<?xml version="1.0" encoding="UTF-8"?>

xmlns:shortName is namespace

mc:Ignorable namespace by xaml processor: default design time experience + ignored at runtime

￼￼￼￼￼￼

[AbsoluteLayout](https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.absolutelayout) positions/sizes childes proportional to its own size or by absolute values.

<https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/layouts/absolute-layout#Proportional_Values>

<Label Text="bottom center” AbsoluteLayout.LayoutBounds=".5,1,.5,.1" AbsoluteLayout.LayoutFlags="All” />

var label = new Label {Text = "bottom center"};

AbsoluteLayout.SetLayoutBounds(label, new Rectangle (.5,1,.5,.1)); //x, y,w,h

AbsoluteLayout.SetLayoutFlags(label, AbsoluteLayoutFlags.All); // proportional

[StackLayout](https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.stacklayout) organizes views in a one-dimensional line ("stack"), sized using layout options.

<StackLayout Spacing="10" x:Name="layout">

<Button Text="StackLayout" VerticalOptions="Start" HorizontalOptions="FillAndExpand" />

<BoxView Color="Yellow" VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand" />

<BoxView HeightRequest="75" Color="Blue" VerticalOptions="End" HorizontalOptions="FillAndExpand" />

</StackLayout>

var layout = new StackLayout ();

var button = new Button { Text = "StackLayout", VerticalOptions = LayoutOptions.Start, HorizontalOptions = LayoutOptions.FillAndExpand };

var yellowBox = new BoxView { Color = Color.Yellow, VerticalOptions = LayoutOptions.FillAndExpand, HorizontalOptions = LayoutOptions.FillAndExpand , HeightRequest = 75 };

layout.Children.Add(button); layout.Children.Add(yellowBox); layout.Spacing = 10;

Content = layout;

RelativeLayout:relative layout/sibling views: NO moving anchor, BUT position outside of bounds.

<BoxView Color="Green" WidthRequest="50" HeightRequest="50"

RelativeLayout.XConstraint ="{ConstraintExpression Type=RelativeToParent, Property=Width, Factor=0.5, Constant=-100}"

/>

layout.Children.Add(box,

Constraint.RelativeToParent((parent) =>{ return (.5 \* parent.Width) - 100;}),

Constraint.Constant(50));

FlexLayout =arrange in a stack + wrapping to fit .. options[orientation,alignment,adapting]

<FlexLayout Direction="Column" AlignItems="Center" JustifyContent="SpaceEvenly">

<Label Text="FlexLayout in Action" FontSize="Large" />

<Image Source="{local:ImageResource FlexLayoutDemos.Images.SeatedMonkey.jpg}" />

<Button Text="Do-Nothing Button" />

</FlexLayout>

Grid = rows and columns[proportional/absolute] sizes= not HTML tables=No[row, column ,cell]

ContentView=a single child element = create custom reusable controls.(inherits from TemplatedView)

public static readonly BindableProperty CardTitleProperty = BindableProperty.Create(

"CardTitle", // the name of the bindable property

typeof(string), // the bindable property type

typeof(CardView), // the parent object type

string.Empty); } // the default value for the property

public string CardTitle{

get => (string)GetValue(CardView.CardTitleProperty);

set => SetValue(CardView.CardTitleProperty, value);}

<ContentView x:Name="this" x:Class="CardViewDemo.Controls.CardView">

<Frame BindingContext="{x:Reference this}" BackgroundColor="{Binding CardColor}" BorderColor="{Binding BorderColor}"

<Grid>

<Frame BorderColor="{Binding BorderColor, FallbackValue='Black'}" BackgroundColor="{Binding IconBackgroundColor, FallbackValue='Grey'}" >

<Image Source="{Binding IconImageSource}" />

</Frame>

<Label Text="{Binding CardTitle, FallbackValue='Card Title'}"/>

<BoxView BackgroundColor="{Binding BorderColor, FallbackValue='Black'}" />

<Label Text="{Binding CardDescription, FallbackValue='Card description text.'}" />

</Grid>

</Frame>

</ContentView

Forms layout = wrap a view with a border [ color, shadow..] ,create more complex UI.

ScrollView: contains layouts => scroll offscreen/move to the visible portion if keyboard is showing.

<ContentPage.Content>

<ScrollView> <StackLayout>

<BoxView BackgroundColor="Red" HeightRequest="600" WidthRequest="150" />

<Entry />

</StackLayout> </ScrollView>

</ContentPage.Content>

StackLayout stackLayout = new StackLayout();

stackLayout.Children.Add(new BoxView { BackgroundColor = Color.Red, HeightRequest = 600, WidthRequest = 150 });

stackLayout.Children.Add(new Entry());

ScrollView scrollView = new ScrollView();

scrollView.Content = stackLayout;

Content = scrollView;



update .. restart .. .android=startup ... delete projects/later addthem ...









￼

￼￼

￼￼

￼

<d:ListView.ItemsSource>

<x:Array Type="{x:Type models:Monkey}">

<models:Monkey Name="Baboon" Location="Africa and Asia"/>

<models:Monkey Name="Capuchin Monkey" Location="Central and South America"/>

<models:Monkey Name="Blue Monkey" Location="Central and East Africa"/>

</x:Array>

</d:ListView.ItemsSource>

<ListView.ItemTemplate>

<DataTemplate x:DataType="models:Monkey">

<ImageCell Text="{Binding Name}"

Detail="{Binding Location}"

ImageSource="{Binding Image}"

d:ImageSource="icon.png"/>

</DataTemplate>

</ListView.ItemTemplate>

-->

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

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

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

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

mc:Ignorable="d"

x:Class="App2.Views.NewItemPage"

Title="New Item">

<ContentPage.ToolbarItems>

<ToolbarItem Text="Cancel" Clicked="Cancel\_Clicked" />

<ToolbarItem Text="Save" Clicked="Save\_Clicked" />

</ContentPage.ToolbarItems>

<ContentPage.Content>

<StackLayout Spacing="20" Padding="15">

<Label Text="Text" FontSize="Medium" />

<Entry Text="{Binding Item.Text}" d:Text="Item name" FontSize="Small" />

<Label Text="Description" FontSize="Medium" />

<Editor Text="{Binding Item.Description}" d:Text="Item description" FontSize="Small" Margin="0" />

</StackLayout>

</ContentPage.Content>

</xa:ContentPage>