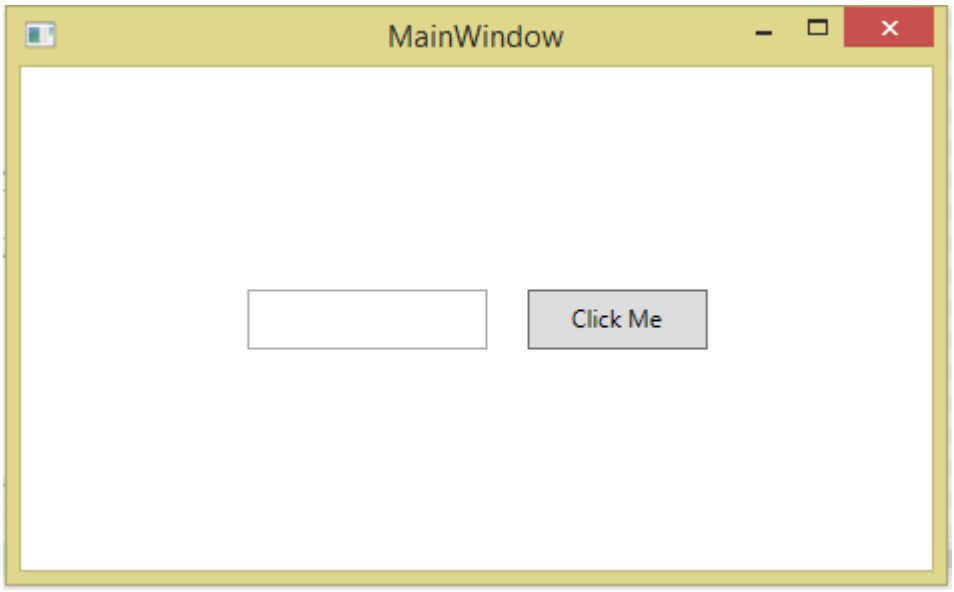
2 2 20.9k

MVVMICommand.zip
Download Free Office API

In this article, we will learn about button click event in WPF with MVVM concept. We use ICommand interface for generating the button click event. Let's take one simple example of one simple textbox and button. After writing the text in textbox, when we click on button, it will display a simple message box.

Before staring this article, please read Explain INotifyPropertyChanged In WPF - MVVM article first.

First, create one WPF application and create a window like the following.



XAML code

```
<Window x:Class="MVVM_ICommand.MainWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/pre
    <StackPanel Orientation="Horizontal" VerticalAlignment="Center" HorizontalAlignment="Center">
        <TextBox Width="120" Height="30" Margin="10"></TextBox>
        <Button Width="90" Margin="10" Content="Click Me"></Button>
    </StackPanel>
</Window>
```

implementation is given in my previous article.

Now, let's start our main code for making a new class RelayCommand. Make it as public.

```
public class RelayCommand {}

Now implement the interface ICommand
public class RelayCommand: ICommand {
    public boolCanExecute(object parameter) {

    }
    public event EventHandlerCanExecuteChanged;
    public void Execute(object parameter) {

    }
}
```

It will create two methods and one event. The first method "CanExecute" decides whether we are allowed to fire the command (the button click event) or not. The second method "Execute" method contains the actual logic. If CanExecute method returns true, then Execute method is run.

Now, Create two action properties that we initialize in constructor of the relay command class.

```
Action < object > _exectеMethod;
Func < bool, object > _canexecuteMethod;

public RelayCommand(Action < object > exectеMethod, Func < bool, object > canexecuteMethod) {
    _exectеMethod = exectеMethod;
    _canexecuteMethod = canexecuteMethod;
}
```

Here, in CanExecute method, we put null validation to check if _canexecutemethod is initialized or not. If not, then it returns false; otherwise it returns true. Also, write the logic for Execute method. Here, we simply initialize the _execute method because we make this RelayCommand class to be generalized.

```
public boolCanExecute(object parameter) {
    if (_canexecuteMethod != null) {
        return true;
    } else {
        return false;
    }
}


public void Execute(object parameter) {
    _exectеMethod(parameter);
}
```

Now, let's define "CanExecuteChanged" the event logic, which will run the canExecute method continuously.

```
public event EventHandlerCanExecuteChanged
{
add { CommandManager.RequerySuggested += value; }
remove { CommandManager.RequerySuggested -= value; }
}
```

```
public class RelayCommand: ICommand
Action < object > _exectеMethod;
Func < bool, object > _canexecuteMethod;
```

```
    _canexecuteMethod = canexecuteMethod;
}

public boolCanExecute(object parameter) {
    if (_canexecuteMethod != null) {
        return _canexecutemethod(parameter);
    } else {
        return false;
    }
}

public event EventHandlerCanExecuteChanged {
    add {
        CommandManager.RequerySuggested += value;
    }
    remove {
        CommandManager.RequerySuggested -= value;
    }
}

public void Execute(object parameter) {
    _exectеMethod(parameter);
}
}
```

Now, go for the View-Model class and create the ICommand property, as shown below.

public ICommandMyCommand { get; set; }

And, also create two methods like,

```
private boolCanExecuteMyMethod(object parameter) {
    if (string.IsNullOrEmpty(Name)) {
        return false;
    } else {
        if (Name < > "") {
            return true;
        } else {
            return false;
        }

    }
}

private void ExecuteMyMethod(object parameter) {
    MessageBox.Show("Hello... " + Name);
}
```

Now, create constructor for View-Model and pass both these methods to it.

```
public ViewModel()
{
MyCommand = new RelayCommand(ExecuteMyMethod, CanExecuteMyMethod);
}
```

Full View-Model Code

```
        get;
        set;
    }


    public string Name {
        get {
            return _name;
        }
        set {
            _name = value;
        }
    }

    public ViewModel() {
        MyCommand = new RelayCommand(ExecuteMyMethod, CanExecuteMyMethod);
    }


    private boolCanExecuteMyMethod(object parameter) {
        if (string.IsNullOrEmpty(Name)) {
            return false;
        } else {
            if (Name < > "") {
                return true;
            } else {
                return false;
            }

        }
    }

    private void ExecuteMyMethod(object parameter) {
        MessageBox.Show("Hello... " + Name);
    }

}
```

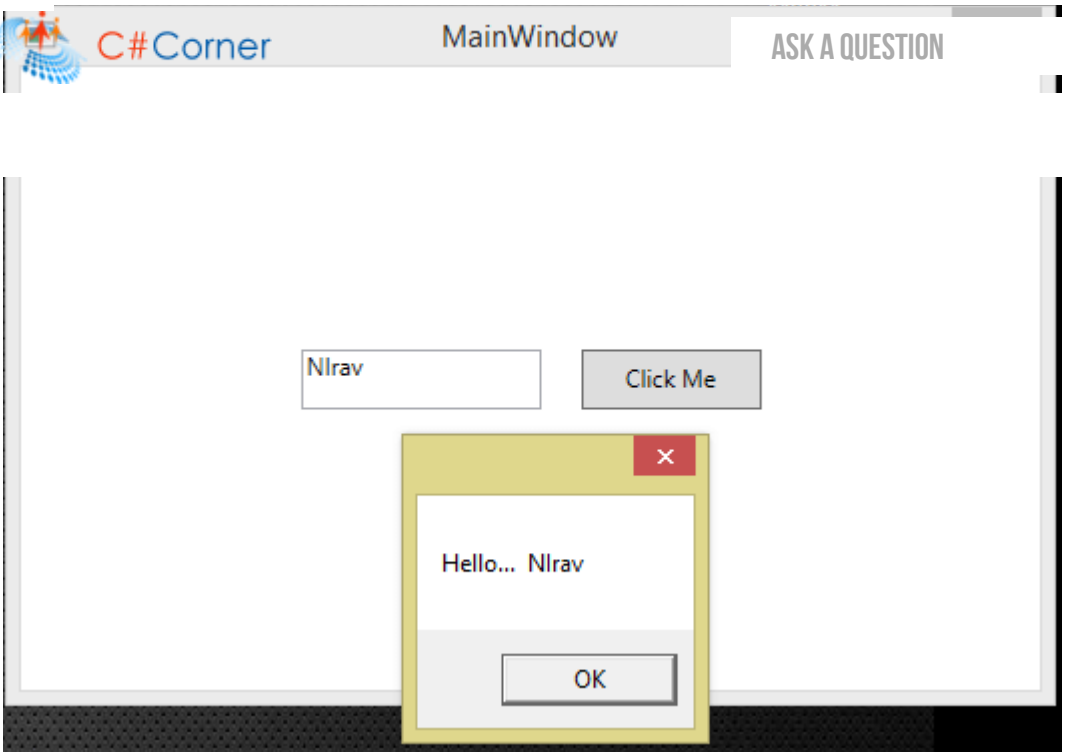
Now, go for the design View. Build the Solution and open the View's XAML code. Set the command property and give command a name.

<Button Width="90" Margin="10" Content="Click Me" Command="{Binding MyCommand}"></Button>

Full XAML Code

```
<Window x:Class="MVVM_ICommand.MainWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/pre
namespace:MVVM_ICommand.ViewModel" Title="MainWindow" Height="350" Width="525">
    <Window.Resources>
        <viewmodel:ViewModel x:Key="vm"></viewmodel:ViewModel>
    </Window.Resources>
    <StackPanel Orientation="Horizontal" VerticalAlignment="Center" HorizontalAlignment="Center" D
{Binding Source={StaticResourcevm}}">
        <TextBox Width="120" Height="30" Margin="10" Text="
{Binding Path=Name , Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"></TextBox>
        <Button Width="90" Margin="10" Content="Click Me" Command="{Binding MyCommand}">
</Button>
    </StackPanel>
</Window>
```

Run the application.



This is how this command interface works in MVVM . If you have any confusion, you can download the attached source code and debug it line by line.

ICommand MVVM WPF

Nirav Daraniya TOP 500

I have experienced in Microsoft technology like Windows application,Windows services,WPF,MVVM,WCF services and also in IOT(Internet of things) And I have also small experienced in developing and customization of Microsof... Read more

http://www.codeoverflow.net

372 **668.5k** **1**

2 **2**

Type your comment here and press Enter Key (Minimum 10 characters)

Nice post..

Prasanna Murali

192 **8.3k** **400.4k**

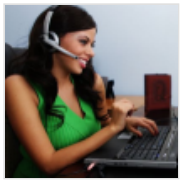
Oct 09, 2016

1 0 Reply

Good..

ketan borsdiya

Oct 06, 2016



TRENDING UP

01 ASP.NET Web API Using MVC And jQuery To Upload And Download Files - Part Seven

02 ASP.NET Web API Using MVC And HttpClient To Upload And Download Files - Part Eight

03 Building A Blockchain In .NET Core - Basic Blockchain

04 Create Your First Azure Machine Learning Workspace

05 Introduction To Blockchain, Ethereum And Smart Contracts

06 ASP.NET Core Blazor CRUD Using Entity Framework And Web API

07 What Is ML.NET

08 Best Practices Of Writing C# Code

09 Comparison Between React And Angular

10 Azure Machine Learning Client And Server Applications

View All

cus

osoft Announces Visual Studio 2019