**Step 1: Create a Silverlight Application in Viual studio 2010 and name it as “simplePersonandageMVVM”**



**Step 2: Once created add a class file and call it as “PersonModel.cs” and paste this code . (this would be the model thats is data object)**

using System.ComponentModel;

namespace simplePersonandageMVVM

{

 public class PersonModel : INotifyPropertyChanged

 {

 string name;

 public string Name

 {

 get { return this.name; }

 set

 {

 this.name = value;

 fire("Name");

 }

 }

 int age;

 public int Age

 {

 get { return this.age; }

 set

 {

 this.age = value;

 fire("Age");

 }

 }

 public PersonModel() { }

 public PersonModel(string name, int age)

 {

 this.name = name;

 this.age = age;

 }

 public void fire(string x)

 {

 if (PropertyChanged != null)

 {

 PropertyChanged(this, new PropertyChangedEventArgs(x));

 }

 }

 public event PropertyChangedEventHandler PropertyChanged;

 }

}

**Step 3: Create another class call it as “PersonViewModel.cs” This would be the viewModel and will take the role of an adapter between view(presentation layer) and the Model(entity class).**

namespace simplePersonandageMVVM

{

 public class PersonViewModel

 {

 public PersonModel p { get; set; }

 public PersonViewModel()

 {

 p = new PersonModel("prabjot", 20);

 }

 public ICommand GetPerson

 {

 get { return new GetPersonCommand(this); }

 }

 public void Increaseage(PersonModel d)

 {

 d.Age++;

 string x = d.Age.ToString();

 MessageBox.Show(x);

 }

 }

}

**Step 4: Create a Command object class implementing the ICOMMAND interface and call it as “GetPersonCommand.cs”. This class overrides both the Icommand methods. Here the method CanExecute() checks for the condition if its met then only the button click is enable. Where as the other method Execute() takes care of execution on button click.**

namespace simplePersonandageMVVM

{

 public class GetPersonCommand : ICommand

 {

 PersonViewModel pincommand;

 public GetPersonCommand( PersonViewModel Pcame)

 {

 pincommand= Pcame;

 }

 public bool CanExecute(object parameter)

 {

 if(pincommand.p.Age > 25)

 {

 return false ;

 }

 else

 {

 return true;

 }

 }

 public event EventHandler CanExecuteChanged;

 public void Execute(object parameter)

 {

 pincommand.Increaseage(pincommand.p);

 }

 }

}

**Step 5: Finally your xaml file looks like this**

<UserControl x:Class="simplePersonandageMVVM.MainPage"

 xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

 xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

 xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

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

 xmlns:local="clr-namespace:simplePersonandageMVVM"

 mc:Ignorable="d"

 d:DesignHeight="300" d:DesignWidth="400">

 <UserControl.Resources>

 <local:PersonViewModel x:Key="pkey" />

 </UserControl.Resources>

 <Grid x:Name="LayoutRoot" Background="White"

 DataContext="{Binding Source={StaticResource pkey}}" >

 <Grid Name="hi" DataContext="{Binding Path=p, Mode=TwoWay}">

 <TextBox Height="23" HorizontalAlignment="Left" Margin="53,30,0,0"

 Name="textBox1" VerticalAlignment="Top" Width="120" Text="{Binding Path=Name, Mode=TwoWay}" />

 <TextBox Height="23" HorizontalAlignment="Left" Margin="53,68,0,0" Name="textBox2"

 Text="{Binding Path=Age, Mode=TwoWay}" VerticalAlignment="Top" Width="120" />

 <Button Content="Button" Height="23" HorizontalAlignment="Left" Margin="53,112,0,0" Name="button1"

 VerticalAlignment="Top" Width="75" Command="{Binding Path=DataContext.GetPerson, ElementName= LayoutRoot }"

 CommandParameter="{Binding Path=Age, ElementName=hi}" />

 </Grid>

 </Grid>

</UserControl>

Please note the videos series on Silverlight including video for this example with explanation on command pattern plus other MVVM pattern examples with Data base and on wcf and linq with entity , is available on [www.elearningfromhome.com](http://www.elearningfromhome.com)



Button Base.Command connects view with ViewModel (GetPerson returntype is Icommand—comand object)- following Commn design pattern

**VIEW**



Action on Button click

Validation of condition for button click

GetPersonCommand (command object) Constructor connects PersonViewModel

**VIEW Model**

PersonViewModel contructor connects to PersonModel



Property changed event results in UI updates – from context (two way binding)

**Model**