using CommunityToolkit.Mvvm.ComponentModel;

namespace ExampleBinding

{

internal class EmployeeViewModel : ObservableObject

{

private string? \_employeeName;

private double \_employeeAge;

public string? EmployeeName

{

get => \_employeeName;

set => SetProperty(ref \_employeeName, value);

}

public double EmployeeAge

{

get => \_employeeAge;

set => SetProperty(ref \_employeeAge, value);

}

public EmployeeViewModel()

{

this.EmployeeName = ".NET MAUI";

this.EmployeeAge = 2.9;

}

}

}

public partial class EmployeeViewModel : ObservableObject

{

[ObservableProperty]

public string? \_employeeName;

[ObservableProperty]

public double \_employeeAge;

public EmployeeViewModel()

{

EmployeeName = ".NET MAUI";

EmployeeAge = 2.9;

}

}

public class EmployeeViewModel : ObservableObject

{

public IRelayCommand MyCommand { get; set; }

public EmployeeViewModel()

{

MyCommand = new RelayCommand(TextValue);

}

private void TextValue()

{

// Implement TextValue logic here }

}

}

public class ExampleViewModel : **INotifyPropertyChanged**

{

private double \_numberOne;

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

public double NumberOne

{

get => \_numberOne;

set

{

if (value != \_numberOne)

{

\_numberOne = value;

**OnPropertyChanged();**

}

}

}

//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));

}

}

public class ExampleViewModel : INotifyPropertyChanged

{

public ICommand MyCommand { get; }

public ExampleViewModel ()

{

MyCommand = new Command(AddNumbers);

}

private void AddNumbers()

{

Result = \_numberOne + \_numberTwo;

}

}

<StackLayout>

<Label x:Name="MyLabel" Text="Default Text" />

<Entry x:Name="MyEntry" />

<Button Text ="Clicked" Clicked="OnChangeLabel" />

</StackLayout>

namespace ExampleBinding;

public partial class Example : ContentPage

{

public Example()

{

InitializeComponent();

MyLabel.Text = "Hello, World!";

MyEntry.TextChanged += (s, e) =>

{

MyLabel.Text = e.NewTextValue;

};

}

public void OnChangeLabel(object sender, EventArgs e)

{

MyLabel.Text = "Hello, World!";

}

}