Le développement cross plateforme avec Xamarin

Module 5 – Afficher un formulaire avec Xamarin.Forms

Objectifs

- Savoir manipuler des composants de base comme les boutons, les champs de saisie, etc.
- Savoir intégrer des images dans un projet Xamarin.Forms et les afficher dans une vue

Les textes

Les zones de texte sont les éléments les plus simples et les plus fondamentaux d'une application mobile

Ces zones de texte peuvent être personnalisées comme dans une application native ou dans une page web : police, taille, disposition, couleur, etc.

Les textes

Une zone de texte est simplement déclarée à l'aide de la balise **Label**

```
<Label TextColor="#77d065" FontSize = "20" Text="Ceci est un label"/>
```

Les textes

- Plusieurs attributs permettent d'affiner l'affichage du texte
  - **BackgroundColor** : couleur du fond de la zone de texte
  - **FontAttributes** : spécialisation de la police : **None**, **Bold**, **Italic**
  - **FontFamily** : type de la police : **Lobster-Regular**, **Roboto**, etc.
  - **FontSize** : taille de la police en unité multiplateforme ou **Micro, Small, Medium, Large**
  - **TextColor** : couleur d'écriture de la police
  - **Text** : texte à afficher à l'écran
  - **LineHeight** : hauteur de la ligne de texte par rapport à sa taille de police

Les textes

- Si le texte dépasse la largeur de son conteneur, plusieurs stratégies peuvent être mises en œuvre pour revenir ou non à la ligne :
  - **HeadTruncation** : tronquer le début du texte pour privilégier l'affichage de la fin
  - **TailTruncation** : tronquer la fin du texte pour privilégier l'affichage du début
  - **MiddleTruncation** : tronquer le milieu du texte qui sera remplacé par …
  - **CharacterWrap** : revenir à la ligne, même au milieu d'un mot
  - **WordWrap** : revenir à la ligne, sans couper les mots
  - **NoWrap** : afficher autant de texte que possible, puis tronquer

Les textes

- Les couleurs peuvent être définies à l'aide de leur nom ou de leur code hexadécimal

```
<Label Text="Sea color" BackgroundColor="Aqua" />
<Label Text="RGB" BackgroundColor="#00FF00" />
<Label Text="Alpha plus RGB" BackgroundColor="#CC00FF00" />
<Label Text="Tiny RGB" BackgroundColor="#0F0" />
<Label Text="Tiny Alpha plus RGB" BackgroundColor="#C0F0" />
```

Les textes

Exemple complet de texte

```
<Label
       Text="Lorem ipsum dolor sit amet, consectetur adipiscing elit. In facilisis
nulla eu felis fringilla vulputate. Nullam porta eleifend lacinia."
       TextColor="Red"
       FontAttributes="Bold"
       LineBreakMode="WordWrap"
       LineHeight="1.8"
/>
```

Les zones de saisie

Une application standard est généralement composée de deux éléments fondamentaux : les listes et les formulaires

Les formulaires reposent sur la combinaison de plusieurs modes de saisie permettant à l'utilisateur de renseigner des données de manière ergonomique : saisie de texte, boutons radio, cases à cocher, date et heure, etc.

Les zones de saisie

Les zones de saisie de texte sont très simples à mettre en œuvre et bénéficient de peu d'attributs de paramétrage

```
<Entry Text="Ceci est une zone de saisie" />
```

Les zones de saisie

- Les attributs suivants permettent de paramétrer la zone de saisie
  - **MaxLength** : longueur maximale de la chaîne de caractères
  - **Keyboard** : type de clavier à afficher : **Default**, **Chat**, **Email**, **Numeric**, **Telephone**, **Text**, **Url**
  - **ReturnType** : personnalisation de la touche [Entrée] : **Default**, **None**, **Go**, **Next**, **Search**, **Send**
  - **Placeholder** et **PlaceholderColor** : affichage d'un texte en attendant la saisie utilisateur
  - **IsPassword** : masquage des caractères saisis dans le cas d'un mot de passe
- D'autres attributs, issus du composant Label sont utilisables
  - **TextColor**
  - **BackgroundColor**

Les zones de saisie

Exemple complet d'une zone de saisie

```
<Entry Placeholder="Entrez votre mail" MaxLength="50" Keyboard="Email"
       ReturnType="Send"/>
```

Les switch, slider et stepper

D'autres composants classiques peuvent être facilement mis en place dans un formulaire

Les switch, slider et stepper sont mis en place rapidement mais offrent un vrai plus en terme d'ergonomie à l'utilisateur

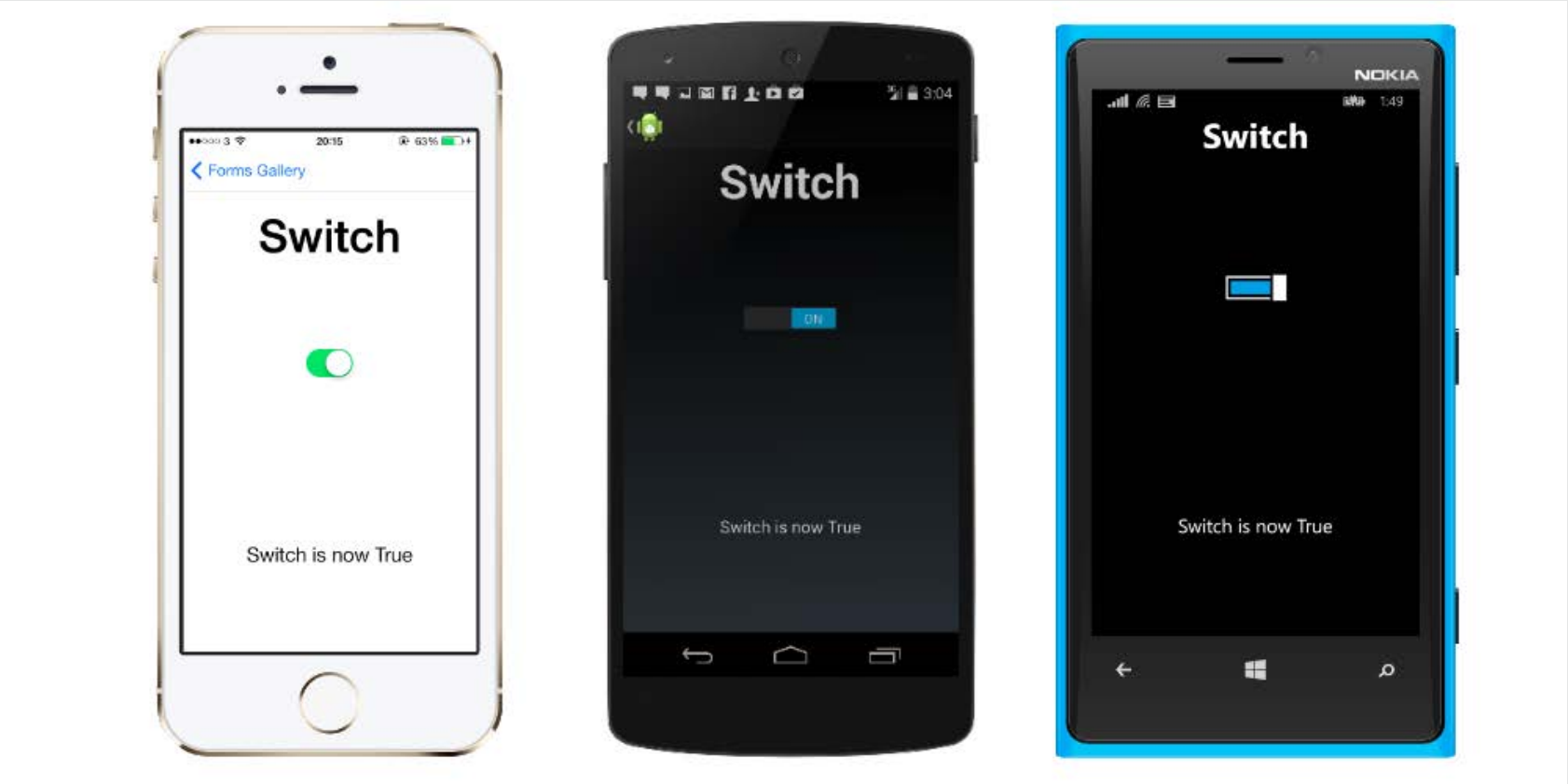
Les switch, slider et stepper

Le switch est l'équivalent d'une case à cocher HTML. Il permet simplement d'indiquer si une option est activée ou non

```
<Switch />
```

Afficher un formulaire avec Xamarin.Forms

Les switch, slider et steppers



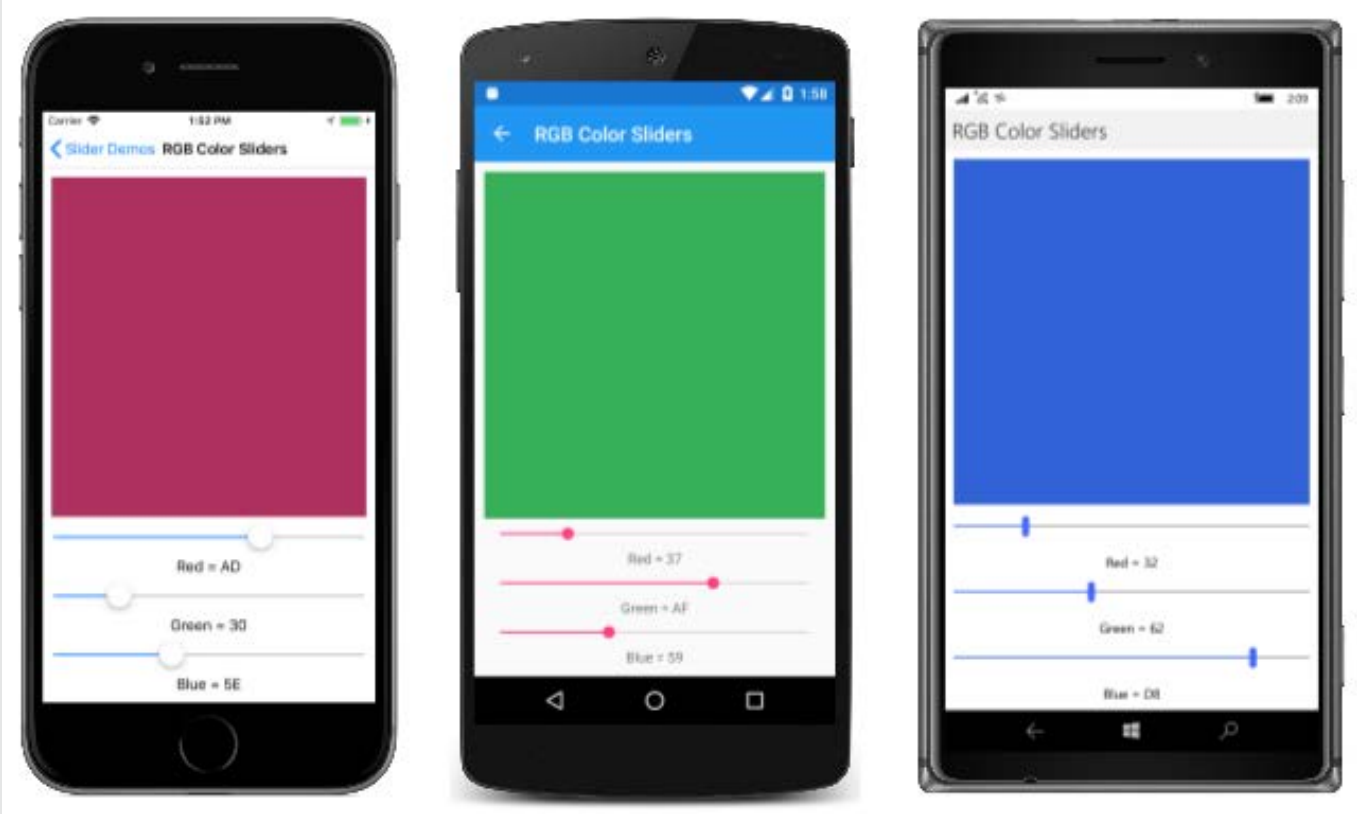
Les switch, slider et stepper

Le slider est un composant simple qui renvoie une valeur comprise entre un minimum et un maximum

```
<Slider Maximum="20" Minimum="10" Value="5" />
```

Afficher un formulaire avec Xamarin.Forms

Les switch, slider et stepper



Les switch, slider et stepper

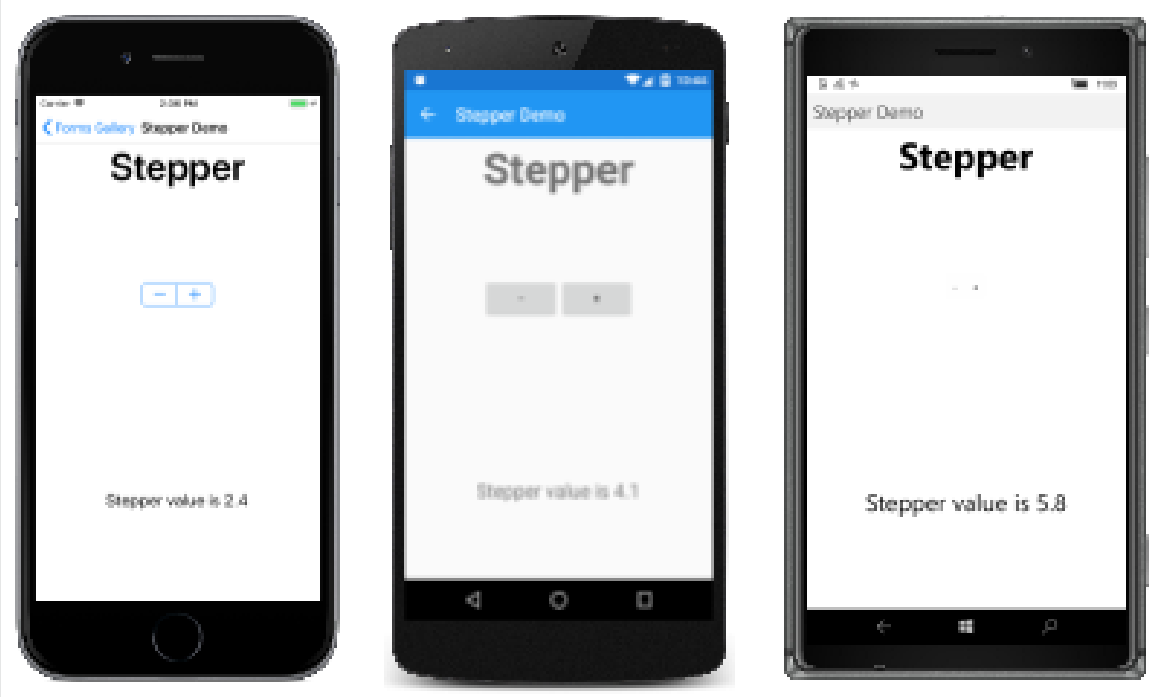
Le stepper est une déclinaison du slider qui fait appel à deux boutons, l'un permettant d'augmenter une valeur numérique et l'autre de la diminuer.

Il prend également en compte un minimum, un maximum, un pas et une valeur.

```
<Stepper Minimum="0"
         Maximum="10"
          Increment="0.1"
/>
```

Afficher un formulaire avec Xamarin.Forms

Les switch, slider et stepper



Les DatePicker et TimePicker

Ces deux composants permettent de rendre plus aisée la saisie des dates ou des heures. Ils sont présents en natif sur toutes les plateformes

Ils sont entièrement paramétrables et servent à de nombreux usages : date de naissance, rendez-vous, alarme, etc.

Les DatePicker et TimePicker

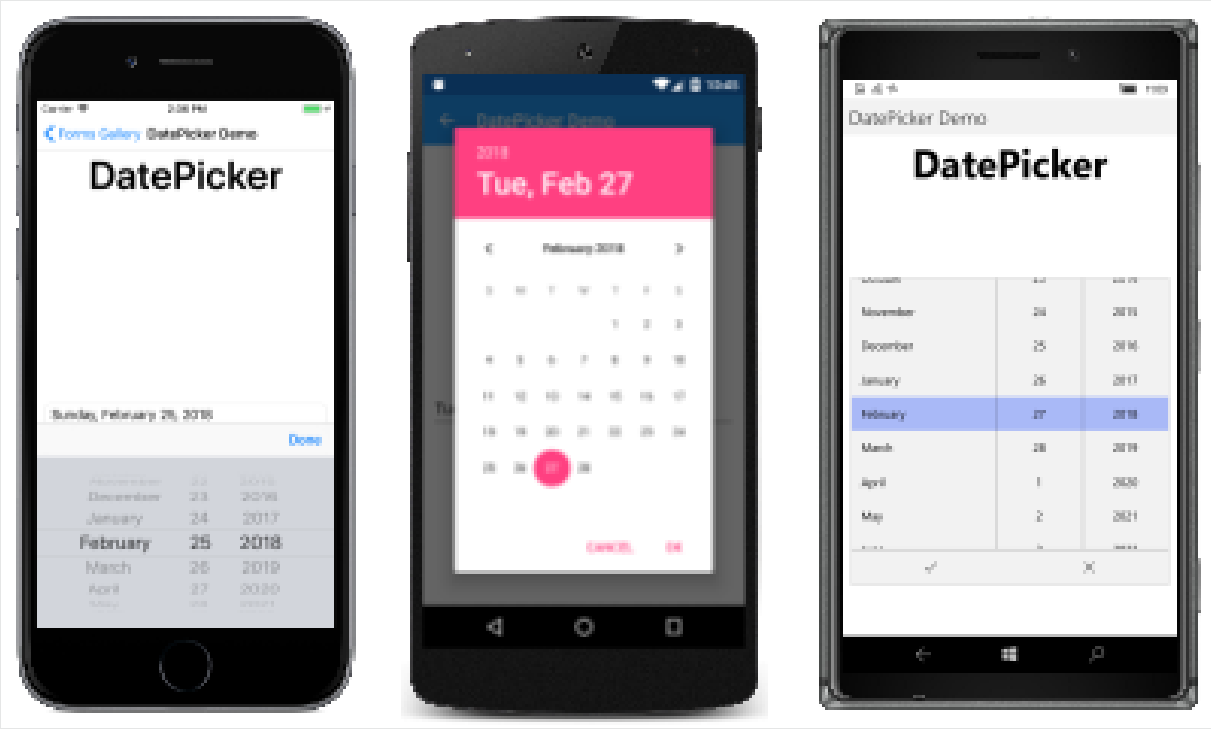
Le DatePicker est un champ de saisie qui peut recevoir un certain nombre d'attributs hérités du Label (tous les attributs de police et de texte) ainsi qu'une date courante, une date minimum et maximum.

Lorsque l'utilisateur clique sur le champ, une fenêtre s'ouvre en surimpression et affiche le sélecteur de date.

```
<DatePicker MinimumDate="01/01/2018" MaximumDate="12/31/2018" Date="06/21/2018" />
```

Afficher un formulaire avec Xamarin.Forms

Les DatePicker et TimePicker



Les DatePicker et TimePicker

Le TimePicker est en tout point similaire au DatePicker.

Néanmoins, il prend seulement en paramètre un attribut Time.

```
<TimePicker Time="10:30" />
```

Afficher un formulaire avec Xamarin.Forms

Les DatePicker et TimePicker

Les images

L'utilisation d'images rend une application plus vivante, moins austère et aère l'interface. Leur intégration est donc indispensable dans une application moderne.

Néanmoins, l'intégration d'images a des contraintes : téléchargement depuis Internet, dimensions et formats hétérogènes, difficultés à les intégrer dans une interface responsive, etc.

Les images

Xamarin.Forms met à disposition du développeur une balise **Image** qui permet d'intégrer une zone dans laquelle sera affichée une image.

Ce composant répond aux mêmes contraintes que les autres en matière de positionnement et de taille.

```
<Image Source="logo.jpg" Aspect="AspectFit" />
```

Les images

- Propriétés de gestion des images
  - **Source** : indique où trouver l'image et avec quelle stratégie
    - Depuis un fichier présent dans chaque plateforme
    - Depuis une URL accessible depuis Internet
    - Depuis une ressource intégrée au projet .NET Standard
  - **Aspect** : indique comment intégrer l'image dans la zone d'affichage
    - **Fill** : applique l'image sur toute la zone d'affichage, quitte à déformer l'image
    - **AspectFill** : rogne l'image tout en préservant son ratio : toute l'image peut ne pas être affichée mais elle ne sera pas déformée
    - **AspectFit** : ne rogne pas l'image quitte à ce que le ratio soit modifié : toute l'image sera affichée mais toute la zone d'affichage ne sera pas utilisée

Les images

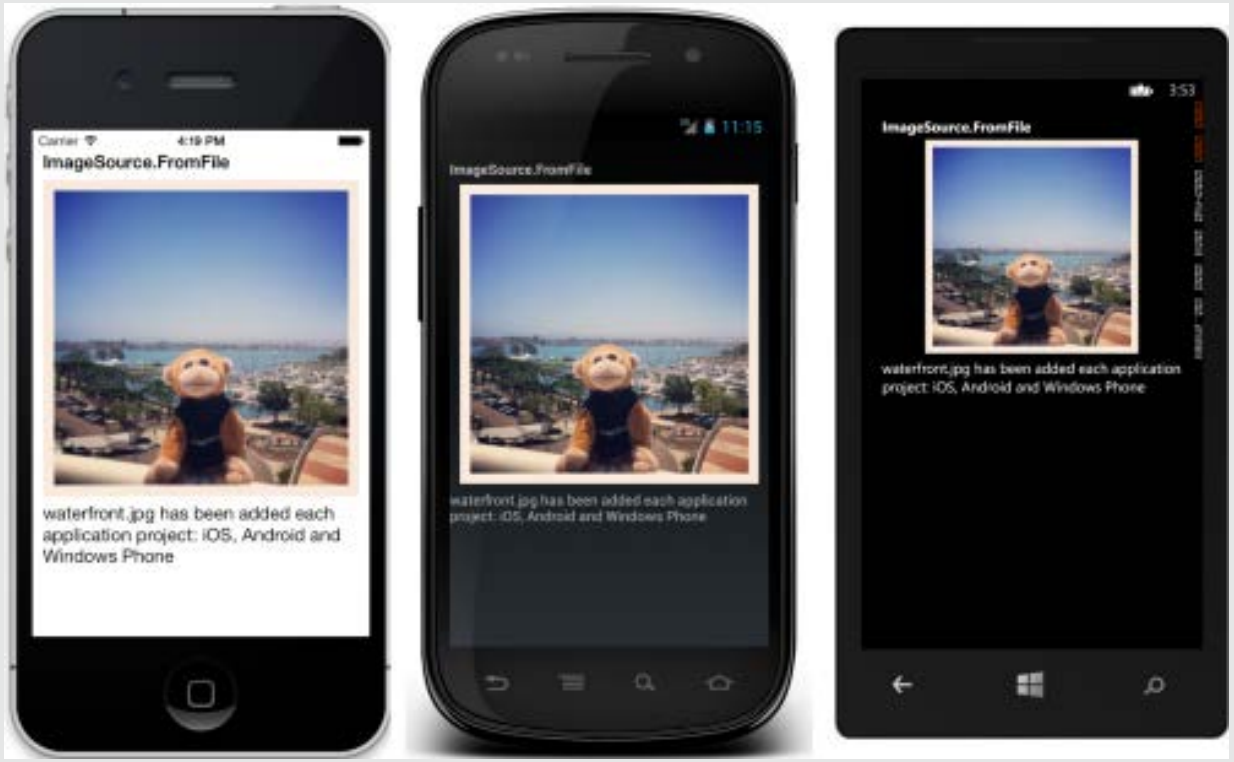
- Les images locales doivent être intégrées manuellement dans chaque projet natif
  - **Android** : répertoire **Resources/drawable**
  - **iOS** : catalogue d'Asset
  - **Windows** : racine du projet en plaçant la propriété **Build Action** à **Content**

- Méthode la plus canonique et la plus efficace car elle permet pour chaque système de décliner l'image dans plusieurs résolutions pour plus de performances et de netteté

Les images

- Pour les images distantes et les images embarquées dans le projet .NET Standard, il est nécessaire d'utiliser le contrôleur C#
- Cette manipulation sera décrite dans le module suivant

Les images


ImageSource.FromFile
waterfront.jpg has been added each application project: iOS, Android and Windows Phone
ImageSource.FromFile
ImageSource.FromFile
waterfront.jpg has been added each application project: iOS, Android and Windows Phone

Les boutons

Les boutons sont, comme les zones de saisie, au cœur du système de formulaire et permettent à l'utilisateur de le valider

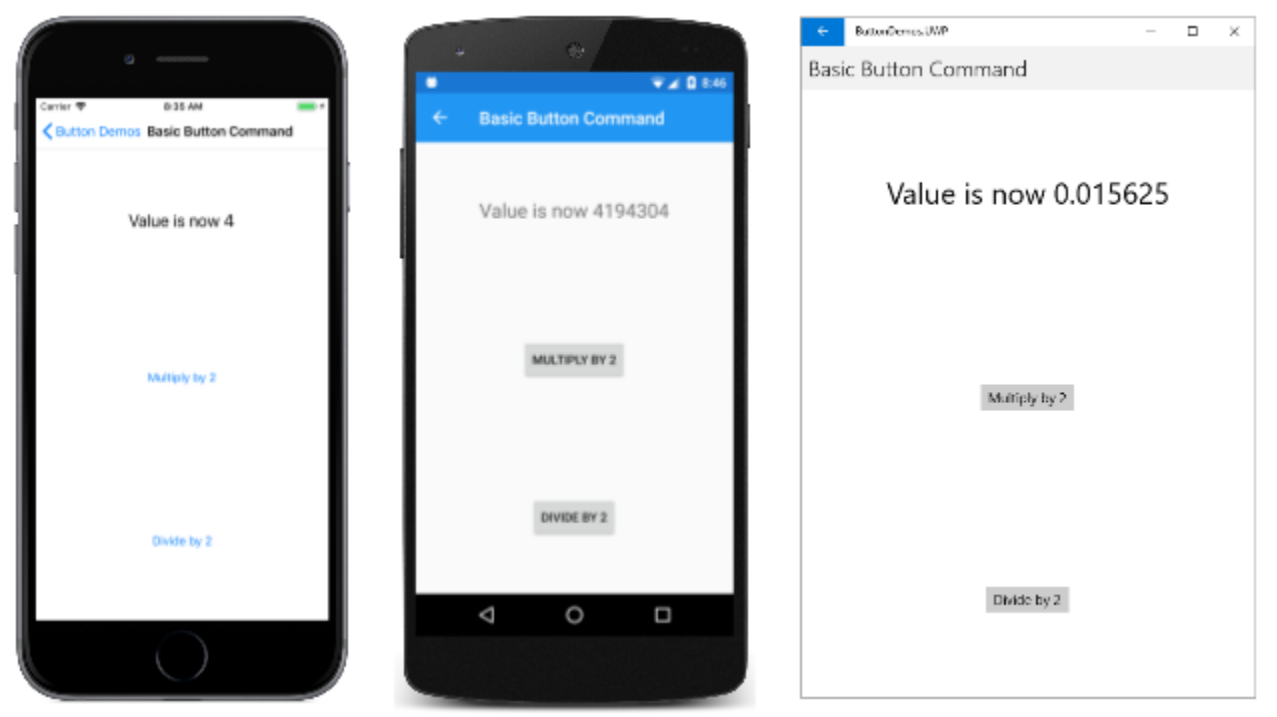
```
<Button Text="Click Me !" />
```

Les boutons

- Les boutons ont, comme tous composants visuels, des attributs qui permettent de les personnaliser
  - **Text** : texte à afficher sur le bouton
  - **IsEnabled** : activation ou non du bouton (réponse au clic)
  - **TextColor** : couleur du texte du bouton
  - **BackgroundColor** : couleur de fond du bouton
  - **BorderColor** : couleur de la bordure du bouton

Afficher un formulaire avec Xamarin.Forms

Les boutons



Démonstration

Afficher un formulaire avec Xamarin.Forms

Créer un formulaire de connexion

- Utiliser les composants visuels **Label**, **Entry**, **DatePicker**, **Button** et **Switch**
- Mettre en forme le formulaire grâce aux **StackLayout**

Afficher un formulaire avec Xamarin.Forms

Créer une page d'ajout de note d'humeur

TP