XAMARIN

**SEC-1: GETTING STARTED**

**CREATING A XAMARIN FORMS APP**

Creamos una aplicación mobile de xamarin en “Crear proyecto”. Luego cuando tengamos el proyecto creado hacemos click derecho en la carpeta del nombre que le dimos al proyecto y elegimos **Administrar paquetes NuGet**. Actualizamos todos los paquetes.

Luego elegimos la carpeta con el tipo de app que queremos (Android, IOS, etc.), y presionamos **ctrl + F5** para descargar el emulador.

**ACCESING ELEMENTS**

**GreetPage.xaml.cs:**

public partial class GreetPage : ContentPage

{

public GreetPage()

{

InitializeComponent();

slider.Value = 0.5;

}

private void Slider\_ValueChanged(object sender, Xamarin.Forms.ValueChangedEventArgs e)

{

label.Text = String.Format("Value is {0:F2}", e.NewValue);

}

}

**GreetPage.xaml:**

<ContentPage

xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="FirstApplication.GreetPage"

>

<StackLayout HorizontalOptions="Center" VerticalOptions="Center">

<Label Text="Hello World" x:Name="label"/>

<Slider ValueChanged="Slider\_ValueChanged" x:Name="slider"/>

</StackLayout>

</ContentPage>

**DATA BINDING**

Utilizamos los **XAML markup extensions**:

<Label Text="{Binding

Source={x:Reference slider},

Path=Value,

StringFormat='Value is {0:F2}'}"

x:Name="label"

/>

**BINDING CONTEXT**

Resumimos el **Source y el Path**.

<ContentPage

xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="FirstApplication.GreetPage"

>

<StackLayout

BindingContext="{x:Reference slider}" HorizontalOptions="Center" VerticalOptions="Center"

>

<BoxView

Color="Green"

Opacity="{Binding Value}"/>

<Label

Text="{Binding Value,

StringFormat='Value is {0:F2}' }"

Opacity="{Binding Value}"

/>

<Slider x:Name="slider"/>

</StackLayout>

</ContentPage>

**DEALING with DEVICE DIFFERENCES**

public GreetPage()

{

InitializeComponent();

slider.Value = 0.5;

//if(Device.OS == TargetPlatform.iOS)

//{

// Padding = new Thickness(0, 20, 0, 0);

//}

Padding = Device.OnPlatform(

iOS: new Thickness(0, 20, 0, 0),

Android: new Thickness(10, 20, 0, 0),

WinPhone: new Thickness(30, 20, 0, 0)

);

}

**PROPERTY ELEMENT SYNTAX**

Utilizamos esta sintaxis cuando estamos lidiando con objetos complejos que no pueden ser representados usando simples strings.

**LAYOUTS**

**STACK LAYOUT IN XAML**

<ContentPage Padding="0,20,0,0"

xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="FirstApplication.StackPage">

<StackLayout

Orientation="Horizontal"

Padding="40"

Spacing="20"

VerticalOptions="Center"

HorizontalOptions="Center">

<StackLayout>

<Image Source="http://placehold.it/100x100"/>

<Label Text="Label 1" />

</StackLayout>

<Label Text="Label 2" />

<Label Text="Label 3" />

</StackLayout>

</ContentPage>

**STACK LAYOUT IN CODE**

var layout = new StackLayout

{

Spacing = 40,

Padding = new Thickness(0, 20, 0, 0),

Orientation = StackOrientation.Horizontal

};

layout.Children.Add(new Label { Text = "Label 1" });

Content = layout;

**GRID IN XAML**

<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="FirstApplication.GridPage"

Padding="0,20,0,0">

<Grid BackgroundColor="Yellow"

RowSpacing="40"

ColumnSpacing="20">

<Grid.RowDefinitions>

<RowDefinition Height="100" />

<RowDefinition Height="2\*" />

<RowDefinition Height="\*" />

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto" />

<ColumnDefinition Width="2\*" />

<ColumnDefinition Width="\*" />

</Grid.ColumnDefinitions>

<Label Grid.Row="0" Grid.Column="0" Text="Label 1" BackgroundColor="Silver" />

<Label Grid.Row="0" Grid.Column="1" Text="Label 2" BackgroundColor="Silver" />

<Label Grid.Row="1" Grid.Column="0" Text="Label 3" BackgroundColor="Silver" />

<Label Grid.Row="1" Grid.Column="1" Text="Label 4" BackgroundColor="Silver" />

<Label Grid.Row="2" Grid.ColumnSpan="3" Text="Column Span" BackgroundColor="Silver" />

<Label Grid.Column="2" Grid.RowSpan="2" Text="Row Span" BackgroundColor="Silver" />

</Grid>

</ContentPage>

**GRID IN CODE**

public GridPage()

{

InitializeComponent();

var grid = new Grid

{

RowSpacing = 20,

ColumnSpacing = 40

};

var label = new Label { Text = "Label 1" };

grid.Children.Add(label, 0, 0);//el primer 0 es la columna, el segundo la fila

Grid.SetRowSpan(label, 2);

Grid.SetColumnSpan(label, 2);

Grid.SetRow(label, 0);

Grid.SetColumn(label, 0);

grid.RowDefinitions.Add(new RowDefinition

{

Height = new GridLength(100, GridUnitType.Absolute)

});

grid.RowDefinitions.Add(new RowDefinition

{

Height = new GridLength(2, GridUnitType.Star)

});

grid.RowDefinitions.Add(new RowDefinition

{

Height = new GridLength(1, GridUnitType.Auto)

});

}

**ABSOLUTE LAYOUT IN XAML**

AbsoluteLayout.LayoutFlags tiene como porpiedades:

* None.
* All.
* WidthProportional.
* HeightProportional.
* XProportional.
* YProportional.
* PositionProportional.
* SizeProportional.

<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="FirstApplication.AbsolutePage"

Padding="0,20,0,0">

<AbsoluteLayout>

<BoxView Color="Aqua"

AbsoluteLayout.LayoutBounds="0, 0, 1, 1"

AbsoluteLayout.LayoutFlags="All"/>

<BoxView Color="White"

AbsoluteLayout.LayoutBounds="0.5, 0.1, 100, 100"

AbsoluteLayout.LayoutFlags="PositionProportional"/>

<Button Text="Get Started" BackgroundColor="Silver" TextColor="White"

AbsoluteLayout.LayoutBounds="0, 1, 1, 50"

AbsoluteLayout.LayoutFlags="PositionProportional, WidthProportional"/>

</AbsoluteLayout>

</ContentPage>

**ABSOLUTE LAYOUT IN CODE**

public AbsolutePage()

{

InitializeComponent();

var layout = new AbsoluteLayout();

Content = layout;

var aquaBox = new BoxView { Color = Color.Aqua };

layout.Children.Add(aquaBox

, new Rectangle(0, 0, 1, 1)

, AbsoluteLayoutFlags.All);

AbsoluteLayout.SetLayoutBounds(aquaBox, new Rectangle(0, 0, 1, 1));

AbsoluteLayout.SetLayoutFlags(aquaBox, AbsoluteLayoutFlags.HeightProportional);

}

**RELATIVE LAYOUT IN XAML**

Tiene las siguientes propiedades:

* XConstraint.
* YConstraint.
* WidthConstraint.
* HeightConstraint.
* BoundsConstraint.

<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="FirstApplication.RelativePage">

<RelativeLayout>

<BoxView Color="Aqua" x:Name="banner"

RelativeLayout.WidthConstraint="{ConstraintExpression

Type=RelativeToParent,

Property=Width,

Factor=1}"

RelativeLayout.BoundsConstraint="{ConstraintExpression

Type=RelativeToParent,

Property=Height,

Factor=0.3}"/>

<BoxView Color="Silver"

RelativeLayout.YConstraint="{ConstraintExpression

Type=RelativeToView,

ElementName=banner,

Property=Height,

Factor=1,

Constant=20}"/>

</RelativeLayout>

</ContentPage>

**RELATIVE LAYOUT IN CODE**

public RelativePage()

{

InitializeComponent();

var layout = new RelativeLayout;

Content = layout;

var aquaBox = new BoxView { Color = Color.Aqua };

layout.Children.Add(aquaBox,

widthConstraint: Constraint.RelativeToParent(parent => parent.Width),

heightConstraint: Constraint.RelativeToParent(parent => parent.Height \* 0.3));

var silverBox = new BoxView { Color = Color.Silver };

layout.Children.Add(silverBox,

yConstraint: Constraint.RelativeToView(aquaBox,(RelativeLayout, element) => element.Height + 20));

}

**IMAGES**

**DOWNLOAD IMAGES**

public ImagePage()

{

InitializeComponent();

var imageSource = new UriImageSource { Uri = new Uri("https://miro.medium.com/max/1400/1\*BP3XeAE4UmoRcv6dh\_Z4iA.jpeg") } ;

imageSource.CachingEnabled = false;

image.Source = imageSource;

}

**XAML:**

<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="FirstApplication.ImagePage">

<Image x:Name="image"/>

</ContentPage>

**ASPECTS**

<Image x:Name="image" Aspect="AspectFill"/>

**ACTIVITY INDICATOR**

<AbsoluteLayout>

<ActivityIndicator

IsRunning="{Binding Source={x:Reference image},Path=IsLoading}"

Color="White"

AbsoluteLayout.LayoutBounds="0.5, 0.5, 100, 100"

AbsoluteLayout.LayoutFlags="PositionProportional" />

<Image x:Name="image" Aspect="AspectFill"

AbsoluteLayout.LayoutBounds="0, 0, 1, 1"

AbsoluteLayout.LayoutFlags="All"/>

</AbsoluteLayout>

**EMBEDDED IMAGE**

En la foto que queramos aplicar el **embedded image** debemos configurarlo desde **propiedades/acción de compilación** y nos debería dar un ID. Si no nos da el ID lo escribimos de la siguiente manera: **nombreDelProyecto.carpetaDondeEstaLaImagen.nombreDeLaImagen.extensiónDeLaImagen**.

public ImagePage()

{

InitializeComponent();

image.Source = ImageSource.FromResource("FirstApplication.Images.background.jpg");

}

**XAML:**

<Image x:Name="image" Aspect="AspectFill"/>

**EMBEDDED IMAGES in XAML**

**EmbeddedImage file:**

using System;

using System.Collections.Generic;

using System.Text;

using Xamarin.Forms;

using Xamarin.Forms.Xaml;

namespace FirstApplication.MarkupExtensions

{

[ContentProperty("ResourceId")]

public class EmbeddedImage : IMarkupExtension

{

public string ResourceId { get; set; }

public object ProvideValue(IServiceProvider serviceProvider)

{

if (String.IsNullOrWhiteSpace(ResourceId))

{

return null;

} else

{

return ImageSource.FromResource(ResourceId);

}

}

}

}

**ImagePage.xaml.cs:**

public ImagePage()

{

InitializeComponent();

image.Source = ImageSource.FromResource("FirstApplication.Images.background.jpg");

}

**ImagePage.XAML:**

<Image Source="{local:EmbeddedImage FirstApplication.Images.background.jpg}" x:Name="image" Aspect="AspectFill"/>