ObservablePropertyChange

```
using System.ComponentModel;
using System.Runtime.CompilerServices;

namespace UMSTasks
{
   public class ObservablePropertyChange : INotifyPropertyChanged
   {
         public event PropertyChangedEventHandler PropertyChanged;
         protected void NotifyPropertyChanged([CallerMemberName] string propertyName = "")
         {
               PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));
         }
   }
}
```

StatusViewModel

```
public class StatusViewModel :  ObservablePropertyChange
   {
         #region Variables
         private bool _IsConnectedToServer = false;
         public bool ConnectedToServer {
               get { return _IsConnectedToServer; }
               set {
                        if (value != _IsConnectedToServer)
                              { this._IsConnectedToServer = value;
                                    NotifyPropertyChanged();
                        }
               }
         }

         #endregion

         public StatusViewModel()
         {
               TestServerConnection();
         }

         private void TestServerConnection()
         {
               var statusModel = new StatusModel();
               var pingTestSuccess = statusModel.ServerPing();
               var taskSvcTestSuccess = statusModel.TaskServiceConnected();

               if (pingTestSuccess & taskSvcTestSuccess)
               {
                     ConnectedToServer = true;

               }

         }// private void TestServerConnection()


   }// public class StatusViewModel : EventArgs
}
```

StatusControl

```
public partial class StatusControl : UserControl
   {
         #region Variables

         public string _serverName = "";
         internal string _userName = null;
         internal string _pwd = null;
         internal const string domain = "COL";
         StatusViewModel svm = new StatusViewModel();
         #endregion
         //ctor
         public StatusControl()
         {
               InitializeComponent();

               this.DataContext = svm;
               svm.PropertyChanged += UpdateLED;

         }

         //StatusViewModel event is true
         public void UpdateLED(object sender, PropertyChangedEventArgs e )
         {
               if (e.Equals(true))
               {
                     LED.Style = (Style)Application.Current.FindResource("GreenLED");

               }else { LED.Style = (Style)Resources["RedLED"]; }

         }


   }// public partial class StatusControl : UserControl
}//namespace UMSTasks.Views.UserControls
```

```
<UserControl x:Class="UMSTasks.Views.UserControls.StatusControl"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
             xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
             xmlns:svm="clr-namespace:UMSTasks.ViewModels"
             xmlns:local="clr-namespace:UMSTasks.Views.UserControls"
             mc:Ignorable="d"
             d:DesignHeight="45" d:DesignWidth="800"

             >
    <Border BorderBrush="#2b5797" BorderThickness="1">
        <Grid Height="30" Margin="0.5" x:Name="statusPanelGrid">
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="2" ></ColumnDefinition>
                <ColumnDefinition Width="*"></ColumnDefinition>
                <ColumnDefinition Width="*"></ColumnDefinition>
                <ColumnDefinition Width="*"></ColumnDefinition>
                <ColumnDefinition Width="*"></ColumnDefinition>
                <ColumnDefinition Width="*"></ColumnDefinition>
                <ColumnDefinition Width="*"></ColumnDefinition>
                <ColumnDefinition Width="*"></ColumnDefinition>
                <ColumnDefinition Width="*"></ColumnDefinition>
                <ColumnDefinition Width="*"></ColumnDefinition>
            </Grid.ColumnDefinitions>
            <Grid.RowDefinitions>
                <RowDefinition Height="25"></RowDefinition>
            </Grid.RowDefinitions>
            <TextBlock x:Name="ConnectionLabel" Grid.Column="1" Grid.ColumnSpan="2"
Margin="0,0,5,0" HorizontalAlignment="Right"  Height="15">
                Server Connection
            </TextBlock>
            <Ellipse x:Name="LED" Grid.Column="3" Margin="1" HorizontalAlignment="Left"
VerticalAlignment="Center" Style="{DynamicResource RedLED}"   />
        </Grid>
    </Border>
</UserControl>
```