using System.ComponentModel;

using System.Runtime.CompilerServices;

namespace ExampleBinding

{

//AmountViewModel class inherits INotifyPropertyChanged interface

public class AmountViewModel : INotifyPropertyChanged

{

private double \_numberOne;

private double \_numberTwo;

private double \_result;

//This property manages and reports changes to the value.

public double NumberOne

{

get => \_numberOne;

set

{

if (value != \_numberOne)

{

\_numberOne = value;

OnPropertyChanged();

OnPropertyChanged(nameof(Result));

}

}

}

//This property manages and reports changes to the value.

public double NumberTwo

{

get => \_numberTwo;

set

{

if (value != \_numberTwo)

{

\_numberTwo = value;

OnPropertyChanged();

OnPropertyChanged(nameof(Result));

}

}

}

//This property is read-only and computed dynamically.

public double Result => \_numberOne + \_numberTwo;

private void AddNumbers()

{

Result = \_numberOne + \_numberTwo;

}

//This function allows you to implement the INotifyPropertyChanged interface

//allows the view to update automatically when the ViewModel properties change

public event PropertyChangedEventHandler PropertyChanged;

protected virtual void OnPropertyChanged([CallerMemberName] string propertyName = null)

{

PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));

}

}

}

namespace ExampleBinding;

public partial class BindingContextPage : ContentPage

{

public BindingContextPage()

{

InitializeComponent();

BindingContext = new AmountViewModel();

}

}

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

<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

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

x:Class="ExampleBinding.BindingContextPage"

Title="BindingContextPage">

<StackLayout Padding="20">

<Entry Placeholder="Enter number 1"

Keyboard="Numeric"

Text="{Binding NumberOne}" />

<Entry Placeholder="Enter number 2"

Keyboard="Numeric"

Text="{Binding NumberTwo}" />

<Label Text="{Binding Result, StringFormat='Result: {0:N2}'}"

FontSize="24"

HorizontalOptions="Center" />

</StackLayout>

</ContentPage>

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

<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

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

x:Class="ExampleBinding.MainPage">

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

<Label Text="{Binding CurrentTime}"

VerticalTextAlignment="Center"

HorizontalTextAlignment="Center"

FontSize="45" />

</StackLayout>

</ContentPage>

namespace ExampleBinding;

public partial class MainPage : ContentPage

{

public MainPage()

{

InitializeComponent();

BindingContext = new TimeViewModel();

}

}

using System.ComponentModel;

using System.Timers;

namespace ExampleBinding;

public class TimeViewModel : INotifyPropertyChanged

{

private readonly Timer \_timer;

private string \_currentTime;

public string CurrentTime

{

get => \_currentTime;

set

{

if (\_currentTime != value)

{

\_currentTime = value;

OnPropertyChanged(nameof(CurrentTime));

}

}

}

public TimeViewModel()

{

\_timer = new Timer(1000);

\_timer.Elapsed += (s, e) =>

{

MainThread.BeginInvokeOnMainThread(() =>

{

CurrentTime = DateTime.Now.ToString("HH:mm:ss");

});

};

\_timer.Start();

}

public event PropertyChangedEventHandler PropertyChanged;

protected void OnPropertyChanged(string propertyName) =>

PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));

}

<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

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

x:Class="ExampleBinding.TwowayPage"

Title="TwoWayPage">

<StackLayout Padding="10">

<Entry Text="{Binding Text, Mode=TwoWay}" />

<Label Text="{Binding Text}" />

</StackLayout>

</ContentPage>

namespace ExampleBinding;

public partial class TwowayPage : ContentPage

{

public TwowayPage()

{

InitializeComponent();

BindingContext = new TwowayViewModel();

}

}

using System.ComponentModel;

using System.Runtime.CompilerServices;

namespace ExampleBinding;

public class TwowayViewModel : INotifyPropertyChanged

{

private string \_text;

public string Text

{

get => \_text;

set

{

if (\_text != value)

{

\_text = value;

OnPropertyChanged();

}

}

}

public event PropertyChangedEventHandler PropertyChanged;

protected void OnPropertyChanged([CallerMemberName] string propertyName = null) =>

PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));

}

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

<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

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

x:Class="ExampleBinding.OneWayToSourcePage"

Title="OneWayToSourcePage">

<StackLayout Padding="10">

<!-- Binding OneWayToSource: UI sends data to the ViewModel -->

<Entry Text="{Binding EntryTextInput, Mode=OneWayToSource}" />

<!-- The Label is automatically updated from the ViewModel -->

<Label Text="{Binding DisplayText}"

VerticalOptions="CenterAndExpand"

HorizontalOptions="CenterAndExpand" />

</StackLayout>

</ContentPage>

namespace ExampleBinding;

public partial class OneWayToSourcePage : ContentPage

{

public OneWayToSourcePage()

{

InitializeComponent();

BindingContext = new OneWayToSourceViewModel();

}

}

using System.ComponentModel;

using System.Runtime.CompilerServices;

public class OneWayToSourceViewModel : INotifyPropertyChanged

{

private string entryTextInput;

public string EntryTextInput

{

get => entryTextInput;

set

{

if (entryTextInput != value)

{

entryTextInput = value;

OnPropertyChanged();

OnPropertyChanged(nameof(DisplayText));

}

}

}

public string DisplayText => EntryTextInput;

public event PropertyChangedEventHandler PropertyChanged;

protected void OnPropertyChanged([CallerMemberName] string name = null) =>

PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(name));

}

public ICommand MyCommand { get; }

public AmountViewModel()

{

MyCommand = new Command(AddNumbers);

}

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

<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

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

x:Class="ExampleBinding.BindingContextPage"

Title="BindingContextPage">

<StackLayout Padding="20">

<Entry Placeholder="Enter number 1"

Keyboard="Numeric"

Text="{Binding NumberOne}" />

<Entry Placeholder="Enter number 2"

Keyboard="Numeric"

Text="{Binding NumberTwo}" />

<Button Text="add numbers"

Command="{Binding MyCommand}"/>

<Label Text="{Binding Result, StringFormat='Result: {0:N2}'}"

FontSize="24"

HorizontalOptions="Center" />

</StackLayout>

</ContentPage>

using System.Globalization;

namespace ExampleBinding

{

public class BoolToStringConverter : IValueConverter

{

public object? Convert(object? value, Type targetType, object? parameter, CultureInfo culture)

{

if (value is bool booleanValue)

{

return booleanValue ? "Yes" : "No";

}

return "No";

}

public object? ConvertBack(object? value, Type targetType, object? parameter, CultureInfo culture)

{

throw new NotImplementedException();

}

}

}

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

<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

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

x:Class="ExampleBinding.BindingContextPage"

xmlns:local="clr-namespace:ExampleBinding"

Title="BindingContextPage">

<ContentPage.Resources>

<ResourceDictionary>

<local:BoolToStringConverter x:Key="BoolToString" />

</ResourceDictionary>

</ContentPage.Resources>

<StackLayout Padding="20">

<Label Text="{Binding IsActive, Converter={StaticResource BoolToString}}" />

</StackLayout>

</ContentPage>