l'image comme venant d'une URI :

```
public MainPage()
{
            InitializeComponent();
            this.logo.Source = ImageSource.FromUri(
               new Uri("https://upload.wikimedia.org/wikipedia/commons/thumb/f/f2/Xamarin-l
                     ogo.svg/726px-Xamarin-logo.svg.png"));
}
```

Les images



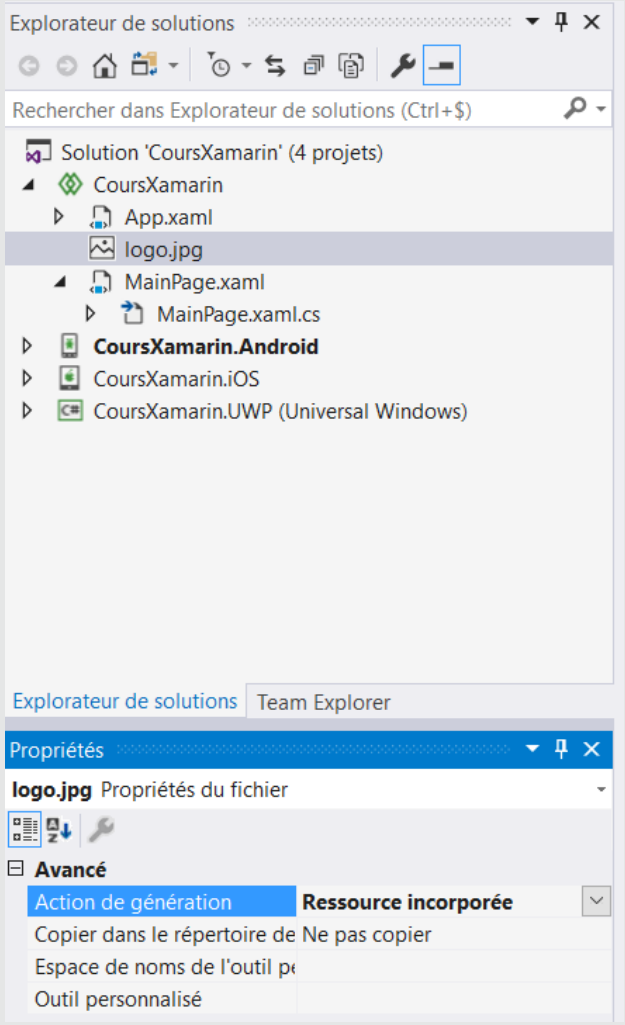
Les images

Pour les images en provenance issues du projet .NET Standard, il faut d'abord les intégrer comme des ressources embarquées du projet

Pour cela il faut les ajouter au projet .NET Standard (Clic droit sur le projet, puis sélectionner **Ajouter**, **Element existant**)

Puis cliquer sur la ressource nouvellement ajoutée et dans le panneau **Propriétés,** sélectionner **Action de génération** : **Ressource Incorporée**

Les images


Explorateur de solutions
Rechercher dans Explorateur de solutions (Ctrl+$)
Solution 'CoursXamarin' (4 projets)
CoursXamarin
App.xaml
logo.jpg
MainPage.xaml
MainPage.xaml.cs
CoursXamarin.Android
CoursXamarin.iOS
CoursXamarin.UWP (Universal Windows)
Explorateur de solutions
Team Explorer
Propriétés
logo.jpg Propriétés du fichier
Avancé
Action de génération
Ressource incorporée
Copier dans le répertoire de
Ne pas copier
Espace de noms de l'outil pe
Outil personnalisé

Les images

Une fois la ressource incorporée, il suffit de déclarer la source de données comme provenant d'une ressource :

```
public MainPage()
{
            InitializeComponent();
            this.logo.Source = ImageSource.FromResource("CoursXamarin.logo.jpg");
}
```

Démonstration

Développer le contrôleur du formulaire de connexion

- Nommer tous les champs de saisie pour pouvoir accéder à leurs propriétés depuis le contrôleur
- S'abonner au clic sur un bouton
- Effectuer des contrôles de surface
- Afficher un message d'erreur si besoin

Développement de contrôleurs Xamarin.Forms

Construire un écran complexe grâce aux layouts

TP