Xamarin Forms
XAML - Parte 1

June 11, 2015

1 Objetivos

- Tener entendimiento y práctica en el uso de XAML con Xamarin.Forms

2 Requerimientos

- Proyecto creado en Xamarin.Forms
- Familiriadad con XML, entendimiento de la declaración de namespace, de los términos *element*, *tag* y *attribute*.

3 Creación de una página

XAML puede jugar un papel en una aplicación Xamarin.Forms de varias maneras, pero sin duda, el más común es el de definir el contenido visual de una página entera, que normalmente es una clase derivada de ContentPage.
En Visual Studio, haga clic en el proyecto (Portable) y seleccione $Add > NewItem$. En la ventana $AddNewItem$, seleccione $VisualC\# > Code$, y $FormsXamlPage$ de la lista.
Dos nuevos archivos se crean en el proyecto El primero es un archivo XAML con la extensión *.xaml*, el segundo archivo se muestra debajo de ella; se trata de un archivo de código C# con la extensión *.xaml.cs*. Los nombres de estos dos archivos revelan que están íntimamente relacionados. El archivo de C# se conoce como el archivo de código subyacente del archivo XAML.

4 Perzonalindo el Contenido de una Página

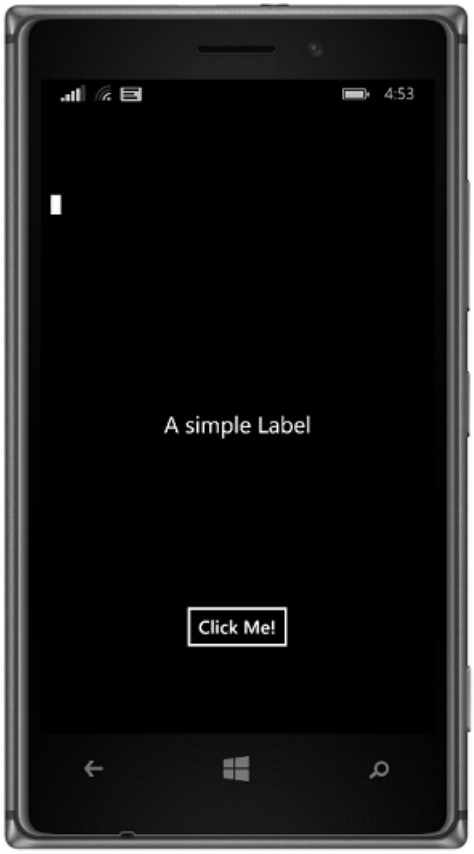
Una aplicación de una sola página en Xamarin.Forms generalmente contiene una clase derivada de *ContentPage*. La propiedad *content* de esta clase generalmente se establece en una sola vista o un diseño con vistas a los hijos. En XAML, una vista (por ejemplo, una etiqueta) se puede ajustar de forma implícita a la propiedad de *content* de la página al ser colocado entre las etiquetas de inicio y fin de *ContentPage*

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="XamlSamples.HelloXamlPage"
             Title="Hello XAML Page"
             Padding="10, 40, 10, 10">

  <Label Text="Hello, XAML!"
         VerticalOptions="Start"
         XAlign="Center"
         Rotation="-15"
         IsVisible="true"
         FontSize="Large"
         FontAttributes="Bold"
         TextColor="Aqua" />

</ContentPage>
```

5 XAML y la Interacción con el código

Crear la siguiente vista con los siguientes elementos

- StackLayout
- Slider
- Label
- Button



Sin embargo, es probable que esta vista sea deficiente en funcionalidad. Es muy probable que se desee manipular el control deslizante para hacer que la etiqueta muestre el valor actual y el botón es, probablemente, con la intención de hacer algo dentro del programa.

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="XamlSamples.XamlPlusCodePage"
             Title="XAML + Code Page">
  <StackLayout>
    <Slider VerticalOptions="CenterAndExpand"
            ValueChanged="OnSliderValueChanged" />

    <Label Text="A simple Label"
           Font="Large"
           HorizontalOptions="Center"
           VerticalOptions="CenterAndExpand" />

    <Button Text="Click Me!"
            HorizontalOptions="Center"
            VerticalOptions="CenterAndExpand"
            Clicked="OnButtonClicked" />
  </StackLayout>
</ContentPage>
```

Tenga en cuenta que la asignación de un controlador para un evento tiene la misma sintaxis que la asignación de un valor a una propiedad, y para agregarle funcionalidad dentro de código .cs es necesario definir los métodos subyacentes

```
void OnSliderValueChanged(object sender, ValueChangedEventArgs args
    )
{

}

void OnButtonClicked(object sender, EventArgs args)
{

}
```

Si el controlador para el evento ValueChanged del deslizador va a utilizar la etiqueta para mostrar el valor actual, el controlador tiene que hacer referencia a ese objeto desde el código. La etiqueta necesita un nombre, que se especifica con los atributo x: Name.

```
<Label x:Name="valueLabel"
       Text="A simple Label"
       Font="Large"
       HorizontalOptions="Center"
       VerticalOptions="CenterAndExpand" />
```

El prefijo x del atributo $x : Name$ indica que este atributo es intrínseca a XAML.
El nombre que asigne al atributo $x : Name$ tiene las mismas reglas que C# para los nombres de variables. Por ejemplo, se debe comenzar con una letra o un guión y no contienen espacios incrustados.
Ahora el controlador de eventos ValueChanged puede establecer la etiqueta para mostrar el nuevo valor de Slider. El nuevo valor se encuentra disponible en los argumentos del evento:

```
void OnSliderValueChanged(object sender, ValueChangedEventArgs args
    )
{
    valueLabel.Text = args.NewValue.ToString("F3");
}
```

O bien, el controlador podría obtener el objeto deslizante que está generando este evento desde el argumento de emisor y obtener la propiedad Value:

```
void OnSliderValueChanged(object sender, ValueChangedEventArgs args
    )
{
    valueLabel.Text = ((Slider)sender).Value.ToString("F3");
}
```

Finalmente agrege una funcionalidad al boton.